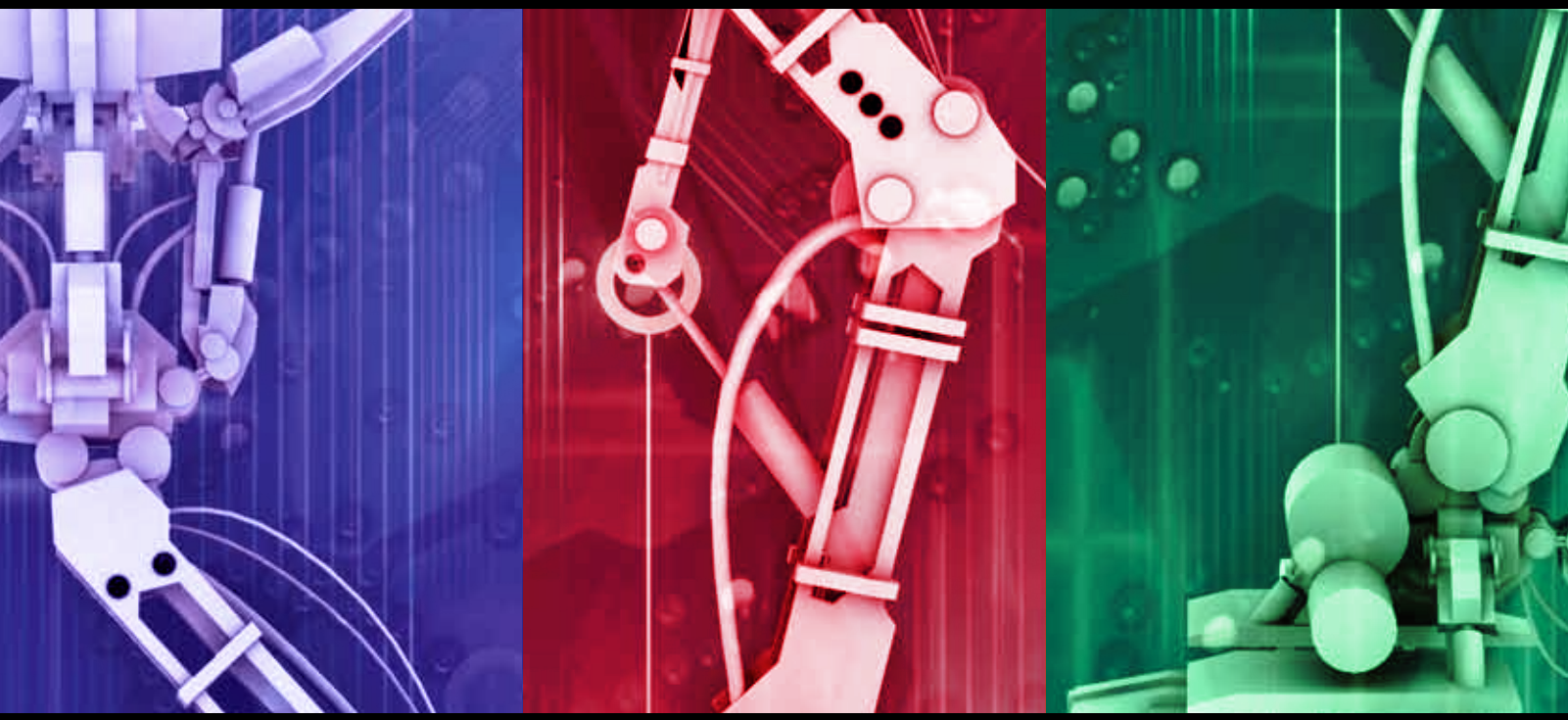
Journal of Robotics

Cognitive and Neural Aspects in Robotics with Applications

Guest Editors: Madan M. Gupta, Noriyasu Homma, Zeng-Guang Hou, and Ivo Bukovsky



Cognitive and Neural Aspects in Robotics with Applications

Journal of Robotics

Cognitive and Neural Aspects in Robotics with Applications

Guest Editors: Madan M. Gupta, Noriyasu Homma, Zeng-Guang Hou, and Ivo Bukovsky

Editorial Board

Contents

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 623140, 2 pages
doi:10.1155/2010/623140

Editorial

Cognitive and Neural Aspects in Robotics with Applications

Madan M. Gupta,[1] Noriyasu Homma,[2] Zeng-Guang Hou,[3] and Ivo Bukovsky[4]

[1] *Intelligent Systems Research Laboratory, College of Engineering, University of Saskatchewan, SK, Canada S7N 5A9*
[2] *Tohoku University, Sendai, Japan*
[3] *Chinese Academy of Sciences, Beijing, China*
[4] *Czech Technical University, Prague, Czech Republic*

Correspondence should be addressed to Madan M. Gupta, madan.gupta@usask.ca

Received 25 August 2010; Accepted 25 August 2010

With the evolution of our complex technological society and the introduction of new notions and innovative theoretical tools such as *cognitive and neural aspects* in the field of robotics systems, this has brought some new evolutions. These evolving and theoretical tools are providing some intelligence and robustness in robotics systems similar to those found in biological species.

Cognition and intelligence—the ability to learn, understand, and adapt—is the creation of nature, and it plays a key role in human actions and in many other biological species. Humans possess some robust attributes of learning and adaptation, and that is what makes them so intelligent. We humans react through the process of learning and adaptation on information received through a widely distributed network of sensors and control mechanisms in our body. The *faculty of cognition,* which is contained in our *carbon-based computer*, the brain, acquires information from the environment through various sensory mechanisms such as vision, hearing, touch, taste, and smell. Then, the process of cognition, through its intricate neural networks—the *cognitive computing*—integrates this information and provides appropriate actions. The cognitive process then advances further towards some attributes such as learning, recollection, reasoning, and control.

We are learning from the carbon-based cognitive computer—*the brain*—and now are in the process of inducing two important aspects of perception and cognition, and thereby that of the intelligence into robotics machines. One of our aims is to construct a robotic vehicle that can think and operate in uncertain and unstructured driving conditions. Robots in manufacturing, mining, agriculture, space and ocean exploration, and health sciences are just a few examples of these challenging applications where humanistic attributes such as cognition and intelligence can play an important role.

The proposal for the special issues of the robotics journal devoted to *cognitive and neural aspects* was conceived in early 2009, and now we are pleased to present 18 invited research papers which cover a wider aspect of cognition and intelligence. Initially, we received 29 research papers, but after going through a thorough review process, we have accepted only 18 papers. These papers cover some wider aspects of cognition and intelligence in the field of robotics, and for this special issue of the Robotics Journal, we have divided them into the following three parts.

In the first part, the first six papers (1–6) cover the fields of cognition, perception, vision, and neural learning in robotics. In the second part, the second six papers (7–12) deal with neuro-control in robotics. Finally, in the third part, the last six papers (7–18) focus on miscellaneous robotics applications.

Thus, as can be seen, these 18 invited papers represent a broad cross-section of the robotics field and have used some cognitive and neural aspects in the design of learning and control algorithms. Also, these papers have been authored by 55 researchers in robotics field from 29 different research institutions or universities located in seven different countries (Austria (1), Canada (1), China (4), India (1), Japan (8), Sweden (2), and USA (3)).

We guest editors believe that this special issue of the Journal of Robotics containing 18 invited research papers authored by international researchers and devoted to the

various aspects of cognition, perception, vision and neural learning, and neuro-control in robotics with various industrial applications is an informative and useful addition to the field of robotics.

Acknowledgments

We the Guest Editors would like to express their sincere appreciation to the Editorial Board of the Journal of Robotics for their confidence and unwavering support for this special issue on "Cognitive and Neural Aspects in Robotics with Applications." We acknowledge the efforts of the authors for their valuable research contributions to this special issue of the journal, and that of the reviewers who adhered to the strict timeline for making this special issue a success.

Madan M. Gupta
Noriyasu Homma
Zeng-Guang Hou
Ivo Bukovsky

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 130285, 16 pages
doi:10.1155/2010/130285

Research Article

Robotic Vision with the Conformal Camera: Modeling Perisaccadic Perception

Jacek Turski

Department of Computer and Mathematical Sciences, University of Houston-Downtown, One Main Street, Houston, TX 77002, USA

Correspondence should be addressed to Jacek Turski, turskij@uhd.edu

Received 21 September 2009; Revised 17 January 2010; Accepted 8 February 2010

Academic Editor: Noriyasu Homma

Humans make about 3 saccades per second at the eyeball's speed of 700 deg/sec to reposition the high-acuity fovea on the targets of interest to build up understanding of a scene. The brain's visuosaccadic circuitry uses the oculomotor command of each impending saccade to shift receptive fields (RFs) to cortical locations before the eyes take them there, giving a continuous and stable view of the world. We have developed a model for image representation based on projective Fourier transform (PFT) intended for robotic vision, which may efficiently process visual information during the motion of a camera with silicon retina that resembles saccadic eye movements. Here, the related neuroscience background is presented, effectiveness of the conformal camera's non-Euclidean geometry in intermediate-level vision is discussed, and the algorithmic steps in modeling perisaccadic perception with PFT are proposed. Our modeling utilizes basic properties of PFT. First, PFT is computable by FFT in complex logarithmic coordinates that also approximate the retinotopy. Second, the shift of RFs in retinotopic (logarithmic) coordinates is modeled by the shift property of discrete Fourier transform. The perisaccadic mislocalization observed by human subjects in laboratory experiments is the consequence of the fact that RFs' shifts are in logarithmic coordinates.

1. Introduction

In this article, we demonstrate that a mathematical data model we have developed for image representation intended for biologically-mediated machine vision systems [1–6] may be useful to process visual information during the motion of a camera with silicon retina that resembles saccadic eye movements. Our data model is based on the projective Fourier analysis that we have constructed in the framework of representation theory of the Lie group $\mathbf{SL}(2,\mathbb{C})$ by restricting the group representations to the image plane of the conformal camera—the camera with image projective transformations given by the action of $\mathbf{PSL}(2,\mathbb{C})$ [4, 5]. The analysis provides an efficient image representation that is well adapted to (a) the projective transformations of retinal images and (b) the retinotopic mappings of the brain's oculomotor and visual pathways. This latter assertion stems from the fact that the projective Fourier transform (PFT) is computable by a fast Fourier transform (FFT) algorithm in coordinates given by a complex logarithm that transforms PFT into the standard Fourier integral, while simultaneously approximating the local retinotopy [7]. Consequently, PFT of the conformal camera integrates the head, eyes, and retinotopic mapping of the visual pathways into a single computational binocular system [6]. As already suggested in [8], this integrated system may efficiently model visuomotor processes during saccadic eye movements that reposition the high-acuity fovea—the retinal region of a central angle subtending a US quarter an arm length from the eyes—on the targets of interest to build up understanding of a scene.

Humans make about three saccades per second at the eyeball's maximum speed of 700 deg/sec, producing about 200,000 saccades per day [9]. As we are not aware of these fast moving retinal images, the brain, under normal circumstances, suppresses visual sensitivity during saccadic eye movements and compensates for these incessant interruptions. This visual stability is maintained by the brain's widespread neural network [10]. Converging evidence from psychophysics, functional neuroimaging, and primate neurophysiology indicates that the most attractive neural basis that underlies visual stability is the mechanism causing visual cells in various visual and visuomotor cortical areas to

respond to stimuli that will fall in their receptive fields (RFs) before the eyes move them there, commonly referred to as the shifting RFs mechanism [11–15]. The identification of the visuosaccadic pathways (see references in [10]) supports the idea that the brain uses a copy of the oculomotor command of the impending saccade, referred to as efference copy or corollary discharge (see [16] for a review), to shift transiently the RFs of stimuli. This shift of RFs, starting 50 ms before a saccade onset and ending 50 ms after the saccade landing, is hypothesized to update (or remap) the retinotopic maps in the anticipation of each upcoming saccade. In fact, in a recent experiment [17], when human subjects shifted fixation to the clock, their reported time was earlier than the actual time on the clock by about 40 ms.

Although interruptions caused by saccades remain unnoticed in our daily life, in laboratory experiments, it becomes possible to probe the unexpected consequences of the saccadic eye movement. Specifically, laboratory experiments in lit environments have shown that briefly flashed probes around the saccade's onset are perceived as compressed toward the saccadic target [18–21], while, in total darkness, the probes' localizations are characterized by a uniform shift in the direction of the saccade [22–24]. The experimental studies [25–27] investigating the influence of the saccade's parameters on perisaccadic mislocalization showed that perisaccadic visual compression and the unidirectional shift are probably governed by different neural processes. Although the perisaccadic shift can be mainly explained by delays in the processing of visual information [24], the mechanism of perisaccadic compression, commonly related to the neural processes of the RF shift [28], remains relatively elusive.

In this article, we argue that the conformal camera's complex projective geometry and the related harmonic analysis (projective Fourier analysis) may be useful in perisaccadic perception. In particular, the image representation in terms of PFT may efficiently model the RFs shift that remaps cortical retinotopy in the anticipation of each saccade and the related phenomenon of perisaccadic perceptual space compression. During fixations the brain acquires visual information, resolving inconsistencies of the brief compression resulting from remapping. The computational significance of this remapping, when incorporated into neural engineering design of a foveate visual system, stems from the fact that it may integrate visual information from an object across saccades, eliminating the need for starting visual information processing anew three times per second at each fixation and speeding up a costly process of visual information acquisition [29]. This transfer of object features across saccadic eye movements [30–33] that is believed to maintain visual stability of transsaccadic perception is not considered here because, at present not much is known about the whole process [13].

This paper is organized as follows. We outline the neural processes of the visuosaccadic system involved in the preparation, execution, and control of the saccadic eye movement in Section 2 and later continue in Section 6. In Sections 3 and 4, we lay out the background that explains the mathematical tools we use in modeling human visual system. To this end, in Section 3, we introduce the conformal camera, discuss its conformal geometry, and evaluate the effectiveness of this geometry in the early- and intermediate-level vision computational aspects of natural scene understanding. Then, in Section 4, we show that the conformal camera possesses its own harmonic analysis—projective Fourier analysis—which provides image representation given in terms of the discrete PFT (DPFT) fast computable by FFT in coordinates given by a complex logarithm. Section 5 of this article deals with the implementation of the DPFT in retinocortical image representation that efficiently integrates the head, eyes, and retinotopic maps into one computational system. We also mention hardware setups that could be supported by DPFT-based software and compare conformal camera-based modeling to other approaches in foveate vision. Using this integrated visual system, in Section 6, we model perisaccadic perception, including the perisaccadic compression observed in psychophysical laboratory experiments. Finally, in Section 7, we discuss and compare our model with other numerical approaches to perisaccadic perception and discuss the directions in advancing our modeling. The paper is summarized in the last section.

2. The Visuosaccadic System

One of the most important functions of any nervous system is sensing the external environment and responding in a way that maximizes immediate survival chances. For this reason, the perception and action have evolved in mammals by supporting each other's functions. This functional link between visual perception and oculomotor action is well demonstrated in primates when they execute the eye-scanning movements (saccades) to overcome the eye's acuity limitation in building up the scene understanding.

In fact, humans can only see clearly the central part of the visual field of a 2 deg central angle. This region is projected onto the central fovea, where its image is sampled by the hexagonal mosaic of photoreceptors consisting mainly of cone cells, the color-selective type of photoreceptors for a sharp daylight vision. The visual acuity decreases rapidly away from the fovea because the distance between cones increases with eccentricity as they are outnumbered by rode cells, photoreceptors for a low-acuity black-and-white night vision. Moreover, there are a gradual loss of hexagonal regularity of the photoreceptor mosaic and an increased convergence of the photoreceptors on the ganglion cells whose axons carry visual information from the eye to the brain. For example, at 2.5 deg radius, which corresponds to the most visually useful region of the retina, acuity drops 50%. In Figure 1, (b) shows a progressively blurred image from (a), simulating the progressive loss of acuity with eccentricity.

With three saccades per second, the saccadic eye movement is the most common bodily movement. The eyes remain relatively still between consecutive saccades for about 180–320 ms, depending on the task performed. During this time period, the image is processed by the retinal circuitry and sent, mainly, to the visual cortex (starting with the

(a) (b) (c)

Figure 1: (a) San Diego skyline and harbor. (b) Progressively blurred image of (a) simulating the progressive loss of retinal acuity with eccentricity. The circle C_1 encloses the part of the scene projected onto the high-acuity fovea of a 2 deg diameter. The circle C_2 encloses the part projected onto the visually useful foveal region of a 5 deg diameter. (c) A scanning path the eyes may take to build up the scene understanding. Adapted from [34].

primary visual cortex, or V1, and reaching higher cortical areas, including the cognitive areas), with a minor part sent to the oculomotor midbrain areas. During the saccadic eye movement, the visual sensitivity is markedly reduced, although some modulations of low spatial frequencies (contrast and brightness) are wellpreserved or even enhanced [35]. This phenomenon is known as saccadic suppression. The sequence of saccades, fixations, and, often, smooth-pursuit eye movements for tracking a slowly moving small object in the scene, is called the scanpath and was first studied in [36]. In Figure 1(c) we depict the scanpath that eyes might actually take to build up understanding of the scene.

Although they are the simplest of bodily movements, the eyes' saccades are controlled by a widespread neural network that involves nearly every level of the brain. In Figure 2 we show the diagram of well-established pathways of the primate visuosaccadic system. They includes most prominently, the superior colliculus (SC) of the midbrain for representing possible saccade targets, and the parietal eye field (PEF) and the frontal eye field (FEF) in the parietal and frontal lobes of the neocortex (which obtains inputs from many visual cortical areas) for assisting the SC in the control of the involuntary (PEF) and voluntary (FEF) saccades. They also project to the simple neural circuits in the brainstem reticular formation (pons) in the midbrain that ensure the saccade's outstanding speed and precision. The course of events in the visuosaccadic system, which is based on [10], is outlined in the caption of Figure 2.

Although many of the neural processes involved in saccade generation and control are amenable to precise quantitative studies [37], some neural processes of the visuosaccadic system remain virtually unknown. The saccadic suppression, the fact that we do not see moving retinal images, is barely understood. There is accumulating evidence that viewers integrate information poorly across saccades during tasks such as reading, visual search, and scene perception [38]. This means that, three times per second at each fixation, there are instant large changes in the retinal images without almost any information consciously carried between fixations. Furthermore, because the next saccade target selection for the voluntary saccades takes place in

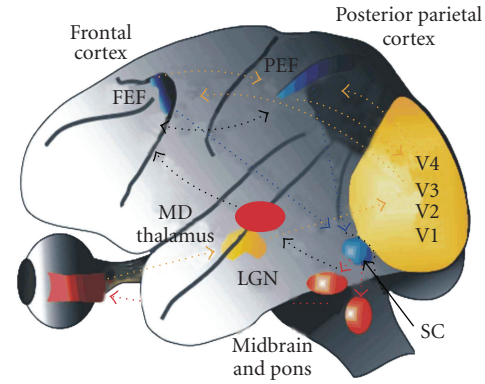


Figure 2: The visuosaccadic system. The course of events is the following. (1) *Fixations* (~300 ms). When eyes complete a saccade, visual information is acquired and sent via LGN to V1 and higher cortical areas including FEF and PEF (yellow paths). The next saccade's target is selected in FEF/PEF and its position is calculated in SC (blue paths) and the oculomotor command (OC) is prepared (Pons). However, before the saccade is executed, a copy of OC is sent via MD of Thalamus to FEF/PEF (black paths) to shift the receptive fields of the current fixation landmarks to the receptive fields of their future positions centered at the impending saccade's target. (2) *Saccades* (~30 ms). The OC executes the eyes' saccade with the motor neurons (red paths). The shift of receptive fields (starting ~50 ms before the saccade onset and ending ~50 ms after the saccade landing) gives access to the upcoming saccade's target before eyes arrival, integrating objects' features between fixations and maintaining stability of perception.

the higher cortical areas involving cognitive processes [39], the time needed for the oculomotor system to plan and execute the saccadic eye movement could take as long as 150 ms. Therefore, it is critical that visual information is efficiently acquired during each fixation without repeating much of the whole process since it would require too much computational resources. However, visual constancy, the fact that we are not aware of any discontinuity in the scene perception when executing the scanpath, is not perfect. About 50 ms before the onset of the saccade, during saccadic movement (~30 ms), and about 50 ms after the saccade,

the salient stimuli are not perceived in veridical locations. In particular, a transient compression around the saccade target, called perisaccadic mislocalization, is observed in lit laboratory experiments; see Section 1. We continue this discussion in Section 6 where we present the algorithm for modeling some of the neural processes underlying perisaccadic perception.

3. The Conformal Camera, Geometry, and Perception

We model the human eyes' imaging functions with the conformal camera; the name of which will be explained later. In the conformal camera, shown in Figure 3, the retina is represented by the image plane $x_2 = 1$ with complex coordinates $x_3 + ix_1$, on which a 3D scene is projected under the mapping

$$j(x_1, x_2, x_3) = \frac{(x_3 + ix_1)}{x_2}. \tag{1}$$

The implicit assumption $x_2 \neq 0$ will be removed later.

The image projective transformations are generated by the two basic transformations h and k shown in Figure 3. Both transformations have the form of linear-fractional transformations

$$z \longmapsto g \cdot z = \frac{\alpha z + \beta}{\gamma z + \delta} \tag{2}$$

with $\alpha\delta - \gamma\beta = 1$. Therefore, all finite iterations of the mappings h and k give the group $\mathbf{SL}(2, \mathbb{C})$ acting on the image plane of the conformal camera by linear-fractional, or Möbius, transformations

$$\mathbf{SL}(2, \mathbb{C}) \ni \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot z = \frac{dz + c}{bz + a}, \quad z = x_3 + ix_1 \equiv (x_1, 1, x_3). \tag{3}$$

Because $\pm\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ have the same action, we need to identify matrices in $\mathbf{SL}(2, \mathbb{C})$ that differ in sign. The result is the quotient group $\mathbf{PSL}(2, \mathbb{C}) = \mathbf{SL}(2, \mathbb{C})/\{\pm Id\}$, where Id is the identity matrix, and the action (3) establishes a group isomorphism between linear-fractional, or Möbius, mappings and $\mathbf{PSL}(2, \mathbb{C})$. Thus,

$$\mathbf{PSL}(2, \mathbb{C}) \ni g = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \longmapsto f(g^{-1} \cdot z) = f\left(\frac{az - c}{-bz + d}\right) \tag{4}$$

gives the image projective transformations of the intensity function $f(z)$.

3.1. Geometry of the Conformal Camera. In the homogeneous coordinate framework of projective geometry [40], the conformal camera is embedded into the complex plane

$$\mathbb{C}^2 = \left\{ \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \mid z_1 = x_2 + iy, z_2 = x_3 + ix_1 \right\}. \tag{5}$$

In this embedding, the "slopes" ξ of the complex lines $z_2 = \xi z_1$ can be numerically identified with the points on the extended image plane $\widehat{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$ where ∞ corresponds to the line $z_1 = 0$. In fact, if $x_2 \neq 0$ and $y = 0$, the slope ξ corresponds to the point $x_3 + ix_1$ at which the ray (line) in $\mathbb{R}^3$ that passes through the origin is intersecting the image plane of the conformal camera. Then, the standard action of the group $\mathbf{SL}(2, \mathbb{C})$, on nonzero column vectors $\mathbb{C}^2$,

$$\begin{pmatrix} z_1' \\ z_2' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} az_1 + bz_2 \\ cz_1 + dz_2 \end{pmatrix} \tag{6}$$

implies that the slope $\xi = z_2/z_1$ is mapped to the slope

$$\xi' = \frac{z_2'}{z_1'} = \frac{c + d\xi}{a + b\xi} \tag{7}$$

agreeing with the mappings in (3). However, the action (3) must be extended to include the line $z_1 = 0$ of "slope" ∞ as follows:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \infty = \frac{d}{b}, \qquad \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \left(-\frac{a}{b}\right) = \infty. \tag{8}$$

The stereographic projection $\sigma = j|_{S^2_{(0,1,0)}}$ (with j in (1)) establishes isomorphism $\widehat{\mathbb{C}} \cong S^2_{(0,1,0)}$ and $\sigma(0, 0, 0) = \infty$ gives a concrete meaning to the point ∞ such that it can be treated as any other point. The set $\widehat{\mathbb{C}}$ is referred to as the Riemann sphere and the group $\mathbf{PSL}(2, \mathbb{C})$ acting on $\widehat{\mathbb{C}}$ consists of the bijective meromorphic mappings of $\widehat{\mathbb{C}}$ [41]. Thus, it is the group of holomorphic automorphisms of the Riemann sphere that preserve the intrinsic geometry imposed by complex structure, known as Möbius geometry [42] or inversive geometry [43].

The mappings in (4) are conformal, that is, they preserve the oriented angles of two tangent vectors intersecting at a given point [41]. Because of this property, the camera is called "conformal". Although the conformal part of an image projective transformation can be removed with almost no computational cost, leaving only a perspective transformation of the image (see [4, 5]), the conformality provides an advantage in imaging because the conformal mappings rotate and dilate the image's infinitesimal neighborhoods, and, therefore, locally preserve the image pixels.

The image plane of the conformal camera does not admit a distance that is invariant under image projective (linear-fractional or Möbius) transformations. Therefore, geometry of the conformal camera does not possess a Riemannian metric; for instance, there is no curvature measure. It is customary in complex projective (Möbius or inversive) geometry to consider a line as a circle passing through the point ∞. Then, the fundamental property of this geometry can be expressed as follows: linear-fractional mappings map circles to circles [41]. Thus, circles can play the role of geodesics.

3.2. The Conformal Camera and the Intermediate-Level Vision. As discussed before, circles play a crucial role in the

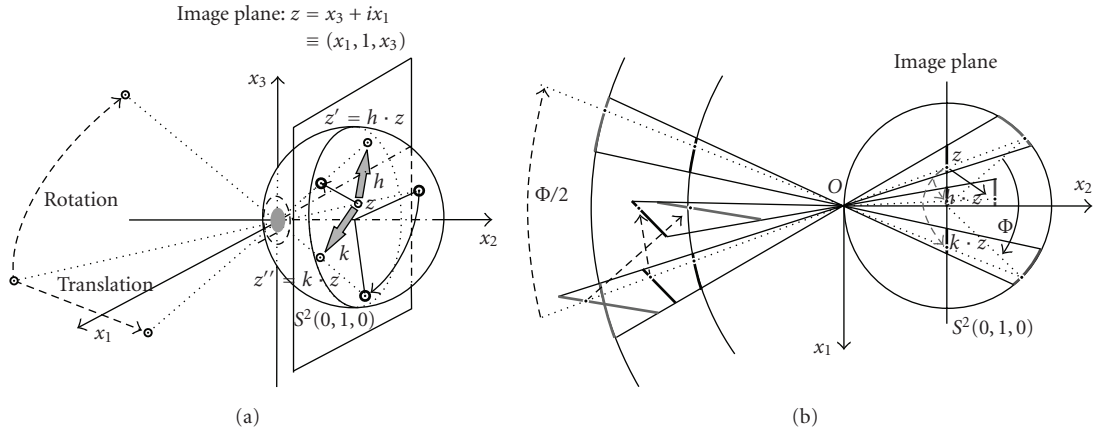


Figure 3: The conformal camera. (a) Image projective transformations are generated by iterations of transformations covering translations "h" and rotations "k" of planar objects in the scene. (b) The 2D section of the conformal camera further explains how image projective transformations are generated and how the projective degrees of freedom are reduced in the camera; one image projective transformation in the conformal camera corresponds to different planar objects translations and rotations in the 3D world.

conformal camera geometry and it should be reflected in psychological and computational aspects of natural scene understanding whether this camera is relevant to modeling primate visual perception.

Neurophysiological experiments demonstrate that the retina filters impinged images extracting local contrast spatially and temporally. For instance, center surround cells at the retinal processing stage are triggered by local spatial changes in intensity referred to as edges or contours. This filtering is enhanced in the primary visual cortex, the first cortical area receiving the retinal output. This area itself is a case study in dense packing of overlapping visual submodalities: motion, orientation, frequency (color), and oculomotor dominance (depth). In psychological tests, humans easily detect a significant change in spatial intensity (low-level vision), and effortlessly and unambiguously group this usually fragmented visual information (contours of occluded objects, for example), into coherent, global shapes (intermediate-level vision). Considering its computational complexity, this grouping is one of the difficult problems that primate visual system has to solve [44].

The Gestalt phenomenology and quantitative psychological measurements established the rules, summarized in the ideas of good continuation [45, 46] and association field [47], that determine interactions between fragmented edges such that they extend along continuous contours joining the edges in the way they will normally be grouped together to faithfully convey a scene meaning. Evidence accumulated in psychological and physiological studies suggests that the human visual system utilizes a local grouping process (association field) with two very simple rules: collinearity (receptive fields aligned along a line) and cocircularity (receptive fields aligned along a circle with the preferred orientation orthogonal to the tangents of the circle [48]) with underlying scale invariant statistics for both geometric arrangements in natural scenes. These rules were confirmed in [49, 50] by statistical analysis of natural scenes. Two basic intermediate-level descriptors that the brain employs in grouping elements into global objects are the medial axis transformation [51], or symmetry structure [52, 53], and the curvature extrema [54, 55]. In fact, the medial axis, which the visual system extracts as a skeletal (intermediate-level) representation of objects [56], can be defined as a set of the centers of maximal circles inscribed inside the contour. The curvatures at the corresponding points of a contour are given by the inverse radii of the circles.

This discussion shows that the conformal camera should effectively model the eye's imaging functions related to lower- and intermediate-level visions of natural scenes.

4. Projective Fourier Analysis

The projective Fourier analysis has been constructed by restricting geometric Fourier analysis of $\mathbf{SL}(2, \mathbb{C})$—a direction in the representation theory of the semisimple Lie groups [57]—to the image plane of the conformal camera [5, Section 5.1]. The resulting projective Fourier transform (PFT) of a given image intensity function $f(z) \in L^2(\mathbb{C})$ is the following:

$$\hat{f}(s,k) = \frac{i}{2} \int f(z) |z|^{-is-1} \left(\frac{z}{|z|} \right)^{-k} dz\, d\bar{z}, \tag{9}$$

where $(s, k) \in \mathbb{R} \times \mathbb{Z}$, and, if $z = x_3 + ix_1$, then $(i/2) dz\, d\bar{z} = dx_3\, dx_1$ is the Euclidean measure on the image plane.

In log-polar coordinates (u, θ) given by $\ln re^{i\theta} = \ln r + i\theta = u + i\theta$, $\hat{f}(k, s)$ takes on the form of the standard Fourier integral

$$\hat{f}(s,k) = \int \int f\left(e^{u+i\theta}\right) e^{u} e^{-i(us+\theta k)} du\, d\theta. \tag{10}$$

Inverting it, we obtain the representation of the image intensity function in the (u, θ)-coordinates:

$$e^{u} f(u,\theta) = \frac{1}{(2\pi)^2} \sum_{k=-\infty}^{\infty} \int \hat{f}(s,k) e^{i(us+\theta k)}\, ds, \tag{11}$$

where $f(u,\theta) = f(e^{u+i\theta})$. We stress that although $f(e^{u+i\theta})$ and $f(u,\theta)$ are numerically equal, they are given on different spaces.

4.1. Discrete Projective Fourier Transform. In spite of the logarithmic singularity of log-polar coordinates, PFT of any function f integrable on $\mathbb{C}^* = \mathbb{C} \setminus \{0\}$ is finite:

$$\begin{aligned}\left|\widehat{f}(s,k)\right| &\le \int_0^{2\pi}\int_{-\infty}^{u_1} f\left(e^{u+i\theta}\right)e^{u}\,du\,d\theta \\ &= \int_0^{2\pi}\int_0^{r_1} f\left(re^{i\theta}\right)dr\,d\theta < \infty.\end{aligned} \tag{12}$$

This observation is crucial in constructing the discrete PFT as follows. By removing a disk $|z| \le r_a$, we can regularize f such that the support of $f(u,\theta)$ is contained within $(\ln r_a, \ln r_b) \times [0, 2\pi)$ and approximate the integral in (10) by a double Riemann sum

$$\widehat{f}\left(\frac{2\pi m}{T}, n\right) \approx \frac{2\pi T}{NM}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} e^{u_k} f\left(e^{u_k}e^{i\theta_l}\right)e^{-2\pi i(mk/M+nl/N)} \tag{13}$$

with $M \times N$ partition points

$$\begin{aligned}(u_k,\theta_l) = \left(\ln r_a + \frac{kT}{M}, \frac{2\pi l}{N}\right), \quad 0 \le k \le M-1, \\ 0 \le l \le N-1,\ T = \ln(r_b/r_a).\end{aligned} \tag{14}$$

Then, introducing

$$f_{k,l} = \left(\frac{2\pi T}{MN}\right) f\left(e^{u_k}e^{i\theta_l}\right), \quad f_{k,l} = \left(\frac{2\pi T}{MN}\right) f(u_k,\theta_l) \tag{15}$$

and defining $\widehat{f}_{m,n}$ by

$$\widehat{f}_{m,n} = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} f_{k,l}e^{u_k}e^{-i2\pi mk/M}e^{-i2\pi nl/N}, \tag{16}$$

we obtain

$$f_{k,l} = \frac{1}{MN}\sum_{m=0}^{M-1N-1}\sum_{n=0}\widehat{f}_{m,n}e^{-u_k}e^{i2\pi mk/M}e^{i2\pi nl/N}. \tag{17}$$

Both expressions (16) and (17) can be computed efficiently by FFT algorithms.

On introducing complex coordinates $z_{k,l} = e^{u_k}e^{i\theta_l}$ into (16) and (17), these expressions are referred to as the discrete projective Fourier transform (DPFT) and its inverse, respectively [4, 5]. When "pixels" locations $e^{u_k}e^{i\theta_l}$ are transformed by the conformal camera's action of $g \in \mathbf{PSL}(2,\mathbb{C})$ so that the function undergoes projective transformation $f(g^{-1} \cdot e^{u_k}e^{i\theta_l}) = f(e^{u'_k}e^{i\theta'_l})$, its representation in (17) is given in terms of u'_k and θ'_l (instead of u_k, and θ_l) but with $\widehat{f}_{m,n}$ unchanged. We refer to the representation transformations as projectively adapted characteristics of the projective Fourier transforms [4, 5]. These projective transformations are not given explicitly here since they are not used in this work.

5. DPFT in Computational Vision

We discussed before the relevance of the conformal camera to the intermediate-level vision task of grouping image elements into individual objects in natural scenes. Here we discuss the relevance of the data model of image representation based on DPFT to image processing in biologically mediated machine vision systems.

5.1. Modeling the Retinotopy with DPFT. The mapping $w = \ln(z \pm a) - \ln a$, where $a > 0$ removes logarithmic singularity and $\pm a$ indicate, for different signs, the left or right brain hemisphere, is the accepted approximation of the retinotopic structure of primate visual cortical areas and the midbrain SC [7, 58]. However, the DPFT that provides the data model for image representation can be efficiently computed by FFT only in log-polar coordinates given by the complex logarithm $w = \ln z$. This mapping has distinctive rotational and zoom symmetries

$$\ln\left(e^{i\theta}z\right) = \ln z + i\theta, \quad \ln(\rho z) = \ln z + \ln\rho \tag{18}$$

important in image identification (rotations and dilations in the retinal space corresponding to translations in the cortical (log-polar) space). Thus, we see that the Schwartz model of the retina comes with drastic consequences by destroying the rotation and zoom symmetries.

The following facts support our modeling of retinotopy with DPFT. First, for small $|z| \ll a$, $\ln(z \pm a) - \ln a$ is approximately linear while, for large $|z| \gg a$, it is dominated by $\ln z$. Secondly, to construct discrete sampling for DPFT, the image is regularized by removing a disc representing the fovea, which is possible because PFT in log-polar coordinates does not have a singularity at the origin; see (12). Thirdly, there is accumulated evidence pointing to the fact that the fovea and periphery have different functional roles in vision [59, 60] and very likely involve different image processing underlying principles. Finally, by the split theory of hemispherical image representation, the foveal region has a discontinuity along the vertical meridian, with each half processed in a different brain hemisphere [61].

We conclude this discussion with the following remark. Both models discussed above, as well as all other similar models, are, in fact, fovea-less models [62]. However, because the fovea is explicitly removed in our model, we plan to complement it in the future by including the foveal image representation.

5.2. Image Sampling for DPFT. The DPFT approximation was obtained using the rectangular sampling grid (u_k,θ_l) in (14), corresponding, under the mapping

$$u_k + i\theta_l \longmapsto z_{k,l} = e^{u_k+i\theta_l} = r_k e^{i\theta_l}, \tag{19}$$

to a nonuniform sampling grid with equal sectors

$$\alpha = \theta_{l+1} - \theta_l = \frac{2\pi}{N}, \quad l = 0, 1, \ldots, N-1 \tag{20}$$

but with exponentially increasing radii

$$\rho_k = r_{k+1} - r_k = e^{u_{k+1}} - e^{u_k} = e^{u_k}\left(e^{\delta} - 1\right) = r_k\left(e^{\delta} - 1\right), \quad k = 0, 1, \ldots, M-1, \tag{21}$$

where $u_{k+1} - u_k$ is the spacing $\delta = T/M$ and $r_0 = r_a$, where r_a is the radius of the disc that has been removed to regularize logarithmic singularity. This sampling interface is shown in Figure 4.

Let us assume that we have been given a picture of the size $A \times B$ in pixel units, which is displayed with K dots per unit length (dpl). Then, the physical dimensions, in the chosen unit of length, of the pixel and the picture are $1/K \times 1/K$ and $A/K \times B/K$, respectively. Also, we assume that the retinal coordinates' origin (fixation) is the picture's center.

The central disc of radius r_0 represents the fovea with a uniform distribution of grid points, the number of the foveal pixels N_f given by $\pi r_0^2 = N_f/K^2$. This means that the fovea cannot increase the resolution, which is related to the distance of the picture from the eye. The number of sectors is obtained from the condition $2\pi(r_0 + r_1)/2 \approx N(1/K)$, where $N = [2\pi r_0 K + \pi]$. Here $[a]$ is the closest integer to a. To get the number of rings M, we assume that $\rho_0 = r_0(e^{\delta} - 1) = 1/K$ and $r_b = r_M = r_0 e^{M\delta}$. We can take either $r_b = (1/K)\min(A, B)/2$ or $r_b = (1/K)\sqrt{A^2 + B^2}/2$. Thus, $\delta = \ln[(1 + 1/r_0 K]$ and $M = (1/\delta)\ln(r_b/r_0)$.

Example 1. We let $A \times B = 512 \times 512$ and $K = 4$ per mm, so the physical dimensions in mm are 128×128 and $r_b = 128\sqrt{2}/2 = 90.5$. Furthermore, we let $N_f = 296$, so $r_0 = 2.427$ and $N = 64$. Finally, $\delta = \ln(10.7084/9.7084) \approx 0.09804$ and $(1/0.09804)\ln(90.5/2.427) \approx M = 37$. The sampling grid consists of points in polar coordinates: $(r_k + \rho_{k+1}/2, \theta_l + \pi/64) = (2.552e^{k0.09804}, (2l + 1)\pi/64)$ $k = 0, 1, \ldots, 36, l = 0, 1, \ldots, 63$.

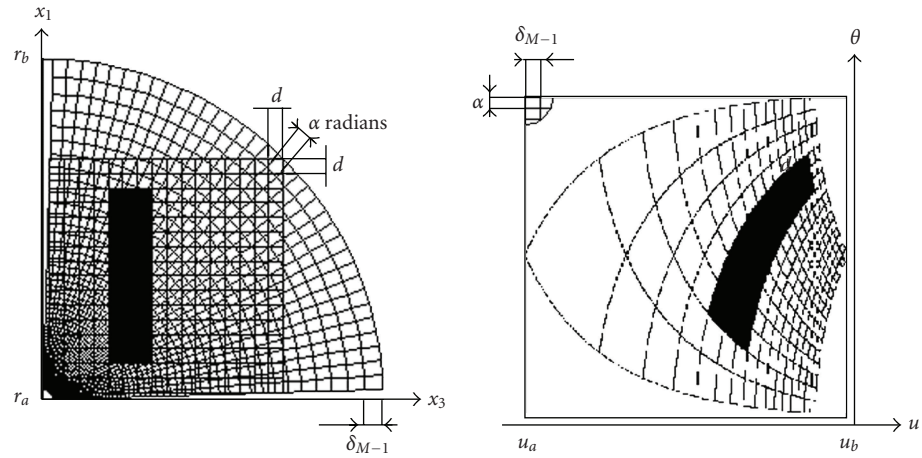
5.3. Imaging with the Conformal Camera. In the example from the previous section, the number of pixels in the original image is 262,144, whereas both foveal and peripheral representations of the image contain only 2,664 pixels. Thus, it results in about 100 times less pixels than in the original image. However, this reduction in the number of pixels comes at a price: not only does the small central region have the resolution required for a clear vision, it also has to be removed to regularize the logarithmic singularity. Therefore, the conformal camera with the DPFT-based image processing in the present state of development can support only the peripheral imaging functions of the visual system.

The most basic and frequent eye's imaging functions are connected with an incessant saccadic eye movement (about 200,000 saccades per day). The neural mechanisms of the RFs shifts and perisaccadic mislocalization, hypothesized to be involved in maintaining visual stability, are mainly supported by the peripheral visual processing. We use DPFT to model these phenomena in Section 6. The DPFT-based image representation could support the following hardware setup. A set of samples $\{f_{k,l}\}_{M\times N}$ of an image f, where $f_{k,l} = (2\pi T/MN) f(e^{u_k} e^{i\theta_l})$, is obtained from a camera with anthropomorphic visual sensors (silicon retina) [63] or from an "exp-polar" scanner with the sampling geometry similar to the distribution density of the retinal ganglion cells. The DPFT is applied to $\{f_{k,l}\}_{N\times M}$ according to (16), and $\widehat{f}_{k,l}$ is efficiently computed with FFT. Next, the IDPFT of $\widehat{f}_{k,l}$, given in (17), is again computed with FFT. However this output from IDPFT renders the retinotopic image $f_{k,l} = (2\pi T/MN) f(u_k, \theta_l)$ (numerically equal to $f_{k,l}$) of the retinal samples in the cortical log-polar coordinates.

When the eyes remain fixed, motion of objects is perceived by the successive stimulation of adjacent retinal loci. These image transformations are modeled in the conformal camera by the corresponding covariant transformations of the image representation in terms of DPFT; see the end of Section 4.1. These transformations are not important in modeling perisaccadic perception and are not dealt with in this work. However, they will become important in modeling smooth-pursuit eye movements, which we plan to undertake in the near future.

5.4. Other Approaches to Foveate Vision. Of the numerical approaches to foveate (also called space-variant) vision, involving, for example, Fourier-Mellin transform or log-polar Hough transform, the most closely related to our work are results reported by Schwartz' group at Boston University. We note that the approximation of the retinotopy by a complex logarithm was first proposed by Eric Schwartz in 1977. This group introduced the fast exponential chirp transform (FECT) [64] in their attempt to develop numerical algorithms for space-variant image processing. Both FECT and its inverse were obtained by the change of variables in the spatial and frequency domains applied to the standard Fourier integrals. The discrete FECT was introduced somehow ad hoc and some basic components of Fourier analysis, such as underlying geometry or Plancherel measure, were not considered. In comparison, projective Fourier transform (PFT) provides an efficient image representation well adapted to projective transformations produced in the conformal camera by the group $\mathbf{SL}(2, \mathbb{C})$ acting on the image plane by linear-fractional mappings. Significantly, PFT can be obtained by restricting geometric Fourier analysis of the Lie group $\mathbf{SL}(2, \mathbb{C})$ to the image plane of the conformal camera. Thus, the conformal camera comes with its own harmonic analysis. Moreover, PFT is computable by FFT in log-polar coordinates given by a complex logarithm that approximates the retinotopy. It implies that PFT can integrate the head, eyes, and visual cortex into a single computational system. This aspect is discussed, with special attention to perisaccadic perception, in the remaining part of the paper. Another advantage of PFT is the complex (conformal) geometric analysis underlying the conformal camera. We demonstrated in Section 3.2 the relevance of this geometry to the intermediate-level vision problem of grouping local contours into individual objects of natural scenes.

The other approaches to space-variant vision use the geometric transformations, mainly based on a complex



(a) (b)

Figure 4: The retinocortical sampling interface. (a) The exp-polar sampling (the distance between circles displayed in the first quadrant changes exponentially) of a bar pattern. (b) The bar pattern in the cortical coordinates rendered by the inverse DPFT computed with FFT. The cortical uniform sampling grid, obtained by applying the complex logarithm to the exp-polar grid in (a), is shown in the upper left corner.

logarithmic function between the nonuniform (retinal) sampling grid and the uniform (cortical) grid for the purpose of developing computer programs for problems in robotic vision. We give only a few examples of such problems: tracking [65], navigation [66], detection of salient regions [67], and disparity estimation [68]. However, in contrast to our projectively covariant image processing carried out with FFT, they share high computational costs in the geometric transformation process for dynamic scenes.

6. Perisaccadic Perception with DPFT

A sequence of fast saccadic eye movements is necessary to process the details of a scene by fixating the fovea successively on the targets of interest. Given the frequency of three saccades per second and limited computational resources, it is critical that visual information is efficiently acquired without starting anew much of this acquisition process at each fixation. This is critically important in robotic designs based on the foveate vision architecture (silicon retina), and in this section we propose the front-end algorithmic steps in addressing this problem.

The model of perisaccadic perception presented in this section is based on the theory in [29] that states (as is most classically assumed) that an efference copy generated by SC, a copy of the oculomotor command to rotate the eyes in order to execute the saccade, is used to briefly shift the flashed probes' RFs in FEF/PEF toward the cortical fovea. Because the shift occurs in logarithmic coordinates approximating retinotopy, the model can also explain observed in laboratory experiments perisaccadic compression shown in Figure 5.

We recall the time course of events (Figure 2) that we are going to model. During the eyes' fixation, lasting, on average, 300 ms, the retinal image is sampled by ganglion cells and sent to cortical areas, including higher areas in the parietal and the frontal lobes. In particular, the next saccade's target is selected in PEF/FEF areas and its position computed in subcortical SC area. About 50 ms before the onset of the saccade, during the saccade (~30 ms), and about 50 ms after the saccade, the visual sensitivity is reduced, and probes flashed around the impending saccade's target are not perceived in veridical locations; see Figure 5. Instead, a copy of the oculomotor command (efference copy) is used to translate the receptive fields of flashes recorded in the fovea-centered frame of the current fixation, remapping them into a target-centered frame. This internal remapping results in the illusory compression of flashes about the target. The cortical locations of the neural correlates of remapping are uncertain; it is required that these areas are retinotopically organized. These areas, of which most likely include PEF/FEF and V4 (and to a progressively lesser degree V3, V2, and V1), can be represented here by one retinotopic area [69].

6.1. The Model. The modeling steps are the following.

Step 1 (see Figure 6). The eye initially fixated at **F** is making the saccade to the target located at **T**. The four probes flashed around the upcoming saccade's target at **T** are projected into the retina and sampled by the photoreceptor/ganglion cells to give the set of samples $f_{m,n} = (2\pi T/MN) f(e^{u_m} e^{i\theta_n})$. Next, DPFT $\widehat{f}_{k,l}$ is computed by FFT in log-polar coordinates (u_k, θ_l), where $u_k = \ln r_k$. The inverse DPFT, computed again by FFT (the gray arrow), renders the image representation $f_{m,n} = (2\pi T/MN) f(u_m, \theta_n)$ in Cartesian log-polar coordinates—the four dots in (u, θ)-coordinates. The fovea, which is shown in yellow in Figure 6, is not included in log-polar coordinates, for these coordinates approximate only the extrafoveal part of the retina. For simplicity, we can take the radius of the fovea to be 1 so that the u-coordinate starts at 0.

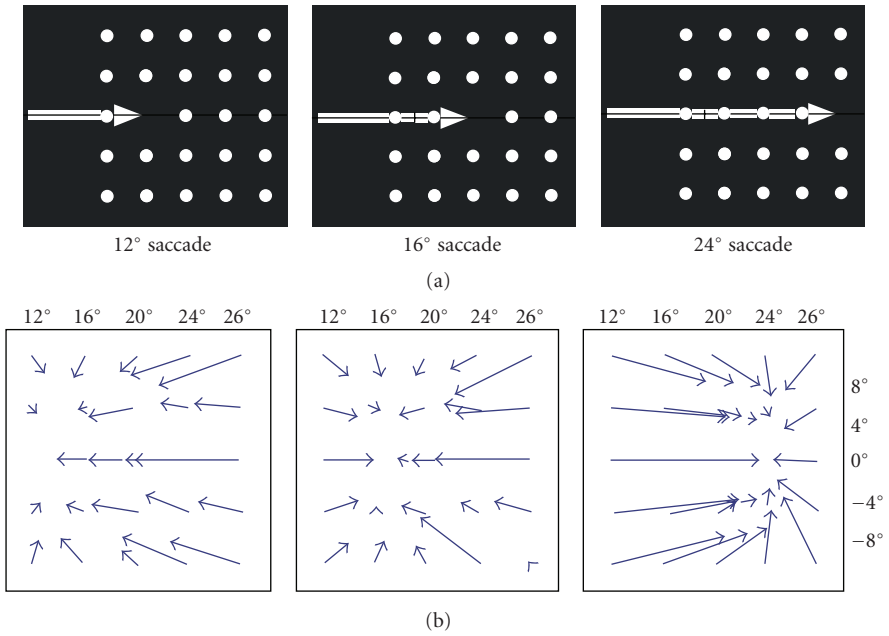


Figure 5: Experimental data (see [20]) of the absolute mislocalization (lower row) referenced to the true position of a flashed dot randomly chosen from an array of 24 dots and three different saccade amplitudes (upper row) are shown.

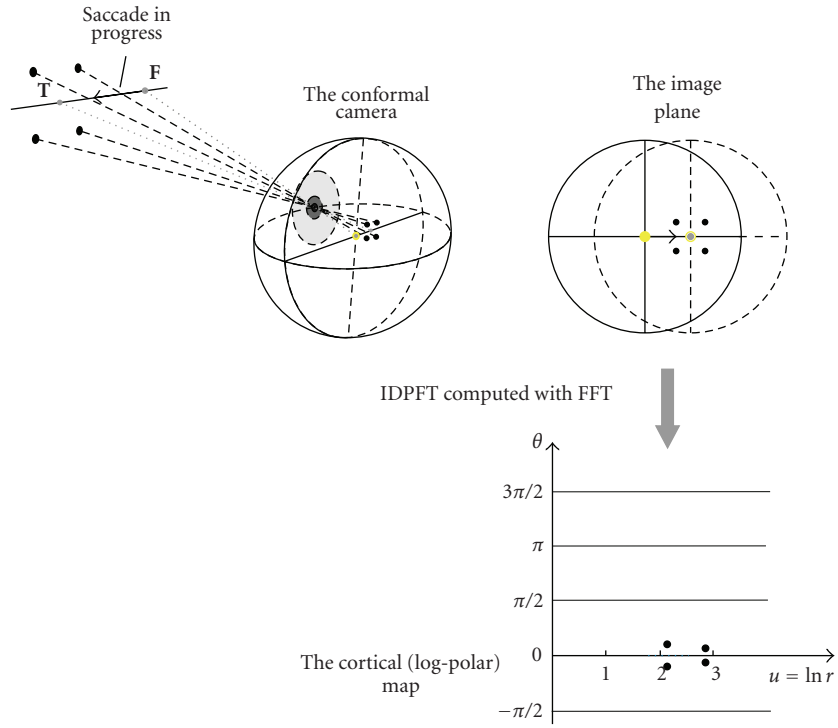


Figure 6: The local retinotopic mapping of four probes flashed around the saccade target at **T**.

Step 2 (see Figure 7). The log-polar image is multiplied by two characteristic set functions, $\chi_{A_i}(u_m, \theta_n)$, $i = 1$ or 2; the domain of each is shown in Figure 7 in a different color, the blue-enclosed region for A_1 and the green-enclosed for A_2. We obtain two images $f_i(u_m, \theta_n) = \chi_{A_i}(u_m, \theta_n) f(u_m, \theta_n)$, $i = 1, 2$ representing cortical half-images into which the image would be divided by the retinal vertical meridian after the fovea landed on the target at **T**. We recall that the characteristic function of a set A is defined by the following condition: $\chi_A(x)$ takes on 1 if $x \in A$ and 0 if $x \notin A$. The blue-enclosed image is reflected both in the vertical ($u = c_1$ line where c_1 is the midpoint of the projection of A_1 into

the u-axis) and in the horizontal ($\theta = 0$ line) axes and translated (blue arrow) while the green-enclosed image is only translated in the u-direction (green arrow).

These transformations are shown on the left of the gray arrow in Figure 7 while the results of these transformations (red dots) are shown on its right. The translation in the (u, θ)-coordinates (blue arrow) is obtained by the shift of the IDPFT

$$f_{m+h,n-j} = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} e^{i2\pi hk/M} e^{-i2\pi jl/N} \hat{f}_{k,l} e^{-(u_k+j\delta)} e^{i2\pi mk/M} e^{i2\pi nl/N}, \tag{22}$$

(here $i = \sqrt{-1}$) which can be computed by FFT. The formula in (22) is the standard shift property of Fourier transform; the cortical image $f_{m,n}$ is translated by $-h$ pixels in the u-coordinate and by j pixels in the θ-direction, as the blue arrow shows in Figure 7. (Equivalently, the coordinate system is translated by h pixels in the u-coordinate and by $-j$ pixels in the θ-direction.) In (22), the inverse discrete Fourier transform is applied to $e^{i2\pi hk/M} e^{-i2\pi jl/N} \hat{f}_{k,l} e^{-(u_k+j\delta)}$ where $\delta = T/M$, $T = \ln R$, is the spacing in the u-coordinate (see (21) in Section 5.2) and $\hat{f}_{k,l} e^{-(u_k+j\delta)}$ is the original Fourier transform with the normalizing area factor in log-polar coordinates.

Further, the image reflection about the vertical axis of the A_1 region can be done with two consecutive transformations, each computable with FFT. These transformations consist of the reflection $f_1(u_m, \theta_n) \rightarrow f_1(-u_m, -\theta_n)$ followed by the translation $f_1(-u_m, -\theta_n) \rightarrow f_1(2c_1 - u_m, -\theta_n)$. We note that the reflection can be obtained by applying Fourier transform twice to the original image, which follows directly from the Fourier transform definition. The red dots represent peripheral receptive fields shifted to the frame centered at the upcoming saccade target.

Step 3 (see Figure 8). This perisaccadic compression is obtained by decoding the cortical image representation to the visual field representation:

$$\begin{aligned} f_{m-h,n+j} &= \left(\frac{2\pi T}{MN}\right) f(u_m + h\delta, \theta_n - j\gamma) \\ &= \left(\frac{2\pi T}{MN}\right) f\left(e^{u_m + h(\delta)} e^{i(\theta_n - j\gamma)}\right) \\ &= \left(\frac{2\pi T}{MN}\right) f\left(e^{h\delta} r_m e^{i(\theta_n - j\gamma)}\right), \end{aligned} \tag{23}$$

where $i = \sqrt{-1}$ and $\delta = T/M$, $\gamma = 2\pi/N$. We see that, under the shift of the coordinate system (u_m, θ_n) by $(h\delta, -j\gamma)$, the original position of a dor at $r_m e^{i\theta_n}$ is transformed to $e^{-h\delta} r_m e^{i(\theta_n - j\gamma)}$, resulting in the compression shown in the scene in Figure 8 with red dots referenced by red arrows to the original positions of flashed probes (black dots). Although this step is not supported by FFT, a commonly used look-up table [65] could efficiently decode the cortical image see the discussion in Section 7.1. Where and how the brain accomplished this step is the greatest mystery of primate perception.

In the modeling steps we presented above, the cortical translation shown by the green arrow in Figure 7 gives only compression of the two corresponding probes in the scene because $j = 0$. However, in the translation shown in Figure 7 by the blue arrow, $j \neq 0$ and the corresponding image undergoes the compression and rotation, both needed to have the fovea at the center of the four red dots in the scene shown in Figure 8, when the saccade lands at **T**. Because of this rotation, we need two reflections to have the original parity of the image.

Although we do not show here quantitative results of the modeling steps, the qualitative results can be seen if we translate the cortical image of the bar in Figure 4(b), say in the u-coordinate to the left by a number of pixels (pixels are shown in the left upper corner), and trace out mentally its retinal copy using the $d \times d$ square lattice in (a) and its log-polar image in (b). We can see that the bar in (a) will be compressed with respect to the origin of (x_3, x_1)-coordinates. We note that, as mentioned in Step 3, we cannot apply FFT to render this compressed retinal image.

The model presented in this section complements the theory proposed in [29]. Experimental results suggest a very tight time course followed by perisaccadic compression with its duration of about 130 ms and with the maximum mislocalization immediately before the saccade. In the model we propose here, this saccadic dynamics can be easily accounted for: the distance of the shift $(-h, j)$ (in terms of cortical pixels) can be taken as a function of time. Another aspect of perisaccadic compression that is accounted for in our modeling is the fact that not all of RFs undergo shift during eyes saccadic motion. In Step 2, the translations are applied to selected (salient) retinotopic areas. These two aspects of perisaccadic perception are not supported in a natural way by the model presented in [29].

6.2. On Modeling Global Retinotopy. The global retinotopy reflects the anatomical fact that the axons in the optic nerve leaving each eye split along the retinal vertical meridian when the axons originating from the nasal half of the retina cross at the optic chiasm to the contralateral brain's hemisphere and join the temporal half, which remains on the eye's side of the brain. This splitting and crossing reorganize the local retinotopy (log-polar mapping) such that the left brain hemisphere receives the right visual field projection and the right brain hemisphere receives the left visual field projection. According to the split theory [61, 70], which provides a greater understanding of vision cognitive processes than the bilateral theory of overlapping projections, there is a sharp foveal split along the vertical meridian of hemispherical cortical projections. Both hemispheric projections are connected by a massive corpus callosal bridge of about 200 M of neuronal fibers [71].

Although crucial for synthesizing 3D representation from the binocular disparities in the pair of 2D retinal images, we cannot fully address the global retinotopy here because

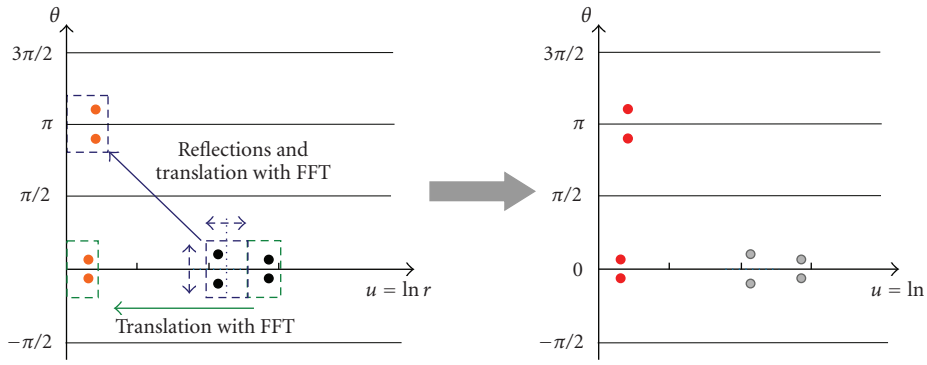


Figure 7: Modeling shifts of RFs of flashed probes. See text for detailed description.

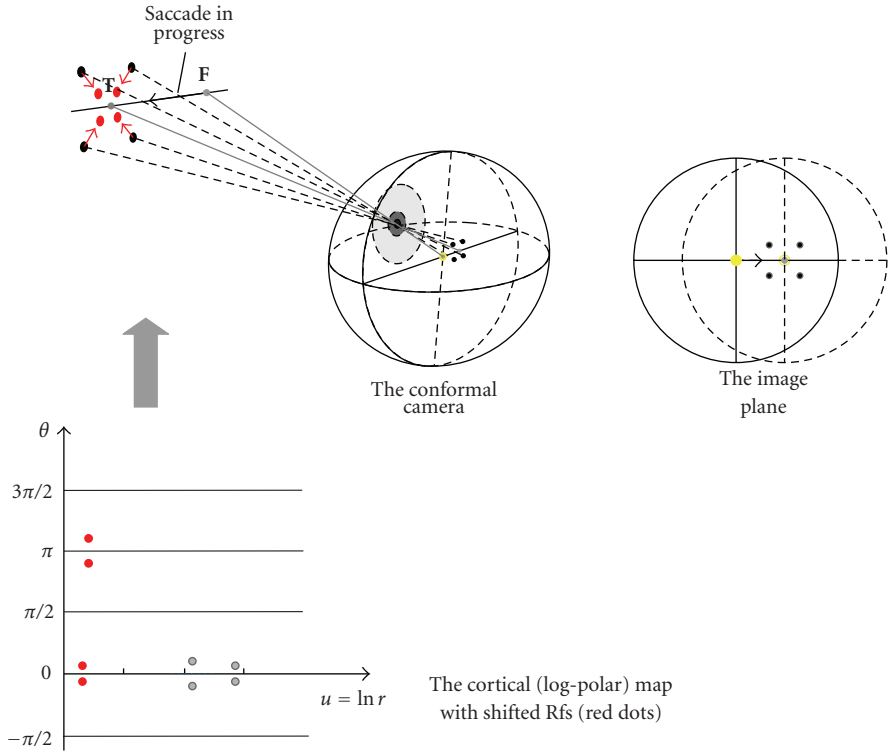


Figure 8: Perisaccadic compression of flashed probes perceived briefly during the saccadic eye movement. Note that this perception does not involve the visual sensory system.

the foveal image representation is not included in out modeling. However, the fovea-less global retinotopy can be easily modeled with DPFT by two reflections (cf., Step 2 in our modeling) both computable with FFT. Figure 9(b) shows the result of the reflection about the line $u = 0$ of the peripheral region given by $\pi/2 \le \theta \le 3\pi/2$. It is followed in 9(c) with the result of the reflection about $\theta = \pi/2$.

At this point, we can only graphically show what we expect to obtain when the foveal image representation complements the peripheral (log-polar) image representation we have developed in terms of projective Fourier transform. To this end, Figure 10 shows the peripheral region (gray) and the central foveal region (yellow). These two regions are connected by the transitional region (shaded with gray lines). The green curve in Figure 10 shows the cortical projection of a straight line making an angle of $\pi/4$ with the retinal horizontal meridian and passing through the center of the fovea.

7. Discussion

Our model, which is based on PFT, uses the approximation of the retinotopy given by the complex logarithmic mapping $z \mapsto \ln z$ with $|z| \ge 1$, where 1 represents, with the appropriate normalization, the radius of the foveal region removed in our modeling. This mapping transforms PFT into the standard Fourier integral that is computable by FFT, providing the benchmark of efficiency not available in any

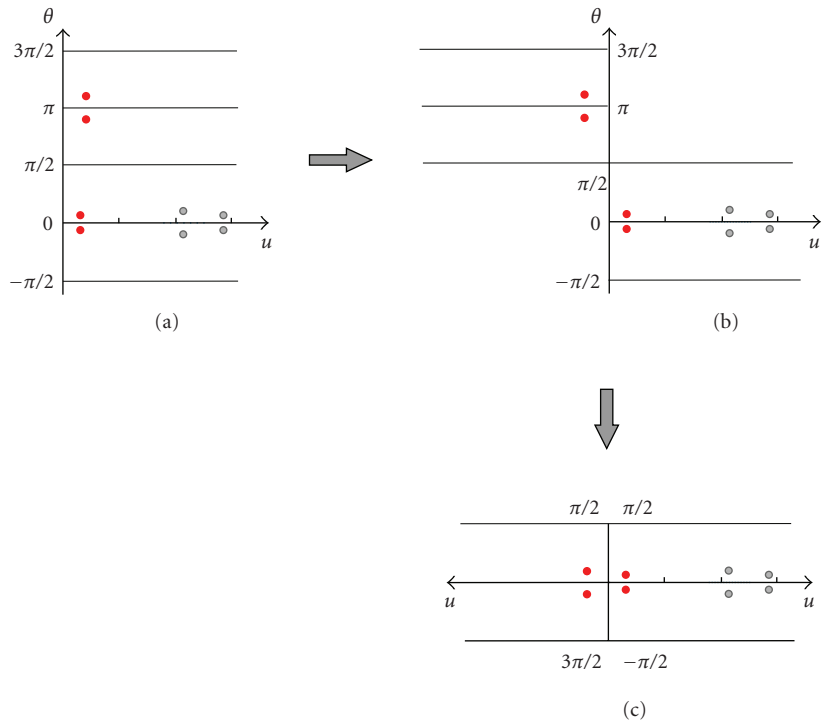


Figure 9: Two consecutive reflections that can be computed with FFT account for the global retinotopy without the foveal region.

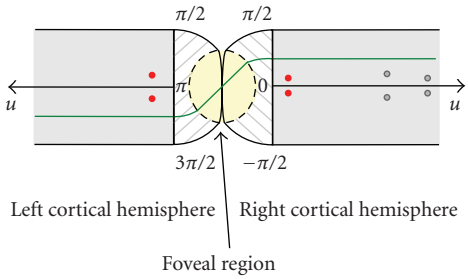


Figure 10: Schematic depiction of the global retinotopy with the peripheral region (gray) and the foveal region (yellow). Shaded in gray lines is the perifoveal region, which connects the other two regions. The red dots show the shifted neuronal activity in the receptive fields from the veridical position (gray dots) resulting in the illusory space compression.

other computational modeling of perisaccadic perception. Although the foveal system is indispensable to primate vision, it is rather less important to the proper functioning of the visuosaccadic system. In fact, neuronal cells in FEF/PEF, the higher cortical areas implicated in the RF shift mechanism that obtain retinotopic projections from the occipital cortex, have larger RFs and primarily code stimuli spatial locations rather than stimuli features [72]. We should note that, although we use the complex logarithmic approximation for all retinotopically organized brain areas, this approximation is well established only for the first visual cortical areas and SC of the midbrain [7, 58]. This is justified if we realize that most algorithmic principles employed by natural visual systems need to be reformulated to better fit modern computational algorithms (FFT in our case).

The results obtained in the simulations in [29], where the Schwartz' retinotopic mapping $z \mapsto \ln(z + a)$ was used to approximate the cortical magnification factor, showed the unidirectional shift component of mislocalization superimposed on perisaccadic compression. However, it was noticed there that this component did not scale with the compression according to the experimental data. The unidirectional shift is absent in the model presented here because, in our approximation of retinotopy, the parameter a is zero. Since the unidirectional perisaccadic shift has a different neural origin (as it is primarily caused by delays in neuronal signals) than perisaccadic compression (caused by remapping), it should not be accounted for by just this parameter.

Further, in both our model and the model in [29], perceptual compression is attributed to a translation of the origin of the logarithmic coordinate system, which results in a linear relation between perceived and actual probes' positions. Thus, the nonlinearity observed in [19, 21] is not accounted for in our modeling. However, asymmetry and nonlinearity present in experimental data could have casual origin resulting from multiple sources; we mention here three: (1) an asymmetric distribution of photoreceptor/ganglion cell density [73], (2) the average preferred fixation area located from the point of the highest cone density a distance of about half of the central fovea's radius (70 μm) [74], and finally, (3) fluctuations of the cortical surface curvature (and therefore the lengths of the geodesics) across hemispheres [75]. Given the incomplete understanding of the neural processes underlying perisaccadic perception,

it is impossible to distinguish between these "accidental" causes and the real neural mechanisms captured in modeling.

7.1. Relations to Other Models and the Current Research. Two computational theories of transsaccadic vision that have been proposed in visual neuroscience are related to our modeling, both with similar functional explanation of perisaccadic mislocalization by the cortical magnification factor. The first theory [29], which motivated our research, was discussed and compared with our model in the previous sections. In summary, our modeling can be seen as complementing the approach proposed in [29] by providing efficient image representation suitable for processing visual information during saccadic eye movement, and in particular, the classically assumed process of active remapping compensating for receptive fields displacements.

The second theory [18, 20] explains perisaccadic compression by spatial attention being directed to the target of a planned saccade. The authors proposed an elaborate computational modeling that assumes that the flashed stimuli RFs in cortical areas dynamically change position toward the saccade target RF as the result of the gain of feedback of the retinotopically organized activity hill of the saccade target in the oculomotor SC layer. This attention directed to the target increases spatial discrimination at the saccade target location before the saccade onset. The perceived spatial distortion of stimuli is the result of the cortical magnification factor of the visuo-cortical mapping (or retinotopy of the visuomotor pathways) when the position of each stimulus is decoded from activity of the neural ensemble. Thus, in this theory, different neural processes to those proposed in [29] are assumed: a local and transient change in the gain control around the saccade target in retinotopic-organized stimulus position represented by a hill of neural activity is inducing perisaccadic distortion of the perceived stimulus location. However, because circuitry underlying receptive field remapping is widespread and not well understood, it cannot be easily decided whether saccadic remapping is the cause or consequence of saccadic compression [29].

What really sets apart our modeling from other models is the fact that the computational efficiency is built into the modeling process, as all algorithmic steps (except the last one) involve computations with FFT. This is especially important because the incessant occurrence of saccades and the time needed for the oculomotor system to plan and execute each saccade require that visual information is efficiently processed during each fixation period without repeating, afresh, the whole process at each fixation [29].

All models proposed so far capture only the initial, front-end stage of remapping for a particularly simple scene of flashed probes and, though they explain the perisaccadic mislocalization phenomenon, they leave out the crucial modeling step of the integration of the objects' features (pattern, color, ...) across saccades [30–33] that achieves stability of perception [14, 76]. Many issues must be understood better before this crucial modeling step is achieved. We give two examples. During a scene viewing, a salient map of the landmarks and behaviorally significant objects of the scene is created and the RF shift updates the retinotopy of only this saliency map [13]. Although it is still unclear what a saliency map should be when viewing complex natural scenes, it points to the possibility of working with a sparse visual information data when performing Step 3 in the model outlined in Section 6.1. Thus, a lookup table approach [65] could be efficient enough for this step even when viewing a complex scene. Further, the time course of different stages in visual information processing in trans-saccadic perception and their influences on other cognitive processes is unclear. It is well known that scene gist recognition, when the scene is viewed for 50 ms [77], is critical in the early stage of scene perception, influencing more complex cognitive processes, such as directing our attention within a scene, facilitating object recognition, and influencing long-term memory. Only very recently [78] has it been found that peripheral vision is more useful for recognizing the gist of a scene than central vision (i.e., foveal + parafoveal vision) is, even though central vision is more efficient per pixel at processing gist.

Although the understanding of neural mechanisms involved in trans-saccadic perception is incomplete, a significant progress in understanding dynamic interaction taking place between different pathways in the visuosaccadic system has been recently made. In particular, the fundamental principles underlying perception of objects across saccades have been outlined [13]. Therefore, we should expect major advances in the near future. As a consequence, in robotic vision research, which is still wedged in-between the limited knowledge about biological visual processing and technological and software restrictions imposed by current cameras, scanners, and computers, it is becoming more important than ever to propose different, even if competing, perspectives on how to model known processes involved in trans-saccadic perception. In this article we proposed a comprehensive, biologically mediated engineering approach to model an active vision system. Our modeling, which is efficiently supporting both hard-wired eccentricity-dependent visual resolution and front-end modeling of mechanisms that may contribute to continuity and stability of trans-saccadic perception, is based on an abstract and less intuitive camera model with underlying nonmetric (conformal) Möbius geometry. However, our initial study of the smooth-pursuit eye movements, which complement fixations and saccades in the scanpath, indicates that the conformal camera with its DPFT-based image representation will also be important in processing visual information during the pursuits.

Further, the conformal camera geometry's effectiveness in the intermediate-level vision problems and the perspectively covariant projective Fourier analysis, well adapted to retinotopy, strongly suggest that DPFT-based image representation should be useful in modeling the neural processes that underlie the transfer of the objects' features across saccades and maintain the continuity and stability of perception.

Finally, it was observed that saccades cause not only a compression of space, but also a compression of time [79]. In order to preserve visual stability during the saccadic

scanpath, receptive fields undergo a fast remapping at the time of saccades. When the speed of this remapping approaches the physical limit of neural information transfer, relativistic-like effects are psychophysiologically observed and may cause space-time compression [80, 81]. Curiously, this suggestion can also be accounted for in our model based on projective Fourier analysis since the group $\mathbf{SL}(2,\mathbb{C})$ of image projective transformations in the conformal camera is the double cover of the group of Lorentz transformations of Einstein's special relativity; for a simple presentation [82, Section 1.2].

8. Summary

In this article, we presented a comprehensive framework we have developed for computational vision over the last decade, and we applied this framework to model some of the processes underlying trans-saccadic perception. We have done this by bringing, in one place, physiological and behavioral aspects of primate visual perception, the conformal camera's computational harmonic analysis, and the underlying conformal geometry. This allowed us to discuss the conformal camera's effectiveness in modeling a biologically mediated active visual system. First, the conformal camera geometry fully accounts for the basic concepts of cocircularity and scale invariance employed by the human vision system in solving the difficult intermediate-level vision problems of grouping local elements into individual objects of natural scenes. Second, the conformal camera has its own harmonic analysis—projective Fourier analysis—providing image representation and processing that is well adapted to image projective transformations and the retinotopic mapping of the brain visual and oculomotor pathways. This later assertion follows from the fact that the projective Fourier transform integrates the head, eyes (conformal cameras), and visual cortex into a single computational system. Based on this system, we presented a computational model for some neural processes of the perisaccadic perception. In particular, we modeled the presaccadic activity, which, through shifts of stimuli current receptive fields to their future postsaccadic locations, is thought to underlie the scene remapping of the current foveal frame to the frame at the upcoming saccade target. This remapping uses the motor command of the impending saccade and may help maintain stability of primate perception in spite of three saccadic eye movements per second with the eyeball's maximum speed of 700 deg/sec producing $200,000$ saccades per day. Our modeling also accounted for perisaccadic mislocalization observed by human subjects in laboratory experiments. Finally, we compared our model with the other computational approaches in the modeling trans-saccadic perception and discussed further developments.

Acknowledgment

The author thanks Dr. Noriyasu Homma for helpful comments.

References

[1] J. Turski, "Projective fourier analysis in computer vision: theory and computer simulations," in *Vision Geometry VI*, A. Y. Wu and L. J. Latecki, Eds., vol. 3168 of *Proceeding of SPIE*, pp. 124–135, 1997.

[2] J. Turski, "Harmonic Analysis on $\mathbf{SL}(2,\mathbb{C})$ and projectively adapted Pattern representation," *Journal of Fourier Analysis and Applications*, vol. 4, no. 1, pp. 67–91, 1998.

[3] J. Turski, "Projective Fourier analysis for patterns," *Pattern Recognition*, vol. 33, no. 12, pp. 2033–2043, 2000.

[4] J. Turski, "Geometric Fourier analysis of the conformal camera for active vision," *SIAM Review*, vol. 46, no. 2, pp. 230–255, 2004.

[5] J. Turski, "Geometric Fourier analysis for computational vision," *Journal of Fourier Analysis and Applications*, vol. 11, no. 1, pp. 1–23, 2005.

[6] J. Turski, "Computational harmonic analysis for human and robotic vision systems," *Neurocomputing*, vol. 69, no. 10–12, pp. 1277–1280, 2006.

[7] E. L. Schwartz, "Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding," *Vision Research*, vol. 20, no. 8, pp. 645–669, 1980.

[8] J. Turski, "Harmonic analysis for cognitive vision: perisaccadic perception," in *Human Vision and Electronic Imaging XIV*, B. E. Rogowitz and T. N. Pappas, Eds., vol. 7240 of *Proceeding of SPIE*, pp. 72401A-1–72401A-5, 2009.

[9] P. H. Schiller, "The neural control of visual guided eye movements," in *Cognitive Neuroscience of Attention: A Developmental Perspective*, J. E. Richards, Ed., pp. 3–50, Erlbnum Associates, Lawrence, Kan, USA, 1998.

[10] R. H. Wurtz, "Neuronal mechanisms of visual stability," *Vision Research*, vol. 48, no. 20, pp. 2070–2089, 2008.

[11] C. L. Colby and M. E. Goldberg, "Space and attention in parietal cortex," *Annual Review of Neuroscience*, vol. 22, pp. 319–349, 1999.

[12] J.-R. Duhamel, C. L. Colby, and M. E. Coldberg, "The updating of the representation of visual space in parietal cortex by intended eye movements," *Science*, vol. 255, no. 5040, pp. 90–92, 1992.

[13] D. Melcher and C. L. Colby, "Trans-saccadic perception," *Trends in Cognitive Sciences*, vol. 12, no. 12, pp. 466–473, 2008.

[14] E. P. Merriam, C. R. Genovese, and C. L. Colby, "Remapping in human visual cortex," *Journal of Neurophysiology*, vol. 97, no. 2, pp. 1738–1755, 2007.

[15] M. A. Sommer and R. H. Wurtz, "A pathway in primate brain for internal monitoring of movements," *Science*, vol. 296, no. 5572, pp. 1480–1482, 2002.

[16] O. J. Grüsser, "On the history of the ideas of efference copy and reafference," *Clio Medica*, vol. 33, pp. 35–55, 1995.

[17] A. Hunt and P. Cavanagh, "Clocking saccadic remapping," *Journal of Vision*, vol. 8, p. 818, 2008.

[18] F. H. Hamker, M. Zirnsak, D. Calow, and M. Lappe, "The peri-saccadic perception of objects and space," *PLOS Computational Biology*, vol. 4, no. 2, pp. 1–15, 2008.

[19] M. Kaiser and M. Lappe, "Perisaccadic mislocalization orthogonal to saccade direction," *Neuron*, vol. 41, no. 2, pp. 293–300, 2004.

[20] M. Lappe, H. Awater, and B. Krekelberg, "Postsaccadic visual references generate presaccadic compression of space," *Nature*, vol. 403, no. 6772, pp. 892–895, 2000.

[21] J. Ross, M. C. Morrone, and D. C. Burr, "Compression of visual space before saccades," *Nature*, vol. 386, no. 6625, pp. 598–601, 1997.

[22] H. Honda, "The time courses of visual mislocalization and of extraretinal eye position signals at the time of vertical saccades," *Vision Research*, vol. 31, no. 11, pp. 1915–1921, 1991.

[23] L. Matin and D. G. Pearce, "Visual perception of direction for stimuli flashed during voluntary saccadic eye movements," *Science*, vol. 148, no. 3676, pp. 1485–1488, 1965.

[24] J. Schlag and M. Schlag-Rey, "Illusory localization of stimuli flashed in the dark before saccades," *Vision Research*, vol. 35, no. 16, pp. 2347–2357, 1995.

[25] H. Awater and M. Lappe, "Mislocalization of perceived saccade target position induced by perisaccadic visual stimulation," *Journal of Neuroscience*, vol. 26, no. 1, pp. 12–20, 2006.

[26] F. Ostendorf, C. Fischer, C. Finke, and C. J. Ploner, "Perisaccadic compression correlates with saccadic peak velocity: differential association of eye movement dynamics with perceptual mislocalization patterns," *Journal of Neuroscience*, vol. 27, no. 28, pp. 7559–7563, 2007.

[27] S. M. C. I. Van Wetter and A. J. Van Opstal, "Experimental test of visuomotor updating models that explain perisaccadic mislocalization," *Journal of Vision*, vol. 8, no. 14, pp. 1–22, 2008.

[28] K. Matsumiya and K. Uchikawa, "The role of presaccadic compression of visual space in spatial remapping across saccadic eye movements," *Vision Research*, vol. 43, no. 18, pp. 1969–1981, 2003.

[29] R. VanRullen, "A simple translation in cortical log-coordinates may account for the pattern of saccadic localization errors," *Biological Cybernetics*, vol. 91, no. 3, pp. 131–137, 2004.

[30] M. Demeyer, P. De Graef, J. Wagemans, and K. Verfaillie, "Transsaccadic identification of highly similar artificial shapes," *Journal of Vision*, vol. 9, no. 4, article 28, pp. 1–14, 2009.

[31] D. Melcher, "Dynamic, object-based remapping of visual features in transsaccadic perception," *Journal of Vision*, vol. 9, no. 14, article 2, pp. 1–17, 2008.

[32] D. Melcher, "Selective attention and the active remapping of object features in trans-saccadic perception," *Vision Research*, vol. 49, no. 10, pp. 1249–1255, 2009.

[33] M. Wittenberg, F. Bremmer, and T. Wachtler, "Perceptual evidence for saccadic updating of color stimuli," *Journal of Vision*, vol. 8, no. 14, pp. 1–9, 2008.

[34] S. Anstis, "Picturing peripheral acuity," *Perception*, vol. 27, no. 7, pp. 817–825, 1998.

[35] D. C. Burr, M. C. Morrone, and J. Ross, "Selective suppression of the magnocellular visual pathway during saccadic eye movements," *Nature*, vol. 371, no. 6497, pp. 511–513, 1994.

[36] Y. Yarbus, *Eye Movements and Vision*, Plenum Press, New York, NY, USA, 1967.

[37] B. Girard and A. Berthoz, "From brainstem to cortex: computational models of saccade generation circuitry," *Progress in Neurobiology*, vol. 77, no. 4, pp. 215–251, 2005.

[38] J. Najemnik and W. S. Geisler, "Optimal eye movement strategies in visual search," *Nature*, vol. 434, no. 7031, pp. 387–391, 2005.

[39] P. W. Glimcher, "Making choices: the neurophysiology of visual-saccadic decision making," *Trends in Neurosciences*, vol. 24, no. 11, pp. 654–659, 2001.

[40] M. Berger, *Geometry I*, Springer, New York, NY, USA, 1987.

[41] G. Jones and D. Singerman, *Complex Functions*, Cambridge University Press, Cambridge, UK, 1987.

[42] M. Henle, *Modern Geometries. The Analytical Approach*, Prentice Hall, Upper Saddle River, NJ, USA, 1997.

[43] D. A. Brannan, M. F. Esplen, and J. J. Gray, *Geometry*, Cambridge University Press, Cambridge, UK, 1998.

[44] S. Ullman, *Higher-Level Vision: Object Recognition and Visual Cognition*, MIT Press, Cambridge, Mass, USA, 1996.

[45] K. Koffka, *Principles of Gestalt Psychology*, Harcourt & Brace, New York, NY, USA, 1935.

[46] M. Wertheimer, *Laws of Organization in Perceptual Forms, Harcourt*, Brace & Jovanovitch, London, UK, 1938.

[47] D. J. Field, A. Hayes, and R. F. Hess, "Contour integration by the human visual system: evidence for a local 'association field'," *Vision Research*, vol. 33, no. 2, pp. 173–193, 1993.

[48] P. Parent and S. W. Zucker, "Trace inference, curvature consistency, and curve detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 8, pp. 823–839, 1989.

[49] C. C. Chow, D. Z. Jin, and A. Treves, "Is the world full of circles?" *Journal of Vision*, vol. 2, no. 8, pp. 571–576, 2002.

[50] M. Sigman, G. A. Cecchi, C. D. Gilbert, and M. O. Magnasco, "On a common circle: natural scenes and gestalt rules," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 4, pp. 1935–1940, 2001.

[51] H. Blum, "Biological shape and visual science," *Journal of Theoretical Biology*, vol. 38, no. 2, pp. 205–287, 1973.

[52] M. Leyton, "A theory of information structure I. General principles," *Journal of Mathematical Psychology*, vol. 30, no. 2, pp. 103–160, 1986.

[53] M. Leyton, "A theory of information structure II: a theory of perceptual organization," *Journal of Mathematical Psychology*, vol. 30, no. 3, pp. 257–305, 1986.

[54] F. Attneave, "Some informational aspects of visual perception," *Psychological Review*, vol. 61, no. 3, pp. 183–193, 1954.

[55] D. D. Hoffman and W. A. Richards, "Parts of recognition," *Cognition*, vol. 18, no. 1–3, pp. 65–96, 1984.

[56] I. Kovacs and B. Julesz, "Perceptual sensitivity maps within globally defined visual shapes," *Nature*, vol. 370, no. 6491, pp. 644–646, 1994.

[57] A. W. Knapp, *Representation Theory of Semisimple Groups: An Overview Based on Examples*, Princeton University Press, Princeton, NJ, USA, 1986.

[58] N. Tabareau, D. Bennequin, A. Berthoz, J.-J. Slotine, and B. Girard, "Geometry of the superior colliculus mapping and efficient oculomotor computation," *Biological Cybernetics*, vol. 97, no. 4, pp. 279–292, 2007.

[59] Y. Petrov, M. Carandini, and S. McKee, "Two distinct mechanisms of suppression in human vision," *Journal of Neuroscience*, vol. 25, no. 38, pp. 8704–8707, 2005.

[60] J. Xing and D. J. Heeger, "Center-surround interactions in foveal and peripheral vision," *Vision Research*, vol. 40, no. 22, pp. 3065–3072, 2000.

[61] M. Lavidor and V. Walsh, "The nature of foveal representation," *Nature Reviews Neuroscience*, vol. 5, no. 9, pp. 729–735, 2004.

[62] C. F. R. Weiman, "Log-polar Binocular Vision System," Transition Research Corporation: NASA Phase II SBIR Final Report, 1994.

[63] F. Berton, G. Sandini, and G. Metta, "Anthropomorphic visual sensors, in encyclopedia of sensors," C. A. Grimes, E. C. Dickey, and M. V. Pishko, Eds., vol. 10, pp. 1–16, American Scientific, New York, NY, USA, 2006.

[64] G. Bonmassar and E. L. Schwartz, "Space-variant fourier analysis: the exponential chirp transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 10, pp. 1080–1089, 1997.

[65] A. Bernardino, J. Santos-Victor, and G. Sandini, "Foveated active tracking with redundant 2D motion parameters," *Robotics and Autonomous Systems*, vol. 39, no. 3-4, pp. 205–221, 2002.

[66] G. Baratoff, C. Toepfer, and H. Neumann, "Combined space-variant maps for optical-flow-based navigation," *Biological Cybernetics*, vol. 83, no. 3, pp. 199–209, 2000.

[67] N. Tamayo and V. J. Traver, "Entropy-based saliency computation in log-polar images," in *Proceedings of the 3rd International Conference on Computer Vision Theory and Applications, (VISAPP '08)*, vol. 1, pp. 501–506, 2008.

[68] R. Manzotti, A. Gasteratos, G. Metta, and G. Sandini, "Disparity estimation on log-polar images and vergence control," *Computer Vision and Image Understanding*, vol. 83, no. 2, pp. 97–117, 2001.

[69] M. A. Silver and S. Kastner, "Topographic maps in human frontal and parietal cortex," *Trends in Cognitive Sciences*, vol. 13, no. 11, pp. 488–495, 2009.

[70] C. D. Martin, G. Thierry, J.-F. Démonet, M. Roberts, and T. Nazir, "ERP evidence for the split fovea theory," *Brain Research*, vol. 1185, no. 1, pp. 212–220, 2007.

[71] E. Zaidel and M. Iacobini, *The Parallel Brain: The Cognitive Neuroscience of the Corpus Callosum, A Bradford Book*, MIT Press, Cambridge, Mass, USA, 2003.

[72] A. P. Saygin and M. I. Sereno, "Retinotopy and attention in human occipital, temporal, parietal, and frontal cortex," *Cerebral Cortex*, vol. 18, no. 9, pp. 2158–2168, 2008.

[73] V. H. Perry and A. Cowey, "The ganglion cell and cone distributions in the monkey's retina: implications for central magnification factors," *Vision Research*, vol. 25, no. 12, pp. 1795–1810, 1985.

[74] N. M. Putnam, H. J. Hofer, N. Doble, L. Chen, J. Carroll, and D. R. Williams, "The locus of fixation and the foveal cone mosaic," *Journal of Vision*, vol. 5, no. 7, pp. 632–639, 2005.

[75] B. Fischl, M. I. Sereno, and A. M. Dale, "Cortical surface-based analysis. II: inflation, flattening, and a surface-based coordinate system," *NeuroImage*, vol. 9, no. 2, pp. 195–207, 1999.

[76] J. Gottlieb, "From a different point of view: extrastriate cortex integrates information across saccades. Focus on "Remapping in Human Visual Cortex"," *Journal of Neurophysiology*, vol. 97, no. 2, pp. 961–962, 2007.

[77] L. Fei-Fei, A. Iyer, C. Koch, and P. Perona, "What do we perceive in a glance of a real-world scene?" *Journal of Vision*, vol. 7, no. 1, article 10, pp. 1–29, 2007.

[78] A. M. Larson and L. C. Loschky, "The contributions of central versus peripheral vision to scene gist recognition," *Journal of Vision*, vol. 9, no. 10, article 6, pp. 1–16, 2009.

[79] M. C. Morrone, J. Ross, and D. Burr, "Saccadic eye movements cause compression of time as well as space," *Nature Neuroscience*, vol. 8, no. 7, pp. 950–954, 2005.

[80] D. Burr and C. Morrone, "Time perception: space-time in the brain," *Current Biology*, vol. 16, no. 5, pp. R171–R173, 2006.

[81] M. C. Morrone, J. Ross, and D. Burr, "Keeping vision stable: rapid updating of spatiotopic receptive fields may cause relativistic-like effect," in *Problems of Space and Time in Perception*, R. Nijhawan, Ed., Cambridge, UK, 2008.

[82] S. Sternberg, *Group Theory and Physics*, Cambridge University Press, Cambridge, UK, 1994.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 860790, 9 pages
doi:10.1155/2010/860790

Research Article

Haptic Perception with Self-Organizing ANNs and an Anthropomorphic Robot Hand

Magnus Johnsson and Christian Balkenius

Lund University Cognitive Science, Kungshuset, Lundagård, 222 22 LUND, Sweden

Correspondence should be addressed to Magnus Johnsson, magnus.johnsson@lucs.lu.se

Received 18 August 2009; Accepted 30 December 2009

Academic Editor: Noriyasu Homma

We have implemented and compared four biologically motivated self-organizing haptic systems based on proprioception. All systems employ a 12-d.o.f. anthropomorphic robot hand, the LUCS Haptic Hand 3. The four systems differ in the kind of self-organizing neural network used for clustering. For the mapping of the explored objects, one system uses a Self-Organizing Map (SOM), one uses a Growing Cell Structure (GCS), one uses a Growing Cell Structure with Deletion of Neurons (GCS-DN), and one uses a Growing Grid (GG). The systems were trained and tested with 10 different objects of different sizes from two different shape categories. The generalization abilities of the systems were tested with 6 new objects. The systems showed good performance with the objects from the training set as well as in the generalization experiments. Thus the systems could discriminate individual objects, and they clustered the activities into small cylinders, large cylinders, small blocks, and large blocks. Moreover, the self-organizing ANNs were also organized according to size. The GCS-DN system also evolved disconnected networks representing the different clusters in the input space (small cylinders, large cylinders, small blocks, large blocks), and the generalization samples activated neurons in a proper subnetwork in all but one case.

1. Introduction

Haptic perception, that is, active tactile perception, is of outmost importance in the field of robotics since a well-performing robot must be able to interact with objects in its environments. However, haptic perception is also important in supporting and sometimes also in substituting the visual modality during the recognition of objects. Like humans, robots should be able to perceive shape and size as well as to discriminate between individual objects by haptic exploration.

The modelling of haptic perception as well as the implementation of haptic perception in robots have been neglected areas of research. Robot hand research has mainly focused on grasping and object manipulation [1–4], and many models of hand control have focused on the motor aspect rather than on haptic perception [5, 6], although there are some exceptions [7–17].

Previously we have designed and implemented haptic size perception systems [18–21], haptic shape perception systems [22–25], and haptic texture/hardness perception systems [26, 27].

The haptic size perception systems used a simple three-fingered robot hand, the LUCS Haptic Hand I, with the thumb as the only movable part. The LUCS Haptic Hand I was equipped with 9 piezo-electric tactile sensors. This system used Self-Organizing Maps, SOMs, [28] and a neural network with leaky integrators and it successfully learned to categorize a test set of spheres and cubes according to size.

The haptic shape perception systems used a three-fingered 8 d.o.f robot hand, the LUCS Haptic Hand II, equipped with a wrist for horizontal rotation and a mechanism for vertical repositioning. This robot hand was equipped with 45 piezo-electric tactile sensors. This system used active explorations of the objects by several grasps with the robot hand to gather tactile information. The LUCS Haptic Hand II was not equipped with any proprioceptive sensors, that is, sensors that register joint angles, but the system used the positioning commands to the actuators as a substitute. Depending on the version of the system, either

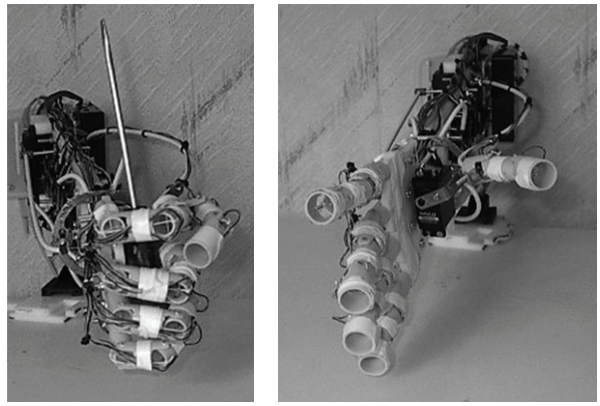
tensor product (outer product) operations or a novel neural network, the Tensor Multiple Peak SOM, T-MPSOM [23–25], was used to code the tactile information in a useful way while a SOM was used for the categorization. The system successfully learned to discriminate between different shapes as well as between different objects within a shape category when tested with a set of spheres, blocks, or cylinders.

The haptic texture/hardness perception systems employed a microphone-based texture sensor and a hardness sensor that measures the displacement of a stick pressed at the object with a constant force. With these sensors, we implemented systems that automatically evolved monomodal as well as bimodal representations of texture and hardness [26], and also a system that evolved monomodal representations of texture and hardness while at the same time learning to associate these representations. The latter was done by using a variant of the SOM called Associative Self-Organizing Map (A-SOM) [27].

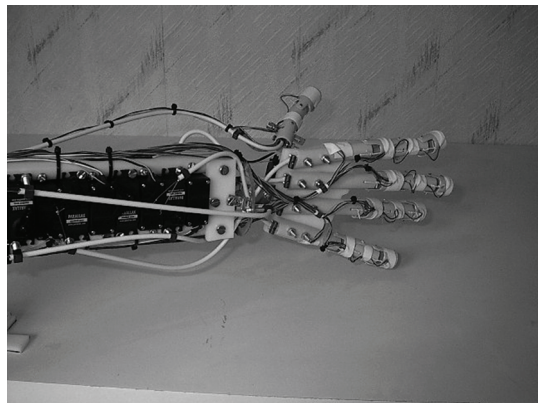
This paper explores a somewhat different approach to shape and size perception which is based solely on proprioception. Using the position of each joint as the only input, we have designed an anthropomorphic robot hand and self-organizing systems that can discriminate objects and categorize them according to shape and size [29–31].

When designing a neural network based on self-organizing perception system a natural question comes up, namely, what kinds of neural network architectures are most suitable to use. A common choice is the self-organizing map (SOM) that we have used in previous work. This is often a very good choice but it suffers from some limitations, for example, the topological structure is fixed and the number of neurons in the neural network has to be preset by the system designer. Other limitations are that parameters like the learning rate, the initial neighbourhood size, and the decreasing rate of the neighbourhood size also have to be set manually by the designer.

To address these limitations we have, in addition to a SOM-based system, explored and compared three other haptic systems based on the same robotic hand. These systems are based on alternative neural network architectures that avoid some or all of the limitations with a SOM-based system. The four systems differ in one respect, namely, in the kind of self-organizing neural network employed to cluster the input. The first system uses the SOM, the second uses the Growing Cell Structures (GCS), the third uses the Growing Cell Structures with Deletion of Neurons GCS-DN [32, 33], and the fourth uses the Growing Grid (GG) [34].



(a)



(b)

Figure 1: The LUCS Haptic Hand III while holding a screw driver, in open position seen in a front view and in a side view. Some of the actuators in the forearm can also be seen in the side view. The 12-d.o.f robot hand has five fingers, is of the same size as a human hand, and all its parts have approximately the same proportions as their counterparts in a human hand. Each finger can be separately flexed/extended in the proximal joint, whereas the medial and distal joints are flexed/extended together as real human fingers. As a human hand, the thumb has only a proximal and a distal phalang. These can also be separately flexed/extended. In addition the thumb can also be adducted/abducted in a way similar to the human thumb. The wrist is capable of flexion/extension. The actuators of the LUCS Haptic Hand III are controlled via an SSC-32 (Lynxmotion Inc.). The proprioceptive sensors are scanned with a MAX396CPI multiplexor chip and digitalized using an NiDaq 6008 (National Instruments). The NiDaq 6008 converts multiple analog input signals to digital signals, which are conveyed to the computer via a USB-port. The robot hand is equipped with two multiplexor chips, which means it is prepared for 21 additional sensors.

2. LUCS Haptic Hand III

The LUCS Haptic Hand III is a five-fingered 12-d.o.f anthropomorphic robot hand equipped with 11 proprioceptive sensors (Figure 1). The robot hand has a thumb consisting of two phalanges, whereas the other fingers have three phalanges. The thumb can be separately flexed/extended in both the proximal and the distal joints and adducted/abducted. The other fingers can be separately flexed/extended in their proximal joints, whereas the middle and the distal joints are flexed/extended together. All this is similar to the human hand. The wrist can also be flexed/extended as the wrist of a human hand. The phalanges are made of plastic pipe segments and the force transmission from the actuators, which are located in the forearm, are handled by tendons inside the phalanges in a similar way to the tendons of a human hand. All fingers, except the thumb, are mounted

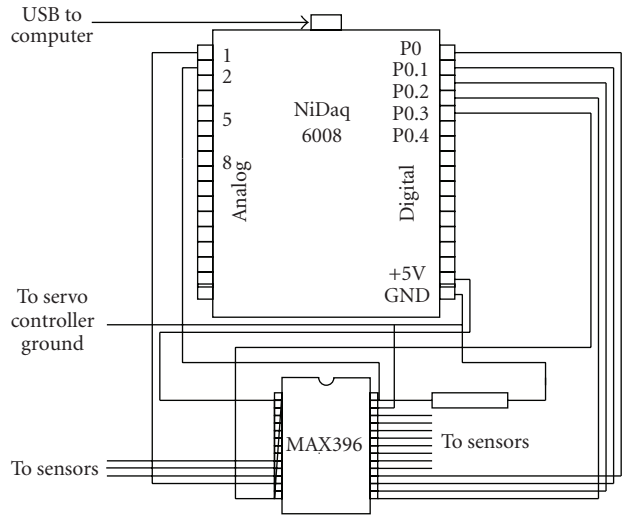


Figure 2: The circuits involved in the proprioceptive part of the LUCS Haptic Hand III. The NiDaq 6008 converts multiple analog input signals to digital signals that are conveyed to the computer via an, USB-port. The MAX396 chip is a multiplexor circuit for selection of sensor channels.

directly on the palm. The thumb is mounted on an RC servo, which enables the adduction/abduction. The RC servo is mounted on the proximal part of the palm, similar to the site of the thumb muscles in a human hand. The actuators of the fingers and the wrist are located in the forearm. This is also similar to the muscles that actuate the fingers of a human hand. The hand is actuated by in total 12 RC servos, and to get proprioceptive sensors, the internal potentiometers in the RC servos, except the RC servo that actuates the wrist, have been included in the sensory circuit (Figure 2). The resistances of these potentiometers are proportional to the angle of the different joints.

The software for the LUCS haptic hand III is developed in C++ and Java, and much of it runs within the Ikaros system [35, 36]. Ikaros provides an infrastructure for computer simulations of the brain and for robot control.

3. Self-Organizing ANNs

3.1. Self-Organizing Map. The SOM consists of an $I \times J$ grid of neurons with a fixed number of neurons and a fixed topology. Each neuron n_{ij} is associated with a weight vector $w_{ij} \in R^n$. During adaptation, the weight vectors for the neurons are adjusted to a degree which is determined by a neighbourhood function $N_{ijc}(t)$ with a size that decreases with time. The adaptation strength $\alpha(t)$ also decreases with time. The SOM variant used in our experiments is a dot product SOM with Gaussian neighbourhood. The adaptation algorithm works as follows.

> At time t, each neuron n_{ij} receives an input vector $x(t) \in R^n$.
>
> The neuron c associated with the weight vector $w_c(t)$ most similar to the input vector $x(t)$ is selected,

$$c = \arg\max_c\{\|x(t)w_c(t)\|\}. \tag{1}$$

The weight vectors w_{ij} of the neurons n_{ij} are adapted according to:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t)N_{ijc}(t)\Big[x(t) - w_{ij}(t)\Big], \tag{2}$$

where $0 \le \alpha(t) \le 1$ is the adaptation strength with $\alpha(t) \to 0$ when $t \to \infty$ and the neighbourhood function $N_{ijc}(t)$ is a Gaussian function the width of which decreases with time.

3.2. Growing Cell Structures. The GCS has a variable number of neurons and a k-dimensional topology, where k can be arbitrarily chosen. The adaptation of a weight vector in the GCS is done in a similar way as in the SOM, but the adaptation strength is constant over time and only the best matching unit and its direct topological neighbours are adapted. The GCS estimates the probability density function $p(x)$ of the input space by the aid of local signal counters that keep track of the relative frequencies of input signals gathered by each neuron. These estimates are used to indicate proper locations to insert new neurons. The insertion of new neurons by this method will result in a smoothing out of the relative frequencies between different neurons. The advantages of this approach are that the topology of the network will self-organize to fit the input space, the proper number of neurons for the network will be automatically determined and the learning rate and neighbourhood size parameters are constant over time. The basic building block and also the initial configuration of the GCS are a k-dimensional simplex. Such a simplex is for $k = 2$ a triangle. The variant of the GCS algorithm used in our experiments works as follows.

> The network is initialized to contain $k + 1$ neurons with weight vectors $w_i \in R^n$ randomly chosen. The neurons are connected so that a k-dimensional simplex is formed.

At time step t, an input vector $x(t) \in R^n$ activates a winner neuron c for which the following is valid

$$c = \arg\min_c\{\|x(t) - w_c(t)\|\}, \tag{3}$$

where $\|\cdot\|$ is the Euclidean distance, and the squared distance between the input vector and the weight vector of the winner neuron c is added to a local error variable E_c:

$$\Delta E_c = \|x(t) - w_c(t)\|^2. \tag{4}$$

The weight vectors are updated by fractions ε_b and ε_n, respectively, according to:

$$\begin{aligned} \Delta w_c(t) &= \varepsilon_b(x(t) - w_c(t)), \\ \forall i \in N_c : \Delta w_i(t) &= \varepsilon_n(x(t) - w_i(t)), \end{aligned} \tag{5}$$

where N_c is the set of direct topological neighbours of c.

A neuron is inserted if the number of input vectors that have been generated so far is an integer multiple of a parameter λ. This is done by finding the neuron q with the largest accumulated error and the neuron f among its direct topological neighbours which has the weight vector with the longest distance from the weight vector of the neuron q, insert the new neuron r in between, remove the earlier connection (q, f), and connect r with q and f and with all direct topological neighbours that are common for q and f. The weight vector for r is interpolated from the weight vectors for q and f:

$$w_r = \frac{w_q + w_f}{2}. \tag{6}$$

The local error counters for all neighbours to r are decreased by a fraction α that depends on the number of neighbours of r:

$$\forall i \in N_r : \Delta E_i = \left(-\frac{\alpha}{|N_r|}\right) \cdot E_i. \tag{7}$$

The error variable for r is set to the average of its neighbours:

$$E_r = \left(\frac{1}{|N_r|}\right) \cdot \sum_{\iota \in Nr} E_i, \tag{8}$$

and then the error variables of all neurons are decreased:

$$\forall i : \Delta E_i = -\beta E_i. \tag{9}$$

In GCS-DN, a neuron (or several if that is necessary to keep a consistent topological structure of k-dimensional simplices) is deleted, provided that the network has reached its maximum size; at the same occasions new neurons are inserted. Thereafter, new neurons are inserted again according to the algorithm described above until the network has reached its maximum size again. This process is repeated a preset number of times, in our experiments 250 times.

3.3. Growing Grid. The GG can be seen as an incremental variant of the SOM. It consists of an $I \times J$ grid of neurons with a fixed topology but with I and J increasing with time as new rows and columns are inserted. In addition to a weight vector $w_{ij} \in R^n$, each neuron n_{ij} also has a local counter variable T to estimate where to insert new rows or columns of neurons in the grid. The self-organizing process of a GG is divided into two phases: a growth phase and a fine-tuning phase. During the growth phase, the grid grows by insertion of new rows and columns until the wanted size of the network has been achieved. During the fine-tuning phase, the network size does not change and a decreasing adaptation strength $\alpha(t)$ is used. The size of the neighbourhood is not decreasing with time. Instead the network is growing with a constant neighbourhood size and therefore the fraction of all neurons that are adapted decreases over time. The variant of the GG algorithm used in our experiments is described below.

Growth Phase. Initialize the network to contain 2×2 neurons with weight vectors randomly chosen.

At time t, an input vector $x(t) \in R^n$ is generated and received by each neuron n_{ij} in the grid.

The neuron c associated with the weight vector $w_c(t)$ most similar to the input vector $x(t)$ is selected:

$$c = \arg\max_c \{\|x(t) w_c(t)\|\}. \tag{10}$$

Increment the local counter variable T_c for c:

$$T_c = T_c + 1. \tag{11}$$

The weight vectors w_{ij} of the neurons n_{ij} are adapted according to:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha N_{ijc}\Big[x(t) - w_{ij}(t)\Big], \tag{12}$$

where $0 \leq \alpha \leq 1$ is the adaptation strength and the neighbourhood function N_{ijc} is a Gaussian function. Notice that α and N_{ijc} are not functions of t though.

A new row or column is inserted if the number of input vectors that have been generated so far is an integer multiple λ of the current number of neurons in the grid. This is done by finding the neuron q with the largest value of the local counter variable T and the neuron f among its direct topological neighbours which has the weight vector with the longest distance from the weight vector of the neuron q. Depending on the relative positions of q and f, a new row or a new column is inserted.

If q and f are in the same row, then a new column is inserted between the columns of q and f. The weight vectors for the new neurons are interpolated from their direct neighbours in the same row.

If q and f are in the same column, then a new row is inserted between the rows of q and f. The weight vectors for the new neurons are interpolated from their direct neighbours in the same column.

Adjust I or J to reflect the new numbers of rows and columns in the grid. Reset all local counter values:

$$T_{n_{ij}} = 0. \tag{13}$$

If the desired network size has not been reached, then go to step 2, that is, generate a new input vector.

Fine-Tuning Phase. This phase is similar to the growth phase but the adaptation strength $\alpha(t)$ is now decreasing with time and no insertions of new rows or columns are done. This phase stops after a preset number of iterations.

4. Proprioception-Based Systems

All the four systems (Figure 3) consist of the LUCS Haptic Hand III, sensory and motor drivers, a commander module that executes the grasping movements, and a Self-Organizing ANN (SO-ANN). The kind of SO-ANN employed is the only thing that distinguishes one system from another. The sensory driver scans the proprioceptive sensors when requested to do so by the commander module, while the

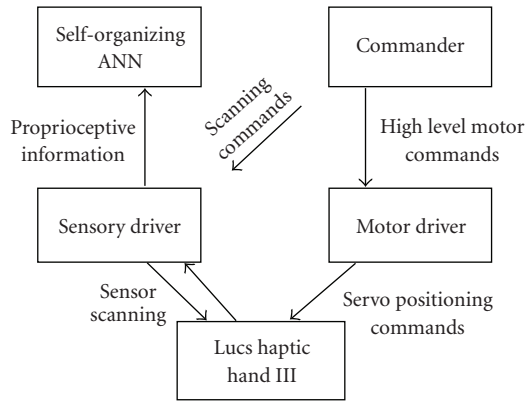


Figure 3: Schematic depiction of the systems. The commander module executes the grasps by sending high-level motor commands to the motor driver, which translates and conveys the information to the servo controller board of the robot hand. When the robot hand has become fully closed around the object, the commander module requests a scanning of the 11 proprioceptive sensors of the robot hand. The sensory information is conveyed as a vector to a Self-Organizing ANN (SO-ANN). The SO-ANN is a Self-Organizing-Map, a Growing Cell Structures, a Growing Cell Structures with Deletion of Neurons, or a Growing Grid.

motor driver translates high-level motor commands from the commander module to positioning commands for the robot hands servo controller board. When the commander executes a grasp, and the robot hand is fully closed around the object, the sensory driver scans the 11 proprioceptive sensors and outputs an eleven-element vector to the SO-ANN, which is adapted.

The SOM-based system uses a 225 neurons dot product SOM with plane topology, which uses softmax activation with the softmax exponent equal to 10 [37]. It is trained by 2000 iterations.

The GCS-based system grows, by inserting a new neuron every 19th iteration, until a size of 225 neurons has been reached.

The GCS-DN based system grows until a size of 225 neurons has been reached, also by inserting a new neuron every 19th iteration, then the deletion/insertion process described in Section 3.2 is repeated 250 times. Finally this yields a number of disconnected networks with altogether 225 neurons.

The GG-based system grows by inserting a new row or column each time the number of time steps t since the previous insertion equals a multiple λ of the current grid size, that is, until $t = \lambda IJ$ with $\lambda = 19$. The growth phase lasts until a minimum grid size of 225 neurons has been reached, then the model runs in fine tuning mode for 1000 iterations.

We have trained the systems with 10 objects (see Table 1 objects a–j). These objects are either cylinder shaped or block shaped. There are five objects of each shape category. All objects are sufficiently high to be of a nonvariable shape in those parts grasped by the robot hand, for example, a bottle is grasped on the part of equal diameter below the bottle neck.

During the grasping tests, the test objects were placed on a table with the open robot hand around them. If the objects were block shaped, we always placed the widest side against the palmar side of the robot hand.

To simplify the testing procedure, each object was grasped 5 times by the robot hand, that is, in total 50 grasps were carried out, and the sensory information was written to a file. Then the SO-ANN were trained and tested with this set of 50 samples. The training phase for the SOM system lasted for 2000 iterations. The GCS system was trained until a network size of 225 neurons was reached. The GCS-DN system was trained until a network size of 225 neurons was reached and then the insertion/deletion process described in Section 3.2 was repeated 250 times. The GG system was trained with a growth phase which lasted until the minimal network size reached 225 neurons, and then for 1000 iterations in fine tuning mode.

Each fully trained system was tested with the original training set and in addition with three new block-shaped and three new cylinder-shaped objects of variable sizes (see Table 1, objects 1–6) as described in the next section.

5. Generalization Tests

We have also tested if the systems were able to generalize their knowledge to new objects, that is, to objects not included in the training set. To this end we used 6 new objects, Table 1. 1–6, 3 cylinder shaped objects and 3 block shaped objects. The new objects were of variable sizes. The fully trained systems were fed by input from grasps of the new objects under the same conditions as the objects in the training set. Each object in the new set was grasped once and the activity in the SO-ANN for each system was recorded.

6. Results

The results are depicted in Figure 4. Figure 4(a) shows the centres of activation in the SOM in the fully trained SOM-based system when tested with the training set and the test set. The SOM seems to be organized according to shape. Four groups of objects can be distinguished in the map, large block shapes, small block shapes, large cylindrical shapes, and small cylindrical shapes. The SOM also seems to be organized in a clockwise manner according to size. The result of the generalization experiment shows that all test objects are mapped so that they are ordered according to size in the same way as the objects in the training set, and that they are also correctly mapped according to shape. The activations in the SOM also indicate that it is possible to discriminate individual object of the training set to a large extent and this is also true for the test objects, since each of the test objects is also mapped so that it can be identified as the most similar object of the training set. The results with the SOM-based system are thoroughly described in [29].

Figure 4(b) shows the centres of activation in the GCS in the fully trained GCS-based system. Only the part of the GCS which is activated by some object is shown in the figure. This system produces similar results as the SOM-based system,

Table 1: The 16 objects used in the experiments with the four systems. The objects a–j were used both for training and testing, whereas the objects 1–6 were used in the generalization tests.

Label	Object	Shape	Size (mm)	Size (mm)
a	Tube	Cylinder	Diameter = 58	—
b	Beer can	Cylinder	Diameter = 64	—
c	Wood block	Block	Length = 75	Width = 47
d	Wine bottle	Cylinder	Diameter = 70	—
e	Plastic block 1	Block	Length = 63	Width = 63
f	Plastic bottle 2	Cylinder	Diameter = 72	—
g	Olive oil bottle	Block	Length = 65	Width = 65
h	Plastic bottle 1	Cylinder	Diameter = 80	—
i	Plastic block 2	Block	Length = 80	Width = 63
j	Coffee package	Block	Length = 97	Width = 67
1	Card board package 1	Block	Length = 77	Width = 66
2	Card board package 2	Block	Length = 84	Width = 62
3	Card board package 3	Block	Length = 95	Width = 62
4	Spice bottle	Cylinder	Diameter = 57	—
5	Treacle bottle	Cylinder	Diameter = 63	—
6	Plastic bottle 3	Cylinder	Diameter = 79	—

that is, the organization of the GCS separates large block shapes, small block shapes, large cylinder shapes, and small cylinder shapes. The GCS is also organized according to size with the smallest objects represented uppermost in the GCS and the largest in the lowermost part. The ability for discrimination of individual objects is approximately similar as that for the SOM-based system. Also this system activates neurons at proper locations when fed with the objects of the generalization test set.

Figure 4(c) shows the final network structure of the fully trained GCS-DN based system. As can be seen, this network structure consists of several disconnected subnetworks. This is due to the removal of neurons that represent parts of the input space with a low value of the probability density function. As a result, such a network tends to self-organize into subnetworks that represent different clusters in the input space. This is also what happened in our experiments. As indicated in the figure, one or more subnetworks can be seen as representing one of the categories large block shapes, small block shapes, large cylinder shapes, and small cylinder shapes. The objects of the generalization test set activate neurons in the proper subnetworks except in one case, namely, the test object 1 is a large block but is identified as a large cylinder.

Figure 4(d) shows the centres of activation in the GG in the fully trained GG-based system. This system produces similar results as the SOM-based system and the GCS-based system, that is, the organization of the GG separates large block shapes, small block shapes, large cylinder shapes, and small cylinder shapes. As indicated in the figure, the GG is also organized according to size. The ability for discrimination of individual objects is approximately similar as that for the SOM-based system. All 6 objects of the generalization test set are mapped so that they can be associated with the correct shape category and identified with the most similar object of the training set.

7. Discussion

We have experimented with four self-organizing systems for clustering of proprioceptive data collected by our anthropomorphic robot hand, the LUCS Haptic Hand III. All four systems were able to cluster the sensory information according to shape, and all four of them resulted in networks which preserve the size ordering of the training objects. The systems could also discriminate individual objects, more or less. The systems have proven to have an excellent generalization capacity. This is clearly illustrated in the categorization of the 6 new objects that offered different characteristics of shape and size.

The results are interesting because they reveal that the proprioceptive information encompasses information about both the shape and the size of the grasped objects, and in addition information that enables discrimination of the individual objects to some extent.

In comparison with our earlier systems for haptic shape perception [22–25], the current systems have turned out to be much more able to correctly categorize objects according to shape in a much wider size range, and this is done with a less computationally expensive model. The current systems were also able to map the sizes of the objects in an ordered fashion, and to discriminate between objects as long as they were not too similar. A human would probably have a similar problem if she, like our systems, was not able to detect the material properties of the objects or expressed differently, if all object were of exactly the same material and weight.

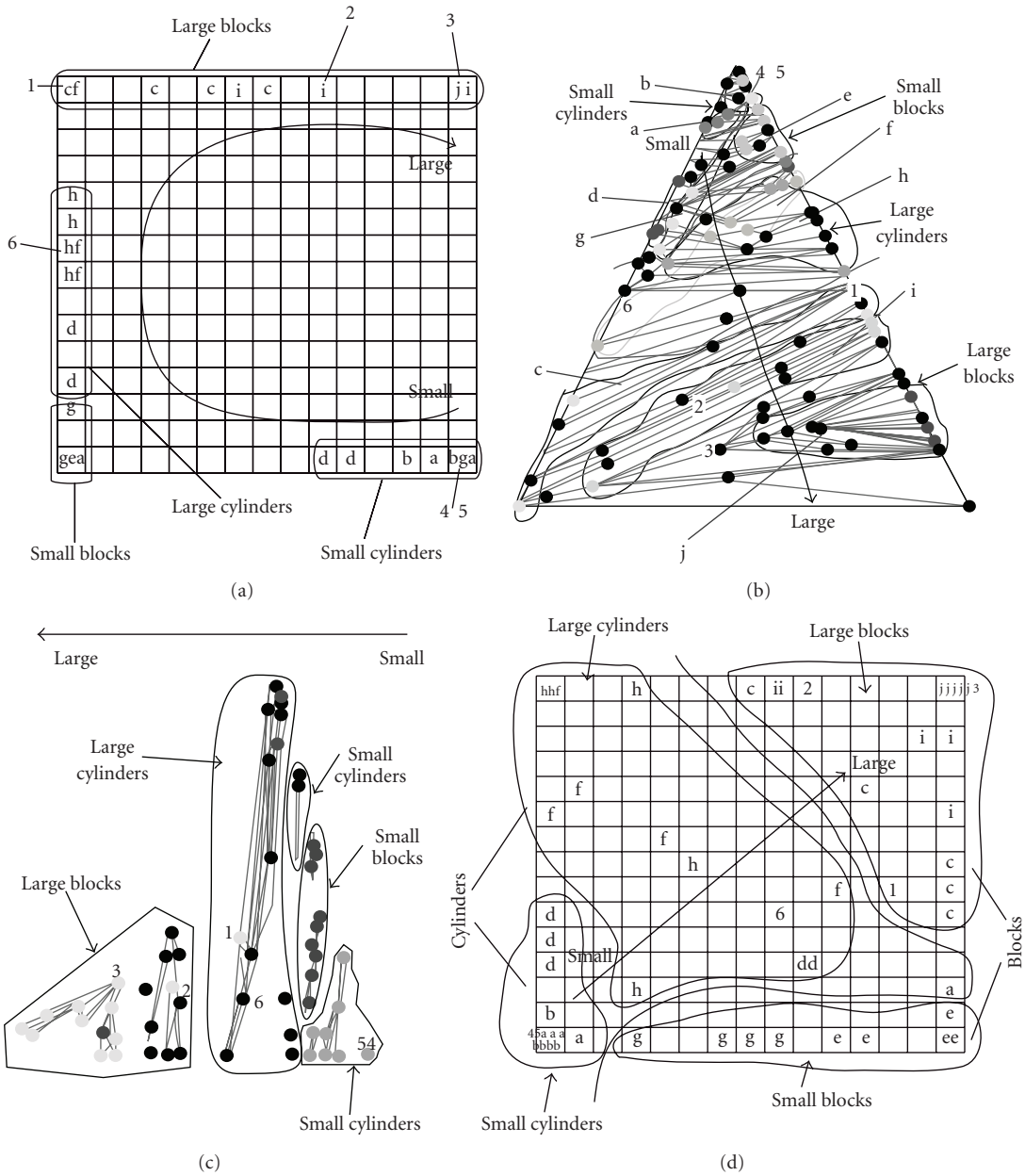


Figure 4: The test results of the four systems. (a) The SOM-based system is organized according to shape and size. Groups of large blocks, small blocks, large cylinders, and small cylinders can be distinguished. The activations tend to be located according to size of the objects in a clockwise manner. Individual objects can be discriminated to a large extent. (b) The GCS-based system produces similar results as the SOM-based system, that is, it is organized according to shape and size. The GCS-based system separates large blocks, small blocks, large cylinders, and small cylinders, and the objects are represented according to size with the smallest objects uppermost and the largest lowermost in the GCS. Also individual objects can be discriminated to a large extent. (c) The GCS-DN based system self-organized into subnetworks, where one or more subnetworks represent the categories large blocks, small blocks, large cylinders, and small cylinders. (d) The GG-based system separates large blocks, small blocks, large cylinders, and small cylinders, and the grid is organized according to size. Individual objects can be discriminated to a large extent. The 6 test objects (indicated with the numbers 1–6) not included in the training set activated neurons at proper locations perfectly in all systems but the GCS-DN based system. In that system, object 1 triggered activation in the wrong subnetwork. (see Table 1 for the meaning of the labels).

The SOM-based, the GCS-based, and the GG-based systems performed at approximately a similar level. This could be an argument for using the alternative neural network architectures GCS and GG instead of the SOM, because that reduces the number of parameters that have to be set. According to Fritzke [38], the performance of the GCS is actually slightly better than the performance of the SOM in complex and realistic problems. The results of our experiments in [39] also point in that direction.

The GCS and the GCS-DN also have the virtue to get organized into networks whose topology reflect the probability density function of the input space. The GCS-DN is especially interesting since it has the property to automatically form disconnected subnetworks that represent clusters in the input space. It should be possible to implement an online version of the GCS-DN algorithm that never stops and that should result in a set of networks, that reflects the probability density function of the input space, which changes if the probability density function happens to be nonstationary. In other words, if the probability density function of the input space changed, then the set of subnetworks would change by the deletion of some subnetworks and the split, followed by growth of others.

It should be mentioned that the graphical presentation of GCS and GCS-DN could be improved. Fritzke [32] suggests a method on how to embed these kinds of networks in the plane for better visualizations. In this method, a physical model is maintained where the neurons are considered as discs influenced by attractive and repulsive forces.

The success with the GCS-based, the GCS-DN based, and the GG-based systems suggests an increased focus on our part on these kinds of self-organizing neural networks. The advantage of getting rid of several parameter settings like network size, learning rate, and neighbourhood settings can be important to succeed with more complex cognitive models with several coupled neural networks at multiple levels. To be forced to set all the parameters in a good way for all included neural networks with complex dependencies in such a model could prove to be overwhelming.

It would be interesting to compare our systems to self-organizing systems developed by others. Heidemann and Schöpfer [11] describe a haptic system, which consists of a plate with a touch sensitive array mounted on a robot arm. The system explores an object by sequences of contacts and feeds a self-organizing neural architecture with input. The system was able to learn to recognize 7 different objects when tested.

Natale and Torres-Jara [14] describe a system consisting of an upper body humanoid robot with a hand equipped with dome-like tactile sensors, which are sensitive to pressure from all directions, as well as position sensors (proprioception). The system also includes a camera together with a visual system for coarse localization of the object. The information gathered by the system was used as input to a SOM. When evaluated with 4 different objects, a bottle, a box, and two cups, these objects were mapped differently. However, the cups could not be distinguished from each other.

When compared with the two systems described above our current systems stand out in that they are able to categorize the objects according to shape, order them according to size, as well as recognize individual objects to a large extent.

In the future, we plan to increase the use of neural networks like GCS and GG as an alternative to the SOM in our haptic systems. By doing so, we will reduce the number of parameters that have to be set explicitly and this should yield more robust systems.

Because of the successful approach with using proprioceptive information as a base for haptic shape perception as well as size perception, we will in the nearest future continue our research in haptic perception with the following task: try to bring the proprioceptive systems to their absolute limits, for example, by exploiting the possibility of the LUCS Haptic Hand III to carry out a more active exploration than simply grasping the objects in only one way. This can be done by adducting/abducting the thumb and by flexing/extending the wrist differently in different grasps

At a later stage, we will study the interaction between haptics and vision. This would be interesting because these modalities interact to a considerable extent [40].

Acknowledgment

The authors want to acknowledge the financial support from Stiftelsen Landshövding Per Westlings Minnesfond to the LUCS Haptic Hand III.

References

[1] P. Dario, C. Laschi, A. Menciassi, E. Guglielmelli, M. C. Carrozza, and S. Micera, "Interfacing neural and artificial systems: from neuroengineering to neurorobotics," in *Proceedings of the 1st International IEEE EMBS Conference on Neural Engineering*, pp. 418–421, 2003.

[2] K. J. DeLaurentis and C. Mavroidis, "Development of a shape memory alloy actuated robotic hand," 2000, http://citeseer.ist.psu.edu/383951.html.

[3] C. Rhee, W. Chung, M. Kim, Y. Shim, and H. Lee, "Door opening control using the multi-fingered robotic hand for the indoor service robot," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, pp. 4011–4016, New Orleans, La, USA, 2004.

[4] H. Sugiuchi, Y. Hasegawa, S. Watanabe, and M. Nomoto, "A control system for multi-fingered robotic hand with distributed touch sensor," in *Proceedings of the 26th Annual Conference of the IEEE Industrial Electronics Society (IECON '00)*, vol. 1, pp. 434–439, Nagoya, Japan, October 2000.

[5] M. A. Arbib, A. Billard, M. Iacoboni, and E. Oztop, "Synthetic brain imaging: grasping, mirror neurons and imitation," *Neural Networks*, vol. 13, no. 8-9, pp. 975–997, 2000.

[6] A. H. Fagg and M. A. Arbib, "Modeling parietal-premotor interactions in primate control of grasping," *Neural Networks*, vol. 11, no. 7-8, pp. 1277–1303, 1998.

[7] P. K. Allen and P. Michelman, "Acquisition and interpretation of 3-D sensor data from touch," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 397–404, 1990.

[8] S. Caselli, C. Magnanini, and F. Zanichelli, "Haptic object recognition with a dextrous hand based on volumetric shape representations," in *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '94)*, pp. 280–287, Las Vegas, Nev, USA, 1994.

[9] P. Dario, C. Laschi, M. C. Carrozza, et al., "An integrated approach for the design and development of a grasping and manipulation system in humanoid robotics," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 1–7, Takamatsu, Japan, October 2000.

[10] I. Erkmen, A. M. Erkmen, A. E. Tekkaya, and T. Pasinlioglu, "Haptic perception of shape and hollowness of deformable

objects using the anthrobot-III robot hand," *Journal of Robotic Systems*, vol. 16, no. 1, pp. 9–24, 1999.

[11] G. Heidemann and M. Schöpfer, "Dynamic tactile sensing for object identification," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '04)*, vol. 1, pp. 813–818, New Orleans, La, USA, 2004.

[12] K. Hosoda, Y. Tada, and M. Asada, "Anthropomorphic robotic soft fingertip with randomly distributed receptors," *Robotics and Autonomous Systems*, vol. 54, no. 2, pp. 104–109, 2006.

[13] J. Jockusch, J. Walter, and H. Ritter, "A tactile sensor system for a three-fingered robot manipulator," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3080–3086, Albuquerque, NM, USA, 1997.

[14] L. Natale and E. Torres-Jara, "A sensitive approach to grasping," in *Proceedings of the 6th International Workshop on Epigenetic Robotics*, pp. 87–94, 2006.

[15] E. M. Petriu, S. K. S. Yeung, S. R. Das, A.-M. Cretu, and H. J. W. Spoelder, "Robotic tactile recognition of pseudorandom encoded objects," *IEEE Transactions on Instrumentation and Measurement*, vol. 53, no. 5, pp. 1425–1432, 2004.

[16] S. A. Stansfield, "A haptic system for a multifingered hand," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 658–664, Sacramento, Calif, USA, April 1991.

[17] D. Taddeucci, C. Laschi, R. Lazzarini, R. Magni, P. Dario, and A. Starita, "An approach to integrated tactile perception," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '97)*, vol. 4, pp. 3100–3105, Albuquerque, NM, USA, April 1997.

[18] M. Johnsson, "LUCS haptic hand I," Tech. Rep. 8, LUCS Minor, Lund University Cognitive Studies, 2004.

[19] M. Johnsson, R. Pallbo, and C. Balkenius, "Experiments with haptic perception in a robotic hand," in *Advances in Artificial Intelligence in Sweden*, pp. 81–86, Mälardalen University, 2005.

[20] M. Johnsson, R. Pallbo, and C. Balkenius, "A haptic system for the LUCS haptic hand I," in *Proceedings of the 1st International Work-Conference on the Interplay between Natural and Artificial Computation (IWINAC '05)*, vol. 3561 of *Lecture Notes in Computer Science*, pp. 386–395, Springer, 2005.

[21] M. Johnsson and C. Balkenius, "Experiments with artificial haptic perception in a robotic hand," *Journal of Intelligent and Fuzzy Systems*, vol. 17, no. 4, pp. 377–385, 2006.

[22] M. Johnsson and C. Balkenius, "LUCS haptic hand II," Tech. Rep. 9, LUCS Minor, Lund University Cognitive Studies, 2006.

[23] M. Johnsson and C. Balkenius, "Haptic perception with a robotic hand," in *Proceedings of the 9th Scandinavian Conference on Artificial Intelligence (SCAI '06)*, Espoo, Finland, 2006.

[24] M. Johnsson and C. Balkenius, "A robot hand with T-MPSOM neural networks in a model of the human haptic system," in *Proceedings of Towards Autonomous Robotic Systems (TAROS '06)*, pp. 80–87, Surrey University, Guildford, UK, 2006.

[25] M. Johnsson and C. Balkenius, "Neural network models of haptic shape perception," *Robotics and Autonomous Systems*, vol. 55, no. 9, pp. 720–727, 2007.

[26] M. Johnsson and C. Balkenius, "Recognizing texture and hardness by touch," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08)*, pp. 482–487, Nice, France, 2008.

[27] M. Johnsson and C. Balkenius, "Associating SOM representations of haptic submodalities," in *Proceedings of Towards Autonomous Robotic Systems (TAROS '08)*, pp. 124–129, University of Edinburgh, Edinburgh, UK, 2008.

[28] T. Kohonen, *Self-Organization and Associative Memory*, Springer, Berlin, Germany, 1988.

[29] M. Johnsson and C. Balkenius, "Experiments with proprioception in a self-organizing system for haptic perception," in *Proceedings of Towards Autonomous Robotic Systems (TAROS '07)*, pp. 239–245, University of Wales, Aberystwyth, UK, 2007.

[30] M. Johnsson and C. Balkenius, "LUCS haptic hand III—an anthropomorphic robot hand with proprioception," Tech. Rep. 13, LUCS Minor, 2007.

[31] M. Johnsson, D. Gil Mendez, and C. Balkenius, "Touch perception with SOM, growing cell structures and growing grids," in *Proceedings of Towards Autonomous Robotic Systems (TAROS '08)*, pp. 79–85, University of Edinburgh, Edinburgh, UK, 2008.

[32] B. Fritzke, "Growing cell structures-a self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, no. 9, pp. 1441–1460, 1994.

[33] B. Fritzke, "Unsupervised ontogenic networks," in *Handbook of Neural Computation*, E. Fiesler and R. Beale, Eds., IOP Publishing and Oxford University Press, London, UK, 1997.

[34] B. Fritzke, "Growing Grid—a self-organizing network with constant neighborhood range and adaptation strength," *Neural Processing Letters*, vol. 2, no. 5, pp. 9–13, 1995.

[35] C. Balkenius and J. Morén, "From isolated components to cognitive systems," *ERCIM News*, p. 16, April 2003.

[36] C. Balkenius, J. Morén, and B. Johansson, *Building System-Level Cognitive Models with Ikaros*, Lund University Cognitive Studies, Lund, Sweden, 2007.

[37] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, New York, NY, USA, 1995.

[38] B. Fritzke, "Kohonen feature maps and growing cell structures—a performance comparision," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS '92)*, pp. 123–130, Denver, Colo, USA, 1992.

[39] D. Gil Mendez, M. Johnsson, A. Soriano Paya, and D. Ruiz Fernandez, "Artificial neural networks for diagnoses of dysfunctions in urology," in *Proceedings of the 1st International Conference on Health Informatics (HEALTHINF '08)*, vol. 2, pp. 191–196, Madeira, Portugal, 2008.

[40] U. Castiello, "The neuroscience of grasping," *Nature Reviews Neuroscience*, vol. 6, no. 9, pp. 726–736, 2005.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 542360, 14 pages
doi:10.1155/2010/542360

Research Article

Human Perception Test of Discontinuous Force and a Trial of Skill Transfer Using a Five-Fingered Haptic Interface

Takahiro Endo, Tomohiro Kanno, Mana Kobayashi, and Haruhisa Kawasaki

Department of Human and Information Systems, Gifu University, 1-1 Yanagido, Gifu 501-1193, Japan

Correspondence should be addressed to Takahiro Endo, tendo@gifu-u.ac.jp

Received 30 October 2009; Accepted 11 March 2010

Academic Editor: Noriyasu Homma

In the transferring of expert skills, it takes a great deal of time and effort for beginners to obtain new skills, and it is difficult to teach the skills by using only words. For those reasons, a skill transfer system that uses virtual reality (VR) and a haptic interface technique is very attractive. In this study, we investigated the human perception of fingertip force with respect to the following changes: (1) the spatial change of the presented force, and (2) the change of the time to present the force. Based on the results of the perception experiments, we considered the skill transfer to a person's five fingers by using a five-fingered haptic interface robot.

1. Introduction

In the medical fields, expert skills such as surgical techniques, palpation techniques, and the like are obtained by long-term training, and the skill is normally acquired by the experience of working with actual patients. However, it is difficult for residents and medical students to train directly with actual patients because of a decrease in volunteers willing to cooperate in the training and the risk of medical error. To transfer expert skills from a trainer (senior doctor) to a trainee (medical student), the trainer has to teach the trainee (1) how to move the hands, and (2) how to exert the exact amount of force with the fingertips. It is difficult to teach the accurate data of position and force by using only words. Because of these challenges, a skill transfer system that uses virtual reality (VR) and a haptic interface has been researched aggressively (e.g., see [1–9] and the references in the survey papers [10–12]), and the results of studies indicate that such a system could contribute to the skill of performing real surgery [7] and to learning of real motor skills [8, 9].

A haptic interface allows a user to communicate with a virtual environment, and the user feels realistic force and tactile sensations when touching virtual objects in a virtual environment. Benefits of a skill transfer system that uses VR and a haptic interface include the following: (1) the movement of the trainer's hand and the operation of the trainer's force can be recorded, so that accurate information can be transmitted to the trainee using a screen and the haptic interface, (2) training according to the trainee's skill level can be selected, and the effect of the training can be presented to the user, and (3) training can occur at a remote site via a network terminal, and several trainees can receive training at the same time.

In most skill transfer systems, a single-point haptic interface, which makes single-point contact between the user and a virtual environment, is used. Thus, the presented force is limited to one point, and the skill transfer for multipoint contacts, which is needed for tasks such as palpation and the like, is not targeted. Another limitation is that only movement in the horizontal plane or the vertical plane is considered, and the skill transfer that requires movement in three-dimensional space is not considered. Multipoint interaction allows a user to perform natural actions such as grasping, manipulation, and exploration of virtual objects, and it will dramatically increase the believability of the haptic experience [13–15]. In performing activities in our daily lives, we usually use multiple fingers; so it is important to exert force at multiple fingertips to make a sensation highly realistic. Multipoint interaction has been achieved in some cases by combining two haptic devices in parallel, which confines the user to a small workspace [5], or by having haptic interfaces that allow the user to exert force at multiple

fingertips of a human hand, but the presented force is only a one-directional force [6]. For example, when the haptic interface generates force by using a wire, the force presented to the operator is only exerted in the direction that the wire pulls. In actual situations, there are huge numbers of tasks that need multiple fingers of contact (with three-directional force). Thus a skill transfer system for multipoint contacts is necessary and important.

To design and develop a skill transfer system in which the operation of the trainer's force is presented to the trainee, it is necessary to investigate and clarify the human perception of the force presented by the haptic interface. In particular, it is important to consider the transfer method based on the human perception. The force that the human being exerts using the hands can be expressed by the direction vector and the magnitude. Many studies of the human perception of the force magnitude have been reported (see [16–22] and the references therein). Although there have been only a few studies about the perception of the force direction, the subject has been researched aggressively in recent years [23–26]. To date, however, there has been no study that evaluated the perception of the spatial fingertip force, that is, the fingertip force in three-dimensional space, and the perception of the fingertip force concerning the time variation. The perception ability is likely to alter based on the spatial change of the presented force and the change of the time used to present the force. If we consider the skill transfer based on human perception, we must evaluate these perception abilities while considering spatial variation and time variation. Although we have examined the time needed for a human being to distinguish a force direction [27], there is no published data regarding the effects on human perception of spatial variation and time variation. In this study, we investigated the perception of the fingertip force concerning the space variation and the presentation time variation, and then we improved the skill transfer method for the multifinger use [27] based on the results of our investigation.

The paper is organized as follows. In the next section, the multifingered haptic interface used here and our previous research are summarized. In Section 3, the human perception of the fingertip force is examined, and the skill transfer to a person's five fingers is developed in Section 4. Then, we consider the simple skill transfer system by using the transfer method and the five-fingered haptic interface, and we describe experimental tests that were carried out to demonstrate the validity of the method. Section 5 contains our conclusions.

2. Five-Fingered Haptic Interface

2.1. Interface Development. The authors have developed multifingered haptic interface robots that are placed opposite a human hand, including HIRO [28], HIRO II$^+$ [29], and HIRO III [30], which is shown in Figure 1. HIRO III can present three-dimensional forces at a human operator's five fingertips. The specifications of HIRO III are shown in Table 1. HIRO III can be briefly summarized as follows.

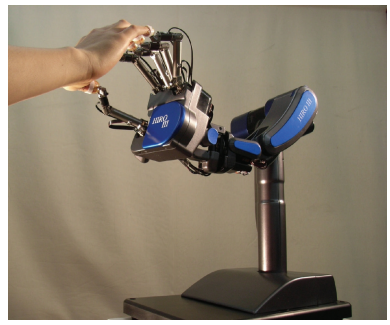
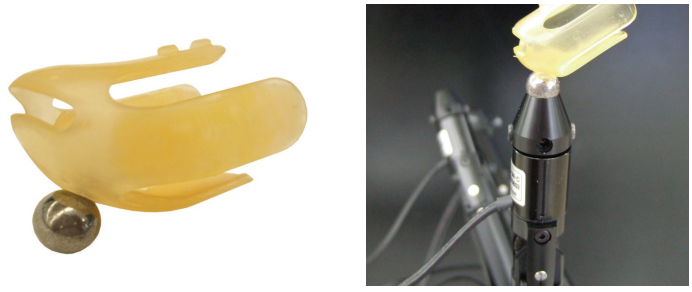


Figure 1: Five-fingered haptic interface robot: HIRO III. An operator connects his/her five fingertips to HIRO III through passive spherical permanent magnet joints.

HIRO III consists of an interface arm and a five-fingered haptic hand. The interface arm is a PUMA-type robot arm consisting of an upper arm (humerus), a lower arm (forearm), and a wrist. The interface arm has 3 degrees of freedom (DOF) at the arm joint and 3 DOF at the wrist joint. The interface arm, therefore, has 6 joints allowing 6 DOF. On the other hand, the haptic hand is constructed of five haptic fingers. Each haptic finger has 3 joints, allowing 3 DOF. The total DOF of HIRO III is 21, and its work space covers VR manipulation on the space of a desktop. Furthermore, a 3-axis force sensor is installed at the top of each finger. To manipulate HIRO III, the operator has to wear a finger holder on his/her fingertips. Figure 2 shows the finger holder and its connection to the haptic finger of HIRO III. The finger holder has a sphere which, when attached to the permanent magnet at the force sensor tip, forms a passive spherical joint. Its role is to adjust for differences between the human and haptic finger orientations, which means that it is safe to use and involves no oppressive feeling for the user. HIRO III allows object manipulation in VR with high realistic sensation. For more details, please see [30].

2.2. Our Previous Skill Transfer System Using Hiro II$^+$. In the skill transfer system considered in this paper, the trainer's work is recorded, and it is reproduced in the VR space. The trainee is then trained to imitate the trainer's work. The trainee's goal is to make his/her fingertip positions and forces track the trainer's positions and forces, respectively. We have proposed cues to assist so that the trainee may efficiently acquire the position information and the force information [31]. Figure 3 shows the visual cues. In this figure, the grasping of a ball is considered as the task of the trainer. In the VR space, a yellow ball is the grasped object. The fingertip positions of the trainer and trainee are shown as small circles in VR space, and the fingertip forces are shown as a tetrahedron (see the right side of Figure 3), which is called a force gauge. The height of the tetrahedron expresses the magnitude of the force, and its direction expresses the direction of the force. That is, we tried to display the position information, force magnitude information, and force direction information on the VR display. However, it turned out that it was extremely difficult for the trainee to



(a) Finger holder

(b) Connection of finger holder to HIRO III

Figure 2: Finger holder and its connection. The operator wears the finger holder at his/her fingertips and connects to HIRO III as shown in (b).

Table 1: Specifications of HIRO III.

Hand	Number of fingers	5
	Degrees of freedom	15 (DOF)
	Weight	0.78 (kg)
Finger	Degrees of freedom	3 (DOF)
	Weight	0.12 (kg)
	Maximum output force	over 3.6 (N)
	Workspace	705 (cm^3) (Thumb)
		587 (cm^3) (Other)
Arm	Degrees of freedom	6 (DOF)
	Weight	3.0 (kg)
	Maximum output force	over 56 (N)
	Workspace	0.09 (m^3)

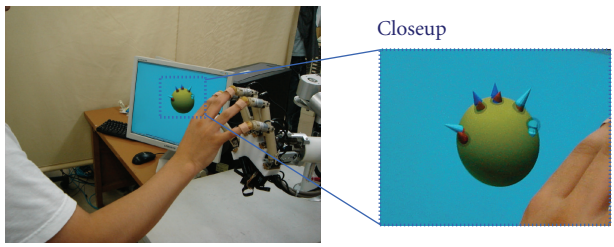


Figure 3: Visual cues.

see the trainer's five-finger position information and force information at the same time, and to control his/her finger positions and forces so that his/her finger positions and forces matched the trainer's finger positions and forces.

In our earlier research, we tried to transfer the trainer's force information and position information to the trainee by using only visual information. However, we obtained feedback that it was difficult for the trainee to learn the trainer's force information and position information on the five fingertips at the same time. So, in this paper, we attempted to transmit the trainer's force information to the trainee by using visuohaptic information, namely, by using not only visual cues but also haptic cues.

3. Measurement of the Perception of Fingertip Force

It is useful to measure human perception ability when we consider the transmission of force. In particular, it is important to know how human beings accurately perceive fingertip force by the haptic interface. The results of investigating the human perception of fingertip force form the foundation of skill transfer. First, we measured the human perception ability with regard to the direction of force by using HIRO III. Here note that all three experiments in this section can be done with a single-point haptic interface. However, in Section 4, we describe how we developed the skill transfer system by using HIRO III and VR technique. To develop the skill transfer system based on the measurement results of human perception, we needed to use HIRO III in the experiments.

3.1. Measurement of the Human Perception of Force Direction. We examined the perception of the force direction with regard to spatial variation and time variation.

3.1.1. Experimental Setup. Ten people in their twenties (nine males and one female) participated in this measurement. All of the participants were right-handed. The participants connected their index finger to the HIRO III at a bar, as

shown in Figure 4(a), and HIRO III presented the force to the participants. Note that we used the hand part of HIRO III in this experiment, and we did not use the arm of HIRO III. In the experiment, the hand part of HIRO III was fixed to the board as shown in Figure 4(a). During the measurement, a cloth covered both HIRO III and the participant's hands; so sight information was not available, as shown in Figure 4(c). The participants responded to the direction of the presented force by using the measuring instrument shown in Figure 4(b). As a measuring instrument, we used a goniometer. One axis of the goniometer was fixed, and the participants responded by using the other axis of the goniometer to indicate the direction of the presented force.

To measure the human perception of the force direction with regard to spatial variation, we considered the following two types of measurement.

(M1) We consider the measurement of the human perception of force in the horizontal direction (x–y plane in Figure 4(d)). In this case, we set the force $\mathbf{F} = [F_x, F_y, F_z]^T$ (N) to show on human index finger as follows:

$$F_x = \|\mathbf{F}\| \cos\theta_h,$$
$$F_y = \|\mathbf{F}\| \sin\theta_h \text{ (N)}, \quad 0 \le \theta_h \le \pi \text{ (rad)}, \tag{1}$$
$$F_z = 0,$$

where θ_h is the angle in the horizontal plane, as shown in Figure 5(a), and $\|\mathbf{F}\|$denotes the magnitude of the force.

(M2) We consider the measurement of the human perception of force in the vertical direction (x–z plane in Figure 4(d)). In this case, we set the force $\mathbf{F} = [F_x, F_y, F_z]^T$ (N) to show on human index finger as follows:

$$F_x = \|\mathbf{F}\| \cos\theta_v,$$
$$F_y = 0 \text{ (N)}, \qquad 0 \le \theta_v \le \pi \text{ (rad)}, \tag{2}$$
$$F_z = \|\mathbf{F}\| \sin\theta_v,$$

where $\theta_v$is the angle in the vertical plane, as shown in Figure 5(b).

In measurements (M1) and (M2), the angles θ_h and θ_v are divided every $\pi/12$ radians, and 13 kinds of forces are presented to the participants in random order. We set $\|\mathbf{F}\|$=1.5 (N). In each measurement, we considered the following three conditions for the time required to present the force, to consider the perception of the force direction concerning the time variation (see Figure 6).

(a) The force **F** was presented until the participant answered.

(b) The cycle in which the force **F** was presented for 0.5 (s) and the force was not presented for 0.5 (s) was repeated until the participant answered. That is, we set $t_1 = 0.5$ (s), $t_2 = 1.0$ (s) in Figure 6.

Table 2: Measurement of the human perception of force direction.

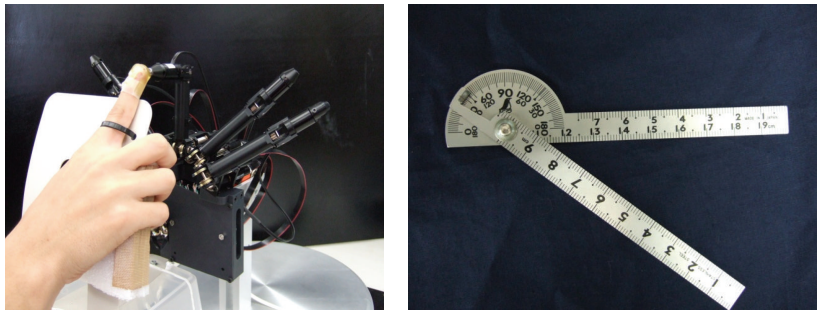
No.	Measurement conditions(*)
1	(M1) and (a)
2	(M1) and (b)
3	(M1) and (c)
4	(M2) and (a)
5	(M2) and (b)
6	(M3) and (c)

(*) (M1) and (M2) are conditions for spatial variation, and (a)–(c) are conditions for time variation.

(c) The cycle in which the force **F** was presented for 0.2 (s) and the force was not presented for 0.8 (s) was repeated until the participant answered. That is, we set $t_1 = 0.2$ (s), $t_2 = 1.0$ (s) in Figure 6.

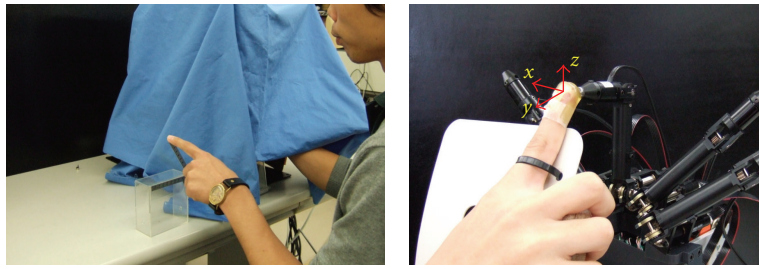
Ten participants carried out the measurement under conditions (a), (b), and (c) for measurements (M1) and (M2). That is, we considered six measurements as shown in Table 2. All participants carried out the six experiments. In particular, to circumvent the effect of the sequence of measurement, the sequence of each participant's measurement was decided in random order. In each measurement, HIRO III showed the force to the participant after an operator of HIRO III gave the signal to start, and the participant felt the force. After the participant recognized the force direction, he/she signaled the operator of HIRO III and the operator stopped the presentation of the force. The participant then responded to the direction of the presented force by using the measuring instrument. To evaluate the measurement, we noted the angular error between the presented force direction by HIRO III and the answered angle by using a measuring instrument. We note that HIRO III cannot present the accurate force when the presented time of the force is shorter than 0.2 (s). Thus we set the minimum presented time at 0.2 (s) in this experiment. As an example, we show the step response of HIRO III in Figure 7(a). In the measurement of the step response, HIRO III was connected to the wall as shown in Figure 7(b), and we measured the step response when HIROIII pressed the wall straight.

3.1.2. Experimental Results. Figures 8(a) and 8(b) show the measurement results in the horizontal direction (M1) and the vertical direction (M2), respectively. In each figure, the horizontal axis shows the experimental condition and the vertical axis shows the average value of the angular error between the presented force direction and the answered force direction. That is, we show the value $(1/130)\sum_{i=1}^{10}\sum_{j=1}^{13}|q_p^{i,j} - q_a^{i,j}|$, where $q_p^{i,j}$ is the angle of the force that presented the jth time to the ith participant in the corresponding condition, and $q_a^{i,j}$ is the angle that the ith participant answered on the jth time. The vertical bar shows the standard variation of the corresponding value. From the experimental results, we find that human perception ability regarding the force direction had an average error of 0.23 (rad) in the horizontal direction and 0.25 (rad) in the vertical direction, in the case of condition (a). In other words,



(a) Connection of HIRO III to the index finger

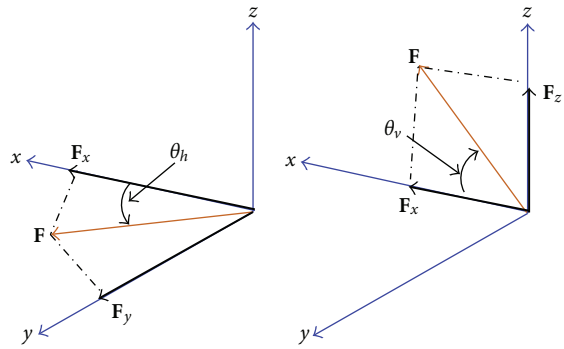
(b) Goniometer



(c) Experimental setup

(d) Coordinate systems of the finger

Figure 4: Measurement environment.



(a) Presented force in the x-y plane

(b) Presented force in the x-z plane

Figure 5: Presented force.

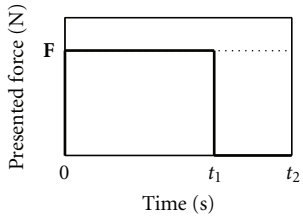
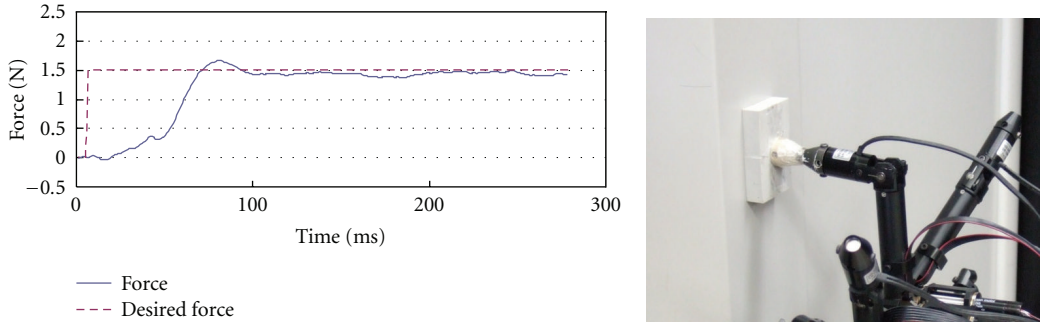


Figure 6: Form of the presented force.

there is no big difference between the perception ability of the force in the horizontal plane and in the vertical plane. On the other hand, the average values of conditions (b) and (c) in the horizontal direction were slightly large in contrast with the value of (a), and the average values of (b) and (c) in the vertical direction were slightly small in contrast with the value of (a). However, according to the analysis of variance (ANOVA) with a 5% significance level, there is no significant difference between the conditions (a) (namely, continuous force) and (b)-(c) (namely, discontinuous force).

Figure 9 shows the average value of the angular error at each presented angle. Figures 9(a) and 9(b) show the results in the horizontal direction (M1) and the vertical direction (M2), respectively. In both figures, the top figure is the result of (a), the middle figure is the result of (b), and the bottom figure is the result of (c). From Figure 9(a), we see that the angular error when the presented force is $\pi/2$ (rad) is the smallest, and it grows as the presented angle approaches 0 and π (rad). This is true in all three conditions. Further, from



(a) Step response of HIRO III

(b) HIRO III is connected to the wall

Figure 7: Performance of HIRO III (Step response).

Figure 9(b), in the vertical direction, the same tendency as the horizontal direction can be perceived. Here note that this tendency, anisotropy, was also described for the perception of the human hand [25, 26]. In [25], the participants held a joystick, and the perception ability concerning the direction of the human hand (not the individual finger) was investigated. The anisotropy of the human hand perception for the direction was shown. In [26], the perception ability concerning the force magnitude of the human hand (not the individual finger) was found to be anisotropic. Based on the above results, we concluded the following: the human perception ability of the force direction has 0.23 (rad) error in the horizontal direction and 0.25 (rad) error in the vertical direction, and whether the presented force is continuous or discontinuous has no influence on the human perception.

We wondered why the angular error increased around 0 and π (rad) of the presented force direction. To investigate whether this was an influence of the performance of the haptic interface, we connected HIRO III with a wall, as shown in Figure 7(b), and then HIRO III presented force as expressed in (1) and (2) with $\|\mathbf{F}\|$=1.5 (N) against the wall. Figure 10 shows the angular error. Figures 10(a) and 10(b) show the results in the horizontal direction and the vertical direction, respectively. From these results, we conclude that the angular error was not caused by the presented force direction and that the angular errors occurred in all the presented force directions were very small. Thus it is not easy to conclude that the above-mentioned phenomenon has happened because of the haptic interface.

3.2. Measurement of the Human Perception of Force Magnitude. We considered the perception of force magnitude with regard to spatial variation.

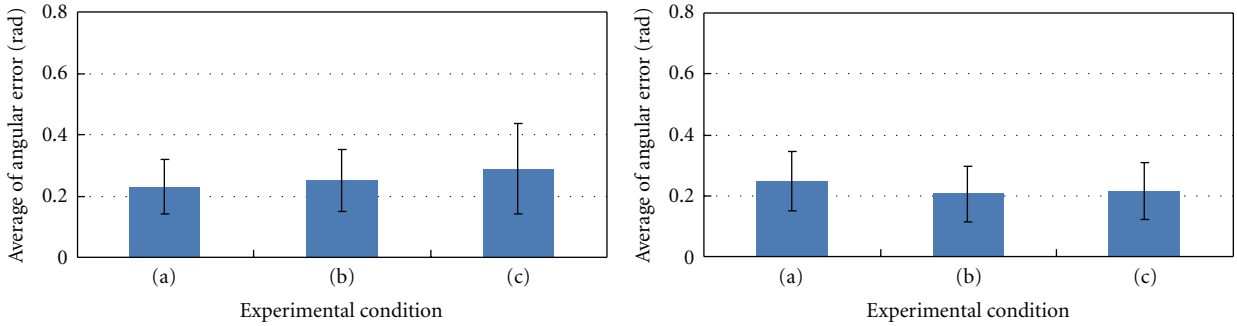
3.2.1. Experimental Setup. Ten people in their twenties (nine males and one female) participated in this measurement. All of the participants were right-handed. The measurement environment was the same as that described in Section 3.1. Here, we measured the point of subjective equality (PSE) of the force magnitude, where the compared forces are presented in more than one direction. As in the experiment of Section 3.1, the participant connected his/her index finger to HIRO III. Then, the following three terms were carried out under conditions (M1) (horizontal direction) and (M2) (vertical direction).

(1) By using HIRO III, the standard stimulus force was presented to the participant, where the standard stimulus force was presented in the reference direction.

(2) By using HIRO III, the comparison stimulus force was presented to the participant, where the comparison stimulus force was presented in a comparison direction.

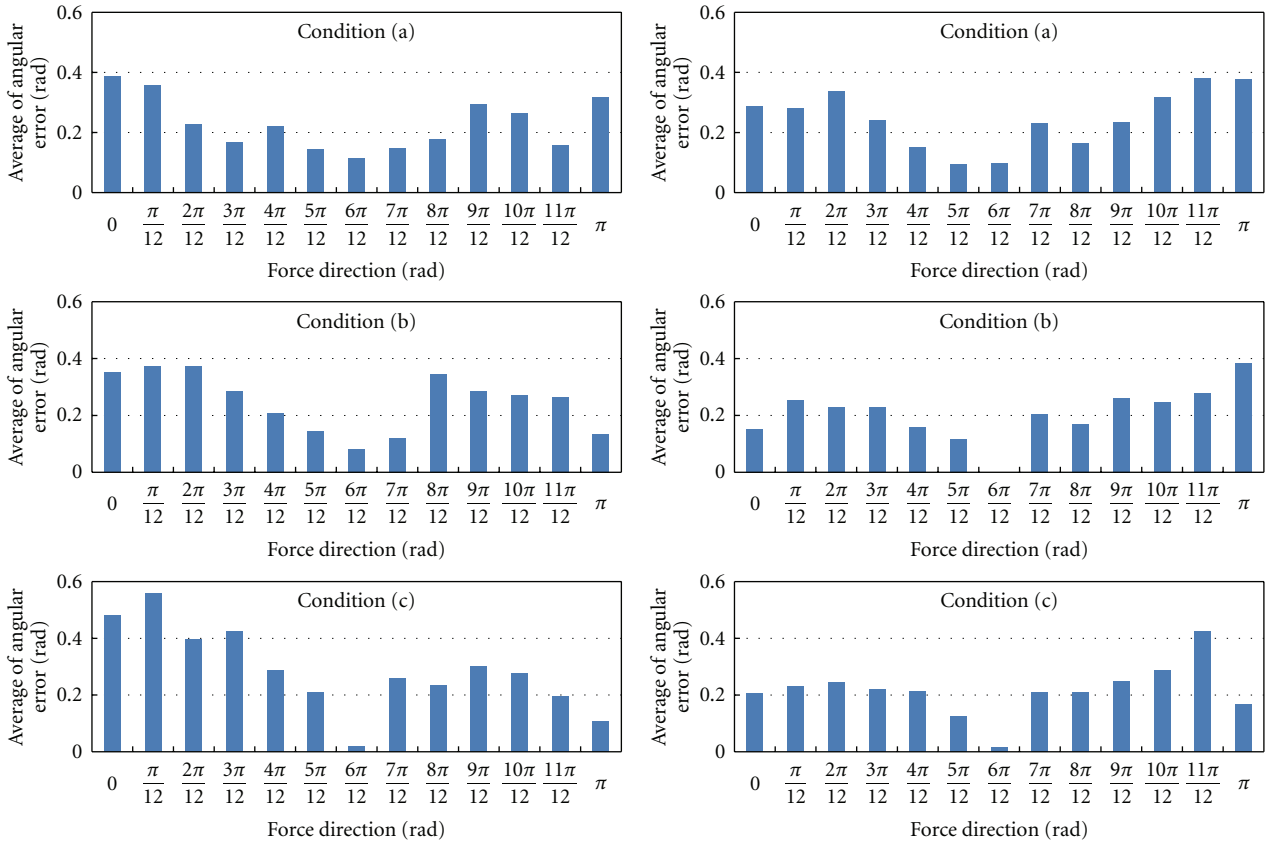
(3) The participant selected one answer from the following three answers: (i) the comparison stimulus force was larger than the standard stimulus force, (ii) the comparison stimulus force was the same as the standard stimulus force, or was not distinguishable, and (iii) the comparison stimulus force was smaller than the standard stimulus force.

The force was presented until the participant answered. The reference directions of (M1) and (M2) were $\theta_h = \theta_v = \pi/2$ (rad), and the comparison direction with respect to the reference direction was in the following five directions: 0, $\pi/4$, $\pi/2$, $3\pi/4$, π (rad). Further, we set the standard stimulus force at 2.2 (N), and we consider the following 11 comparison stimulus forces: 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6, 2.8, 3.0 (N). Before the experiment, we conducted a preliminary experiment several times. From the preliminary experimental result, we found there were comparison directions for which we could not obtain the PSE when the standard stimulus force was set smaller than 2.2 (N). Thus, to obtain the PSE in all comparison directions (0, $\pi/4$, $\pi/2$, $3\pi/4$, π (rad)), we set the standard stimulus force to 2.2 (N). For each participant, all the comparison stimulus forces were presented in the comparison directions, where the order of the presentation of the comparison stimulus force was random, and the comparison direction was selected from the above five directions before the measurement. After the measurement, we selected another comparison direction and then conducted measurements (1)–(3). The number of times that the participant answered "the comparison stimulus force is the same as the standard



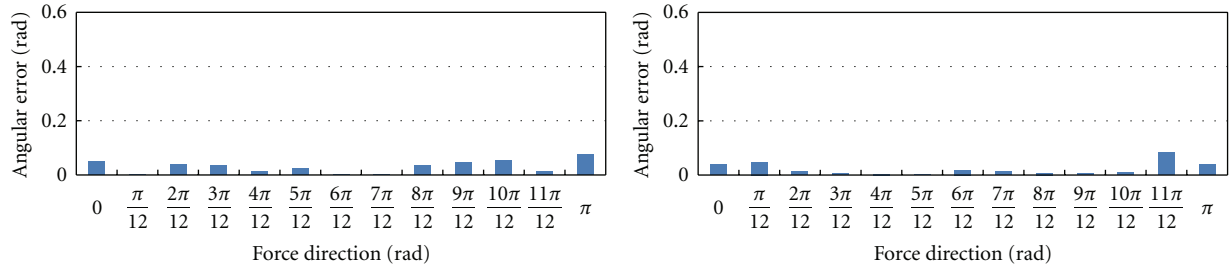
(a) Average value of the angular error in the horizontal direction (M1) (b) Average value of the angular error in the vertical direction (M2)

Figure 8: Measurement results of the angular error.



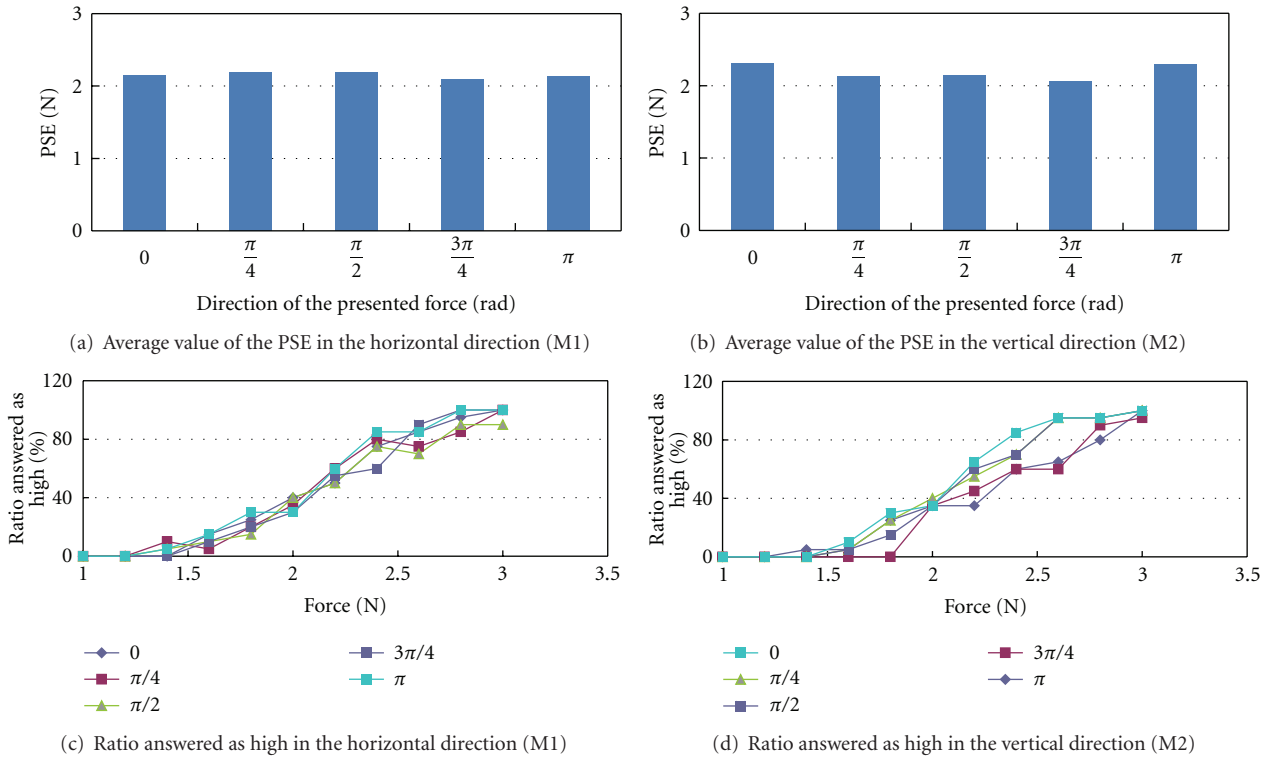
(a) Average value of the angular error in the horizontal direction (M1) (b) Average value of the angular error in the vertical direction (M2)

Figure 9: Measurement results of the angular error at each presented force direction.



(a) Average value of the angular error in the horizontal direction (M1) (b) Average value of the angular error in the vertical direction (M2)

Figure 10: Performance of HIRO III (angular error of the presented force).



(a) Average value of the PSE in the horizontal direction (M1)

(b) Average value of the PSE in the vertical direction (M2)

(c) Ratio answered as high in the horizontal direction (M1)

(d) Ratio answered as high in the vertical direction (M2)

Figure 11: Measurement results of the PSE at each direction of the presented force.

stimulus force, or is not distinguishable" was halved and added to the number of times that the participant answered "the comparison stimulus force is larger than the standard stimulus force" and the number of times that the participant answered "the comparison stimulus force is smaller than the standard stimulus force". We derived the ratio f_i of the number of times that the participant answered "the comparison stimulus force is larger than the standard stimulus force". By using the least-squares method, we derived the approximate curve of f_i, and we set the PSE as the force when f_i is 50%.

3.2.2. Experimental Results. Figures 11(a) and 11(b) show the value of the PSE in the horizontal direction and the vertical direction, respectively. In the figure, the horizontal axis is the direction of the presented force, and the vertical axis is the value of the PSE. From the figure, we see that the same tendency was obtained in both directions, indicating that the participant correctly recognized the magnitude of the presented force at all directions of the presented force. Further, we could not obtain the phenomenon of Section 3.1.2: the angular error increased when the presented force tends to 0 and π (rad). For reader's reference, we show the ratio that the participants answered "the comparison stimulus force is larger than the standard stimulus force" in Figures 11(c) and 11(d). Note that the experiment of Sections 3.1.2 and 3.2.2 considered human perception ability under the condition that the direction of the presented force was unknown or the direction of the presented force (comparison force) differed from the standard stimulus force. Thus, as a difference perception, we are attracted to the perception ability when the direction of the presented force is known (or the direction of the comparison stimulus force is the same as the standard stimulus force). To examine this, we next measured the perception ability when the direction of the presented force was well known.

3.3. Measurement of the Human Perception of Force Magnitude When the Direction Is Well Known. We measured the human perception ability with a force presented in a well-known direction. In Section 3.2, we considered the PSE under the condition that the direction of the comparison stimulus force differed from the direction of the standard stimulus force. To examine the perception of the force in a well-known direction, we measure the PSE with the condition that the direction of the comparison stimulus force was the same as the direction of the standard stimulus force.

3.3.1. Experimental Setup. Ten people in their twenties (nine males and one female) participated in this measurement. All of the participants were right-handed. The measurement environment was the same as that described in Section 3.1. As in the experiment of Section 3.1, the participant connected his/her index finger to HIRO III. Then, the following three terms were carried out under conditions M1 (horizontal direction) and M2 (vertical direction).

(1) By using HIRO III, the standard stimulus force was presented to the participant.

(2) By using HIRO III, the comparison stimulus force was presented to the participant.

(3) The participant selected one answer from the following three answers: (i) the comparison stimulus force was larger than the standard stimulus force, (ii) the comparison stimulus force was the same as the standard stimulus force, or was not distinguishable, and (iii) the comparison stimulus force was smaller than the standard stimulus force.

The force was presented until the participant answered. In this experiment, for M1 and M2, we measured the PSE in the following five directions: $\theta_h = 0, \pi/4, \pi/2, 3\pi/4, \pi$ (rad) and $\theta_v = 0, \pi/4, \pi/2, 3\pi/4, \pi$ (rad), respectively. Further, we set the standard stimulus force to 2.2 (N), and we considered the following 11 comparison stimulus forces: 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6, 2.8, 3.0 (N). For each participant, the comparison stimulus forces were presented in random order. The difference between the measurements of Sections 3.2 and 3.3 was that in the measurement of Section 3.2, the direction of the comparison force differed from the direction of the standard stimulus force, while in Section 3.3, the direction of the comparison force was the same as the direction of the standard stimulus force.

3.3.2. Experimental Results. Figures 12(a) and 12(b) show the value of PSE in the horizontal direction and the vertical direction, respectively. In these figures, the horizontal axis shows the direction in which the PSE was measured, and the vertical axis shows the PSE value. From these figures, we see that the same tendency was obtained in both directions. Furthermore, there was no difference between the PSE values with respect to the direction of the presented force. For reader's reference, we show the ratio that the participants answered "the comparison stimulus force is larger than the standard stimulus force" in Figures 12(c) and 12(d).

From the results of this section, it seems that human ability to perceive the force direction is anisotropic and the ability to perceive the force magnitude is isotropic. Further, it seems the following: compared with the case that the human answers the force magnitude, he/she makes the ambiguous answer when he/she answers the force direction.

4. Skill Transfer System and Its Experimental Evaluation

4.1. Transfer Method of Force and Position Information. For the skill transfer examined in this study, the aim of the trainee is to make his/her five-fingertip positions and forces track a trainer's five-fingertip positions and forces, respectively.

For fingertip positions tracking, we used the visual cues shown in Figure 13. The five-fingertip positions of the trainer and trainee are shown as small circle in VR space, and the trainee controlled his/her fingertip positions to track the trainer's positions.

For the fingertip forces tracking, we considered the transfer method of force based on the measurement of the human perception as described in the previous section. In the expert skill transfer system, there are two kinds of forces transferred to the user, as follows:

(1) the reaction force $\mathbf{F}_r$ from the virtual object (referred to herein as the reaction force),

(2) the force $\mathbf{F}_{\text{trainer}}$ that a trainer exerts on an object. (When we present this force to the user, we consider the force in the opposite direction, i.e., $-\mathbf{F}_{\text{trainer}}$. In the following, the force $-\mathbf{F}_{\text{trainer}}$ is called the trainer's force, $\mathbf{F}_t$.)

As the force transfer method, the reaction force, $\mathbf{F}_r$, and the trainer's force, $\mathbf{F}_t$, were presented to the user and were switched over time, as shown in Figure 14. In particular, if the time to show the trainer's force was long, the user could not feel the reaction force; so, it was necessary to shorten the time of presenting $\mathbf{F}_t$ as much as possible. Therefore, by considering the results in Section 3, we set HIRO III to show $\mathbf{F}_r$ to the user for 0.5 seconds and then to show $\mathbf{F}_t$ to the user for 0.2 seconds and then repeat the process. In practice, if the force such as that shown in Figure 14 was presented to the user, the user would feel the pulse force that is the difference between the trainer's force and the trainee's force. Thus if the user regulates his/her fingertip forces so that the pulse forces become small, the force transfer is achieved. That is, if the user grasps the virtual object by using the same force as the trainer, the pulse forces disappear. This method has the following advantages: (1) even if the force changes periodically, both force $\mathbf{F}_r$ and $\mathbf{F}_t$ can be recognized; (2) the number of visual cues for the skill transfer decreases in contrast with the method described in Section 2.2. In the next section, we consider an experiment to evaluate this method.

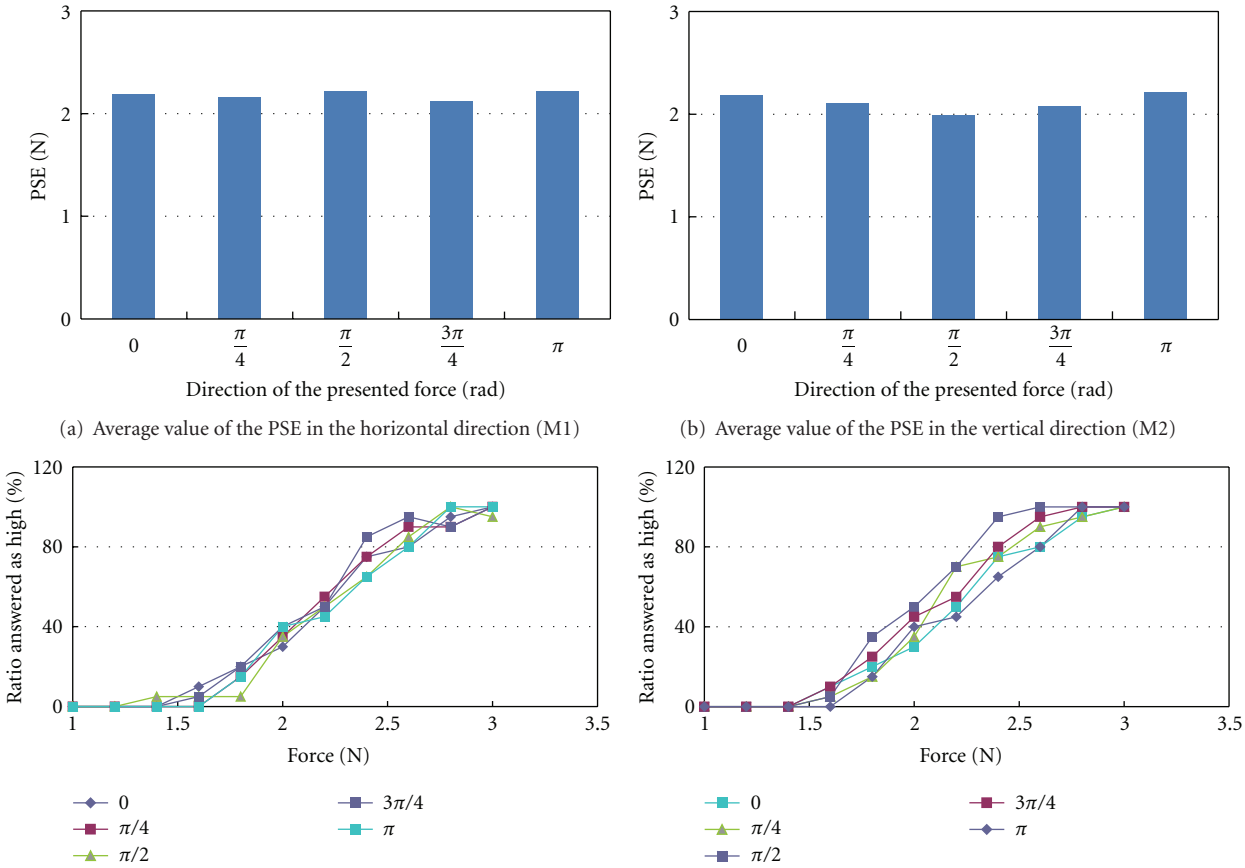
4.2. Skill Transfer System and Its Experimental Evaluation. Let us evaluate the skill transfer system based on the results reported in the previous subsection. As the task of a trainer, we consider the grasping of an object in VR space. The goal of the trainee is to assume a fingertip position and exert a grasping force that agree with the fingertip position and grasping force of the trainer, respectively. Figure 15(a) shows the experimental environment. In this figure, the black box is the display system used to present the image at around the fingertips [32]. By using this display, the trainee sees the screen as shown in Figure 15(b).

4.2.1. Experimental Setup. For the comparison, we consider the following two kinds of skill transfer method.

(P1) The fingertip positions of the trainer and the trainee are shown as small circles, and the fingertip forces of the trainer and trainee are shown as the force gauge (described in Section 2.2) in VR space.

(P2) The fingertip positions of the trainer and the trainee are shown as small circles, and the trainer's fingertip forces are presented to the trainee by using the method described in Section 4.1.

As the task of the trainer, we considered the following procedure: (1) the trainer approached the virtual object,



(a) Average value of the PSE in the horizontal direction (M1)

(b) Average value of the PSE in the vertical direction (M2)

(c) Ratio answered as high in the horizontal direction (M1)

(d) Ratio answered as high in the vertical direction (M2)

Figure 12: Measurement results of the PSE.

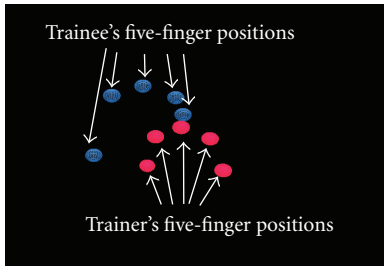


Figure 13: Visual cues for position tracking.

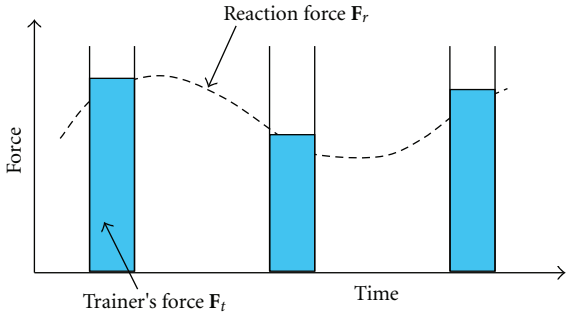
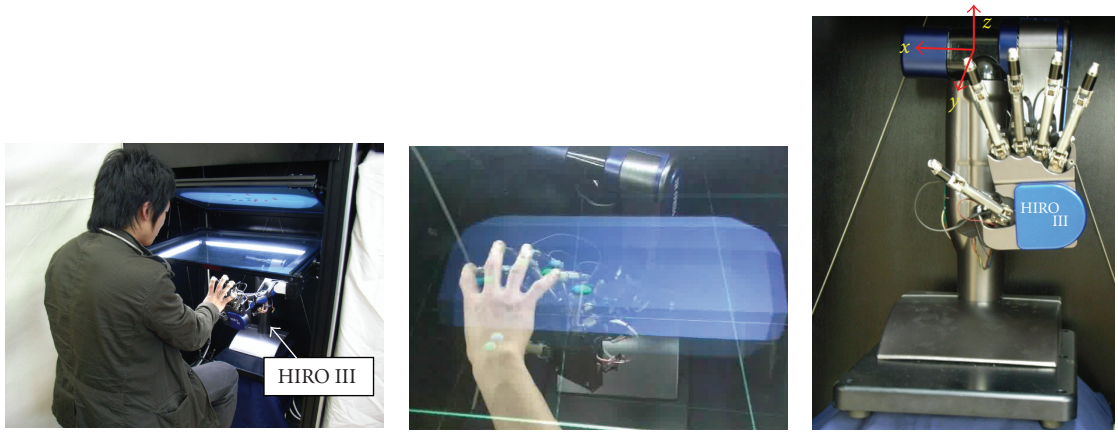


Figure 14: Presentation of two kinds of forces, $\mathbf{F}_r$ and $\mathbf{F}_t$.

and (2) the trainer grasped the object. The stiffness and the damping coefficient of the object are 570 (N/m) and 2×10^{-3} (Ns/m), respectively. Before we carried out our experiment, the person who acted as the trainer performed the task. This person was not included among the eight participants in the experiment described below. After we obtained the force and position trajectories of the trainer's fingertips, we set the trajectories as the trainer's trajectories. For example, the force trajectory of the trainer's middle finger and the position trajectory of the trainer's middle finger are shown in Figures 16(a) and 16(b), respectively. We used the coordinates in Figure 15(c).

To evaluate the above two methods, we prepared two experiments (E1 and E2). Before we performed the experiments, all participants familiarized themselves with manipulating HIRO III.

(E1) (1) The participant confirmed the trainer's trajectory. That is, the fingertip positions of the trainer were shown as small circles graphically in VR space, and the participant saw and confirmed the trajectories of the trainer's fingertips. (2) Then, the fingertip positions of the trainer and the trainee were shown as small circles graphically in VR space, and the trainee carried out the task based on this visual information



(a) Experimental environment (b) The virtual object (c) The coordinate system of HIRO III

Figure 15: Experimental environment of skill transfer.

Table 3: The sequence of experiments E1 and E2.

Order of experiment	Group A	Group B
1	E1	E2
2	E2	E1

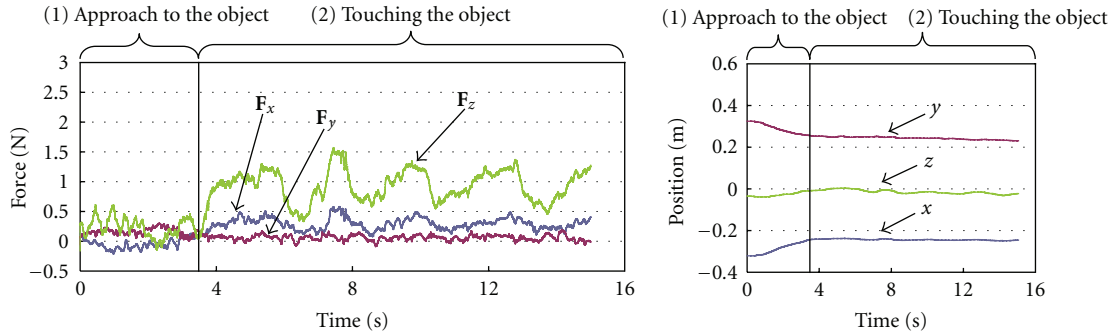
only, while we measured the initial errors. (3) The trainee carried out the task 20 times continuously under method P1. (4) Finally the trainee carried out the task under the condition that the fingertip positions of the trainer and the trainee were shown as small circles graphically in VR space, and we measured the error after training.

(E2) The trainee carried out the same experiment as E1, with P2 in place of P1.

Eight people in their twenties (seven males and one female) participated in this measurement, and we divided the subjects into two groups, group A and group B. So there would not be an effect caused by the sequence of experiments; the subjects in group A carried out experiment E1 first and then carried out experiment E2. The subjects in group B carried out experiment E2 first and then carried out experiment E1. Table 3 shows the sequence of experiments.

4.2.2. Experimental Results. Figures 17(a), 17(b), and 17(c) show the experimental results. In (a), the vertical axis shows the average value of the position error between the trainer's position and the trainee's position. The trainer's work lasted 15 (s), and the sampling time of the PC was 1 (ms). The horizontal axis shows the experimental condition. Figure 17(b) shows the average value of the error of the force magnitude, and Figure 17(c) shows the average value of the directional error of the force. When we use the polar coordinate, we can express the force by using two variables, θ and φ, for example, $F_x = \|\mathbf{F}\| \sin\theta\cos\varphi$, $F_y = \|\mathbf{F}\| \sin\theta\sin\varphi$, and $F_z = \|\mathbf{F}\| \cos\theta$. As the error of the force direction, we considered the following value: $(1/10)(1/15001)\sum_{j=1}^{10}\sum_{i=0}^{15000}(|\theta_d^i - \theta^{i,j}| + |\varphi_d^i - \varphi^{i,j}|)/2$, where θ_d^i and φ_d^i are the trainer's angle variables at i (ms), and $\theta^{i,j}$ and $\varphi^{i,j}$ are the jth participant's angle variables at i (ms). In each figure, the vertical bar shows the standard variation of the corresponding value.

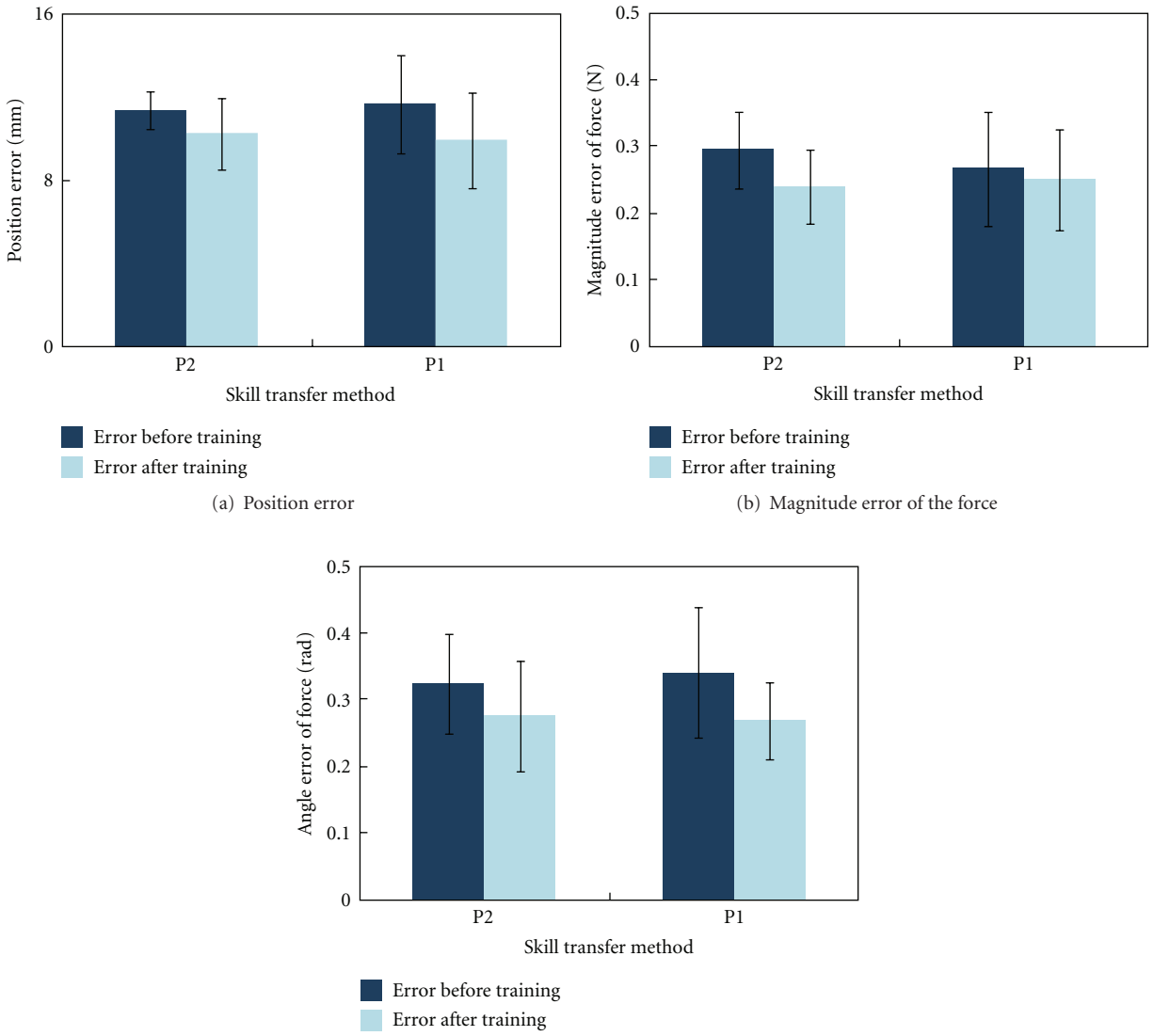
According to the t-test with a 1% significance level, there are significant differences between the errors before training P2 and the errors after the training P2. On the other hand, in the error of the force magnitude in P1, there is no difference between the errors before training and the errors after the training (even if we consider the t-test with a 5% significance level, there was no difference). Regarding the participants' opinions, we obtained the following. In P1, there were a lot of visual cues and the users were confused. The users can track his/her one or two positions and one or two forces to the corresponding trainer's positions and forces, but they were confused when they consider the five positions and forces. In contrast with this, cues were intuitive and comprehensible in P2. Many participants said that even if the position and the magnitude of the force were understood, the difference in the direction of the force was not understood. However, on the result of Figure 17(c), there are significant difference between the errors before training (P1 and P2) and the errors after the training (P1 and P2). We have considered that this is because of the simple task. In this experiment, we considered the simple task that the user just pushes the object, thus eliminating variations in the force direction while the user performs the task. In fact, the variation of z-axis force of the trainer was large, but the variations of x- and y-axis force were small, as shown in Figure 16(a). Therefore, it seems that there were differences between the errors before training and the errors after the training even if we used method P1. We cannot draw any certain conclusions because of the small sample size. However, from Figure 17(b) and the subjects' opinions, it seems to be more effective to present the trainer's force periodically rather than using only visual cues. Further, it seems that we need another assistance cue when we consider the training of the direction of the force.



(a) Force trajectories: middle finger

(b) Position trajectories: middle finger

Figure 16: The force and position trajectory of the trainer.



(a) Position error

(b) Magnitude error of the force

(c) Angle error of the force

Figure 17: Experimental results.

5. Conclusions

In this study, we investigated the human perception of force with respect to the following changes when a user engaged with the five-fingered haptic interface HIRO III: (1) the spatial change of the presented force, and (2) the change of the time to present the force. Since the force can be expressed by the direction vector and the magnitude, we first measured the perception of the force direction, as described in Section 3.1. From Figure 9(a), we see that the angular error when the presented force direction is $\pi/2$ (rad) is the smallest and it grows as the presented force direction approaches 0 and π (rad) in the horizontal direction. This is true in the vertical direction as well. However, several parts of Figure 9 show that the angular error when the presented force direction is 0 or π (rad) is smaller than the angular error when the presented force direction is $\pi/12$ or $11\pi/12$ (rad). We believe that this was due to the measuring instrument we employed. We used a goniometer as the measuring instrument, and its measuring range is $0\sim\pi$ (rad). Therefore, even if the participant feels an angle larger than π (rad) or an angle smaller than 0 (rad), he/she answered π (rad) or 0 (rad).

From Figure 8, which shows the measurement results of the average angular error, we see that the human perception ability regarding the force direction had an average error of 0.23 (rad) in the horizontal direction and 0.25 (rad) in the vertical direction. In addition, we determined that there is no difference in human perception of force direction between when the presented force is continuous and when the presented force is discontinuous, based on ANOVA with a 5% significance level.

Next, we measured the perception of magnitude considering the force direction. In particular, in Section 3.2 we described the measurement when the direction of the comparison stimulus force differed from the direction of the standard stimulus force (for simplicity, we called the direction "unknown"), and in Section 3.3, we described the measurement when the direction of the comparison stimulus force is the same as the direction of the standard stimulus force (for simplicity, we called the direction "well known"). From Figures 11 and 12, we see that there was no change caused by the presented force direction on the value of PSE. This differed from the case of perception of the force direction. In particular, by using ANOVA with a 5% significant level, we determined that there was no difference between the following four groups: PSE in the horizontal direction when the direction is unknown, PSE in the vertical direction when the direction is unknown, PSE in the horizontal direction when the direction is well known, and PSE in the vertical direction when the direction is well known. Therefore, in both the horizontal and vertical directions, there was no difference in the feeling of magnitude caused by the difference of the presented force direction, and we conclude that humans perceive magnitude, regardless of the presented force direction.

These results of perception measurements showed that the human perception of the fingertip force direction is anisotropic and perception of the fingertip force magnitude is isotropic. Furthermore, regarding the perception of the fingertip force direction, there is no difference between the perception of the continuously presented fingertip force and the discontinuous presentation.

We also considered the transfer method based on our measurement results by using the multi-fingered haptic interface HIRO III. For skill transfer, the reaction force $\mathbf{F}_r$ and the trainer's force $\mathbf{F}_t$ were transferred to the user. The proposed skill transfer system consists of the following two parts: (1) for the fingertip position tracking, we used the visual cue shown in Figure 13, and (2) for the fingertip force tracking, HIRO III alternately showed $\mathbf{F}_r$ and $\mathbf{F}_t$. This method has the following advantages: (1) even if the force changes periodically, both force $\mathbf{F}_r$ and $\mathbf{F}_t$ can be recognized, and (2) the number of visual cues decreases, in contrast with the method described in Section 2.2.

We performed tests to demonstrate the validity of the proposed method. According to the t-test with a 5% significance level, in the proposed method, there was a difference between the error before training and the error after training. Therefore, we believe that the transfer of the force was achieved more efficiently by presenting the force intermittently, as described in Section 4.1.

The next problem to be tackled is to increase the number of experimental subjects (in particular, the number of female subjects) and to confirm how the results change at that time. We must also extend this research to a more complicated skill transfer system, such as the human body model in VR space. Further, we will attempt to develop an efficient transfer method for transmitting the direction of force.

Acknowledgment

This paper was supported by the Ministry of Internal Affairs and Communication R & D Promotion Programme (SCOPE).

References

[1] Y. Yokokohji, R. L. Hollis, T. Kanade, K. Henmi, and T. Yoshikawa, "Toward machine mediated training of motor skills-skill transfer from human to human via virtual environment," in *Proceedings of the IEEE International Workshop on Robot and Human Communication*, pp. 32–37, Tsukub, Japan, 1996.

[2] R. Kikuuwe and T. Yoshikawa, "Haptic display device with fingertip presser for motion/force teaching to human," in *Proceedings of the EEE International Conference on Robotics and Automation*, vol. 1, pp. 868–873, Seoul, South Korea, 2001.

[3] S. Saga, N. Kawakami, and S. Tachi, "Haptic teaching using opposite force presentation," in *Proceedings of the 1st Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '05)*, vol. 1, Pisa, Italy, 2005, (CD-ROM).

[4] G. Srimathveeravalli and K. Thenkurussi, "Skill training assistance using haptic attributes," in *Proceedings of the 1st Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '05)*, vol. 1, pp. 209–214, Pisa, Italy, 2005.

[5] M. Nakao, K. Minato, T. Kuroda, M. Komori, H. Oyama, and T. Takahashi, "Transferring bioelasticity knowledge through haptic interaction," *IEEE Multimedia*, vol. 13, no. 3, pp. 50–60, 2006.

[6] Y. Kuroda, M. Hirai, M. Nakao, et al., "Construction of training environment for surgical exclusion with a basic study of multi-finger haptic interaction," in *Proceedings of the 2nd Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '07)*, pp. 525–530, Tsukuba, Japan, March 2007.

[7] N. E. Seymour, A. G. Gallagher, S. A. Roman, et al., "Virtual reality training improves operating room performance: results of a randomized, double-blinded study," *Annals of Surgery*, vol. 236, no. 4, pp. 458–464, 2002.

[8] P. L. Youngblood, S. Srivastava, M. Curet, W. L. Heinrichs, P. Dev, and S. M. Wren, "Comparison of training on two laparoscopic simulators and assessment of skills transfer to surgical performance," *Journal of the American College of Surgeons*, vol. 200, no. 4, pp. 546–551, 2005.

[9] D. Morris, T. Hong, F. Barbagli, T. Chang, and K. Salisbury, "Haptic feedback enhances force skill learning," in *Proceedings of the 2nd Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '07)*, pp. 21–26, Tsukuba, Japan, March 2007.

[10] V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, and G. Robles-De-La-Torre, "Haptic interfaces and devices," *Sensor Review*, vol. 24, no. 1, pp. 16–29, 2004.

[11] A. El Saddik, "The potential of haptics technologies," *IEEE Instrumentation & Measurement Magazine*, vol. 10, no. 1, pp. 10–17, 2007.

[12] K. E. Maclean and V. Hayward, "Do it yourself haptics: part II," *IEEE Robotics & Automation Magazine*, vol. 15, no. 1, pp. 104–119, 2007.

[13] N. Magnenat-Thalmann and U. Bonanni, "Haptics in virtual reality and multimedia," *IEEE Multimedia*, vol. 13, no. 3, pp. 6–11, 2006.

[14] F. Barbagli, K. Salisbury, and R. Devengenzo, "Toward virtual manipulation: from one point of contact to four," *Sensor Review*, vol. 24, no. 1, pp. 51–59, 2004.

[15] Z. Najdovski and S. Nahavandi, "Extending haptic device capability for 3D virtual grasping," in *Procedings of the 6th International Conference on Haptics: Perception, Devices and Scenarios (EuroHaptics '08)*, vol. 5024 of *Lecture Notes in Computer Science*, pp. 494–503, Madrid, Spain, June 2008.

[16] L. A. Jones, "Perception of force and weight: theory and research," *Psychological Bulletin*, vol. 100, no. 1, pp. 29–42, 1986.

[17] X. D. Pang, H. Z. Tan, and N. I. Durlach, "Manual discrimination of force using active finger motion," *Perception & Psychophysics*, vol. 49, no. 6, pp. 531–540, 1991.

[18] H. Z. Tan, M. A. Srinivasan, B. Eberman, and B. Cheng, "Human factors for the design of force-reflecting haptic interfaces," in *Proceedings of the 3rd International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, vol. 55–1, pp. 353–359, ASME Dynamic Systems and Control Division, New York, NY, USA, 1994.

[19] S. Allin, Y. Matsuoka, and R. Klatzky, "Measuring just noticeable differences for haptic force feedback: implications for rehabilitation," in *Proceedings of the 10th Symposium on Haptic Interfaces for Virtual Environment & Teleoperator Systems*, pp. 299–303, Orlando, Fla , USA, 2002.

[20] S. Yamakawa, H. Fujimoto, S. Manabe, and Y. Kobayashi, "The necessary conditions of the scaling ratio in master-slave systems based on human difference limen of force sense," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 35, no. 2, pp. 275–282, 2005.

[21] S. Li, "Perception of individual finger forces during multi-finger force production tasks," *Neuroscience Letters*, vol. 4009, no. 3, pp. 239–243, 2006.

[22] S. A. Cholewiak, H. Z. Tan, and D. S. Ebert, "Haptic identification of stiffness and force magnitude," in *Proceedings of the Symposium on Haptics Interfaces for Virtual Environment and Teleoperator Systems*, pp. 87–91, Reno, Nev, USA, 2008.

[23] F. Barbagli, K. Salisbury, and H. Z. Tan, "Haptic discrimination of force direction and the influence of visual information," *ACM Transactions on Applied Perception*, vol. 3, no. 2, pp. 125–135, 2006.

[24] H. Z. Tan, F. Barbagli, K. Salisbury, C. Ho, and C. Spence, "Force-direction is not influenced by reference force direction," *Haptics-e*, vol. 4, no. 1, 6 pages, 2006.

[25] I. Elhajj, H. Weerasinghe, A. Dika, and R. Hansen, "Human perception of haptic force direction," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 989–993, Beijing, China, 2006.

[26] E. Dorjgotov, G. R. Bertoline, L. Arns, Z. Pizlo, and S. R. Dunlop, "Force amplitude perception in six orthogonal directions," in *Proceedings of the Symposium on Haptics Interfaces for Virtual Environment & Teleoperator Systems*, pp. 121–127, 2008.

[27] T. Endo, H. Kawasaki, K. Kigaku, and T. Mouri, "Transfer method of force information using five-fingered haptic interface robot," in *Proceedings of the 2nd Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '07)*, pp. 599–600, Tsukuba, Japan, March 2007.

[28] H. Kawasaki, J. Takai, Y. Tanaka, C. Mrad, and T. Mouri, "Control of multi-fingered haptic interface opposite to human hand," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems(IROS '03)*, vol. 3, pp. 2707–2712, Las Vegas, Nev, USA, 2003.

[29] H. Kawasaki and T. Mouri, "Design and control of five-fingered haptic interface opposite to human hand," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 909–918, 2007.

[30] T. Endo, H. Kawasaki, T. Mouri, et al., "Five-fingered haptic interface robot: HIRO III," in *Proceedingsof the 3rd Joint EuroHaptics conference, and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '09)*, pp. 458–463, Salt Lake City, Utah, USA, 2009.

[31] K. Kigaku, H. Kawasaki, M. O. Alhalabi, and T. Mouri, "Research on skill transmission using multi-fingered haptic interface robot," in *Proceedings of the 10th Virtual Reality Society of Japan Annual Conference*, Tokyo, Japan, 2005, (CD-ROM).

[32] Y. Yokokohji, R. L. Hollis, and T. Kanade, "WYSIWYF display: a visual/haptic interface to virtual environment," *Presence*, vol. 8, no. 4, pp. 412–434, 1999.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 437654, 9 pages
doi:10.1155/2010/437654

Research Article

Emergence of Prediction by Reinforcement Learning Using a Recurrent Neural Network

Kenta Goto and Katsunari Shibata

Department of Electrical and Electronic Engineering, Oita University, 700 Dannoharu, Oita 870-1192, Japan

Correspondence should be addressed to Kenta Goto, kenta.0510@gmail.com

Received 6 November 2009; Revised 1 March 2010; Accepted 17 May 2010

Academic Editor: Noriyasu Homma

To develop a robot that behaves flexibly in the real world, it is essential that it learns various necessary functions autonomously without receiving significant information from a human in advance. Among such functions, this paper focuses on learning "prediction" that is attracting attention recently from the viewpoint of autonomous learning. The authors point out that it is important to acquire through learning not only the way of predicting future information, but also the purposive extraction of prediction target from sensor signals. It is suggested that through reinforcement learning using a recurrent neural network, both emerge purposively and simultaneously without testing individually whether or not each piece of information is predictable. In a task where an agent gets a reward when it catches a moving object that can possibly become invisible, it was observed that the agent learned to detect the necessary factors of the object velocity before it disappeared, to relay the information among some hidden neurons, and finally to catch the object at an appropriate position and timing, considering the effects of bounces off a wall after the object became invisible.

1. Introduction

Unlike factories and laboratories, the real world is too complicated and diverse for robots to behave flexibly while following some specific programs. To develop robots that behave flexibly like humans in the real world, the robots must autonomously learn in various environments and acquire necessary knowledge and functions by themselves. The acquired knowledge and functions enable the robots to behave more appropriately, even in unfamiliar environments. Living beings acquire various functions and achieve appropriate purposes by using these skills. "Prediction" is one such function. It is a higher function that estimates a future state from the past and present states considering both dynamics of the environment and actions of the robot.

For developing highly intelligent robots, recent focus has been on autonomous learning of prediction. When we predict some information, we can usually know in the future whether or not the prediction was correct. For a learning system that predicts a future state from the present and past states and actions, this signifies that training signals can be obtained in the future even though they are not provided by humans. In this sense, the learning of prediction is autonomous although supervised learning is actually utilized.

Many studies have investigated learning of prediction. The dynamics or context that appears in a learning system to predict the future states is utilized for state representation [1–6]. In some of these, a recurrent neural network is used as the learning system [1–4]. The learning of prediction has also been used to establish curiosity-driven learning [7–10]. However, typically, the prediction target is given, that is, what information should be predicted at what future timing; in many cases, the sensor signals at the next time step are the prediction target.

Regarding the abstraction process in robots, Brooks [11] pointed out the following: "This abstraction is the essence of intelligence and the hard part of the problems being solved. Under the current scheme the abstraction is done by the researchers leaving little for AI programs to do but search." When prediction is learned, the same holds true. A robot can get many sensor signals, such as visual sensor signals, but it

seems difficult and meaningless to predict all the signals in some future. When considering the case of humans, we do not seem to predict all the visual signals at all the steps in the future. Therefore, the process of choosing a prediction target should be considered.

The information about the prediction target should be useful for achieving certain purpose and might be learned through experiences. For example, when humans chase and catch a batted ball, they seem to extract necessary information from numerous sensor signals and then predict the place where the ball will land as a prediction target and go ahead of the ball. We can easily know that "Prediction of the landing site" is useful "for catching the ball", but that must be a very intelligent process actually.

Schmidhuber has identified a very interesting and important point that we do not care either unpredictable or easily predictable information, but seem to "explore the predictable" [9]. He also proposed a learning system to realize this. However, to know whether a piece of information is predictable, the system must first conduct tests to predict the information. A method for discovering the prediction target from linear independence has also been proposed [12], but it only considers the sensor signals as inputs and does not provide a way to consider the purpose of the robot. Accordingly, it is difficult for this approach to realize a purposive prediction.

In reinforcement learning, an agent or robot learns from rewards and punishments based on trial and error. Therefore, it is a highly autonomous and purposive learning although the learning is generally very slow. If a neural network is used to connect the sensors to the motors in parallel and trained by training signals generated based on reinforcement learning, the network is optimized to obtain more rewards and less punishment, that is, to represent the value function more accurately and to generate actions with more gain for the value. Accordingly, with reinforcement learning it can be expected that the functions that contribute toward obtaining more rewards emerge in the neural network [13].

The objective of this paper is to clearly show that the prediction function, including the choice of prediction target, emerges purposively through reinforcement learning using a recurrent neural network when a given task requires prediction. In the learning system, the prediction of only predictable and useful information emerges without individual testing because only those predictions that contribute toward obtaining a reward emerge through reinforcement learning. Therefore, the system does not determine whether each piece of information is predictable or not individually. The curiosity-driven acceleration of learning is beyond the scope of this study and will be addressed in the future work.

To compensate for the missing information in solving a partially observable problem in reinforcement learning, a recurrent neural network or other finite-state controllers are often used [2, 3, 14–17]. Compensation of the missing information from the past series of sensor signals can be considered as a kind of "prediction" in a wide meaning. However, none of these studies has claimed that prediction should be considered in reinforcement learning. Furthermore, we usually call the function prediction when the environment changes dynamically, and the regularity in the dynamics is found. In the multiagent task in [15], the agent needs to predict the other agent's behavior to some extent to accomplish its purpose in a discrete state space. However, it is not shown how the agent predicts the other agent behavior and how the predicted information is represented inside the learning system after learning. Here, emergence of both spatial and temporal prediction in a continuous and dynamic environment through reinforcement learning is examined, and it is analyzed how the internal states represent the predicted information in the recurrent neural network.

2. Learning System

The learning system does not employ any special techniques for prediction, and a recurrent neural network is simply trained by the training signals that are derived autonomously on the basis of reinforcement learning. Therefore, it can be understood that reinforcement learning trains the recurrent network. Since the agent's actions are discrete in the task, while the state space is continuous, Q-learning [18] is used as a reinforcement learning algorithm.

For the recurrent neural network, a popular 3-layer Elman-type network is used, in which the outputs of hidden neurons, 40 in number, are fed back as inputs of the network at the next time step. The network is trained by back propagation through time (BPTT) technique [19]. The output function of each hidden or output neuron is the sigmoid function ranging from -0.5 to 0.5, and 0.4 is added before it can be used as a Q-value to match the value range between the output and Q-value.

The initial weights from input to hidden neurons are chosen randomly from -0.5 to 0.5, and those from hidden neurons to output are all 0.0. The initial feedback connection weights between hidden neurons are 0.0 except that the weights for the self-feedback connections are 4.0. The maximum derivative of the output function is 0.25. Therefore, under this setup, the error signal in BPTT propagates to the past without divergence, because the products of the self-feedback connection weight and the maximum derivative of output function become 1.0. Furthermore, in the forward computation, forming of bistable dynamics is promoted in each hidden neuron.

First, the present state is given as an input to the network. The recurrent network is expected to extract and store the necessary information in its hidden layer without holding the past state in the external memory. The output layer has the same number of neurons as that of the possible actions, and each output is used as the Q-value of the corresponding action.

For the action selection, a two-step stochastic selection is used. A small random number is added to each Q-value derived by the network computation, and then an action is chosen according to ε-greedy [4]. This not only assigns a higher priority to the actions with larger Q-values but also occasionally selects an action with a small Q-value. Both these random factors decrease with the number of episodes and finally become almost 0. We have not used a soft-max

type action selection, such as Boltzmann selection, because the actions with very small Q-value are rarely chosen.

In the learning phase, a training signal is given only to the output of the chosen action a_t. The training signal $T_{a_t,t}$ for the action a_t in the present state S_t is generated autonomously using the maximum Q-value at the future state s_{t+1} after the action a_t:

$$T_{a_t,t} = r_{t+1} + \gamma \max_{a'} Q_{a'}(S_{t+1}) - 0.4, \quad (1)$$

where r indicates a reward and γ indicates a discount factor. 0.4 is subtracted to match the value range between the training signal and Q-value. The discount factor γ is set to 0.96 in this task. Further, if the training signal is greater than 0.4 or less than -0.4, the value is set to 0.4 or -0.4, respectively. Equation (1) indicates that at time $t+1$, the state of the network at time t is restored and the network is trained by the training signals generated by the equation, according to BPTT.

3. Task Setting

To examine whether the prediction function emerges, we used a task in which it is impossible to achieve a purpose without a prediction. This task is performed on a field of size 7.5×3.0, as shown in Figure 1. In the task, an agent catches an object. The initial direction of motion and velocity of the object are randomly chosen for each episode, and it cannot be seen moving in some area. The object bounces off walls. The agent gets a reward when the object approaches it and the agent catches the moving object at an appropriate position and timing. Therefore, considering the bounces, the agent has to predict the appropriate position and time for the approaching object before it becomes invisible.

The bottom left corner is defined as the origin, and the initial location of the moving object is fixed at (0.0, 1.5). For each episode, its initial velocity is chosen randomly between 0.50/step and 0.70/step and its direction is chosen between -45° and 45° from the x-axis. The object's direction of motion and velocity is constant during the episode unless it bounces off a wall. When it bounces, the angles of incidence and reflection are equal, and the velocity is reduced to 80% of the previous value. The agent is fixed on the line of $x = 6.0$ and can move only in the y direction. The initial y location is chosen randomly from 0.25 to 2.75 for every episode, enabling the agent to get a reward from any initial location, each time it chooses the optimal action. At every time step, the agent can choose one of the four actions: "catch," "wait," "move up," or "move down". When it chooses move up or move down, it moves 0.25 or -0.25 in the y direction. When the action chosen is wait, the agent does not catch or move.

When the agent chooses the action catch, the episode is complete, and the agent gets a reward when the relative distance between the agent and the moving object is less than 1.0. The reward value r is generated by

$$r = 0.40 \times (2.0 - d), \quad (2)$$

where r varies depending on the relative distance d. When d is 0.0, r becomes 0.8, which is the maximum value, and when

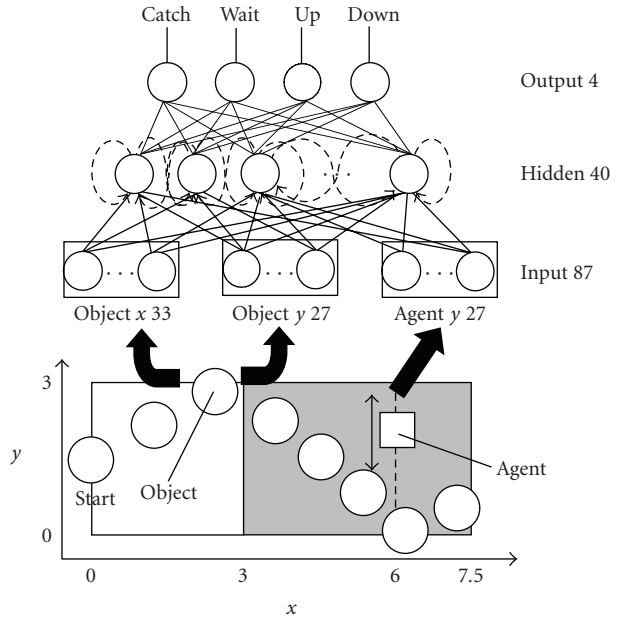


Figure 1: Object catching task and a recurrent neural network. An agent moves up or down and catches a moving object. The initial direction of motion and velocity of the object are chosen randomly for every episode. The invisibility area is also chosen randomly in the range of $x > 3.0$. x, y coordinates of the object and y coordinate of the agent are input to an Elman-type recurrent neural network. Each input signal represents local information, as shown in Figure 2.

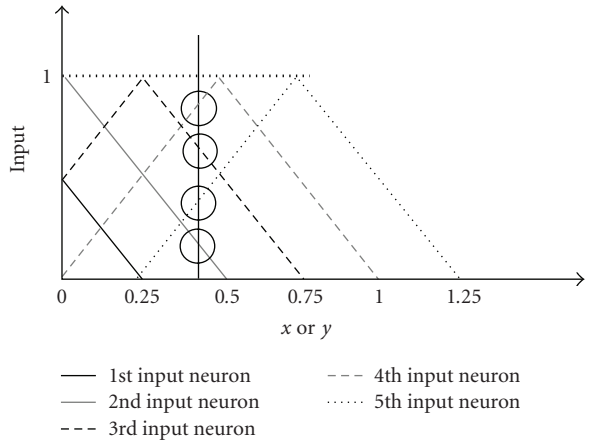


Figure 2: Localized input signals. Each input signal responds when the object or agent exists around one specific location in x or y coordinate. This localized representation helps to learn a strong nonlinear mapping.

d is 1.0, r becomes 0.4, which is the minimum value. When the agent chooses the catch action at $d > 1.0$, or when it does not choose the catch action until the object goes beyond the x limit of the field ($x = 7.5$), the episode is considered as failed and is forced to terminate. In such cases, to decrease the Q-value for the catch action, it is modified by the training signal as

$$T_{a_t,t} = r_{t+1} + Q_a(s_t) - 0.4, \quad (3)$$

and $r_{t+1} = -0.1$ is a punishment.

Three signals, the x, y coordinates of the moving object and y coordinate of the agent, are provided to the agent as inputs. However, each signal is represented by signals, each of which actually responds to a local area of the original signal. With this representation, it becomes easy for the neural network to learn a strong nonlinear input-output relation. The x position of the moving object that ranges from 0.0 to 7.5 is represented by 33 signals. Each signal represents local information as shown in Figure 2. 4 of the 33 neurons have a value other than 0.0, and the others have the value 0.0. The y position of the object or agent is represented by 27 signals. There are a total of 87 signals that are inputs of the neural network.

An invisibility area lies over the region where the x coordinate is more than 3.0. When the object is in the invisibility area, the agent cannot see it and all the input signals representing the object location are all 0.0. Both the beginning and end of the invisibility area are set randomly within a range of 3.0–7.5 for every episode on the condition that the end position is larger than the beginning position. The agent is unaware of the beginning or end of the invisibility area in advance. Therefore, the agent cannot get a reward unless it predicts the position and timing of the moving object when it comes close to the agent. The prediction should be done using information acquired before the moving object enters into the invisibility area.

4. Experimental Results and Investigation

4.1. Learning Results. The learning curve is shown in Figure 3. The horizontal axis shows the number of episodes and the vertical axis shows the average reward.

This figure indicates that the agent can catch the object more accurately through iterative learning. The maximum reward generated by (2) is 0.80, but the object and agent locations are computed on a discrete time scale. Therefore, even though the agent always chooses the optimal action, the average relative distance is 0.144, as shown in Table 1. Table 1 also shows the performance after learning, compared with the case where the agent always chooses the optimal action.

Figure 4 shows an example of agent behavior after learning. In this case, the invisibility area is maximum, that is, from $x = 3.0$ to $x = 7.5$, the velocity is 0.5/step, and the angle of the object's direction of motion from the x axis is 35°. The agent can move only in the y direction along $x = 6.0$, but for easy understanding, Figure 4 is plotted assuming the agent moves together with the object in the x direction. In this case, the initial y location of the agent is 2.0.

According to Figure 5, the changes in Q-values for all actions, except the catch action, are similar to each other; however, for the catch action, the Q-value increases suddenly from the 13th step (around $x = 4.5$), and the agent finally chooses catch action at the 16th step. Therefore, the agent is considered to choose the catch action at an appropriate time step without catching the object at a wrong time before it comes into the reward area.

Figure 5 shows the change of the Q-value for each action in this episode.

The prediction of the catch timing for variable object velocities is observed. While maintaining the angle of the direction of motion constant, as shown in Figure 4, the velocity of the object is varied, and the catch timing is observed. Figure 6 shows the change in the Q-value of the catch action for three velocities. Although the agent cannot see the object in the latter half, the Q-value suddenly increases around the reward area in all the three cases.

Next, the authors examine whether the agent can predict the object location when it comes into the reward area. Figures 7 and 8 show the y coordinate where the agent catches the object for the two cases: with no invisibility area and with the maximum invisibility area. In these figures, the horizontal axis shows the initial angle of the object's motion from the x axis, and the vertical axis shows the y position where the agent catches the object. In these cases, the initial angle is determined from −45° to 45° at an interval of 1°, and the velocity is fixed at 0.6/step. To demonstrate whether the agent can catch the object at appropriate positions, the optimal position where the agent gets the maximum reward for each angle is plotted, denoted by squares.

Figures 7 and 8 show that both position and timing of the agent catching the object are appropriate except for the case around −45° initial angle and no invisibility area. Even in the case where the agent cannot see the object in the latter half of the episode, the agent catches the object at an appropriate position, considering that the object bounces off the wall. However, it seems difficult for the agent to catch the object

Table 1: The agent's ideal and actual performance after learning for three cases of invisibility area.

	Range of the invisibility area			Ideal
	Random	Nothing	Maximum	
Average reward	0.685	0.685	0.681	0.742
Percentage with which the agent gets the reward	99.0	98.4	99.9	100
Relative distance between the agent and object when the agent chooses catch action	0.270	0.260	0.296	0.144

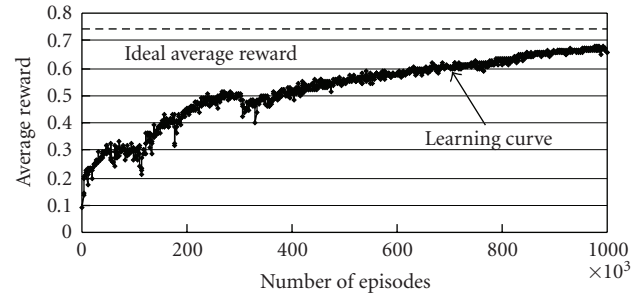


Figure 3: Learning curve. Change in the average reward according to the number of episodes is shown. Broken line shows the ideal average reward under the assumption that the agent always catches the object at the optimal timing and place.

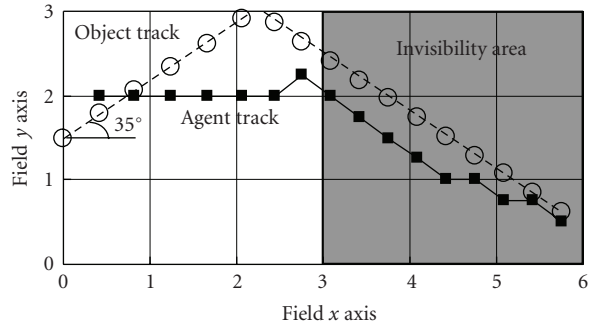


Figure 4: Sample object and agent trajectories for the object moving at 35° with a velocity of 0.50/step, and the object cannot be seen at $x > 3.0$. The agent does not move in the x direction actually, but for easy understanding, it is shown to be moving in the x direction along with the object.

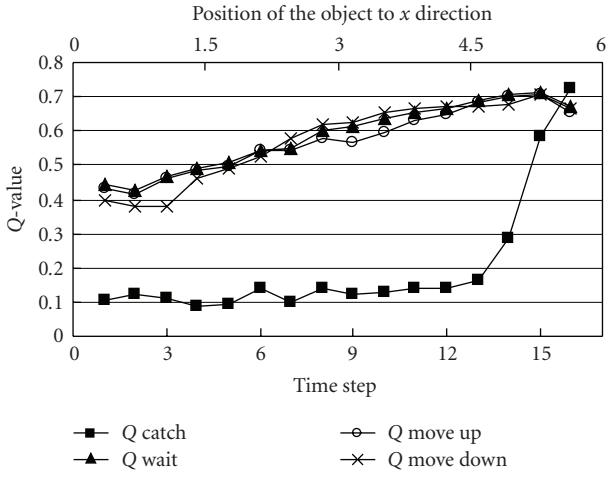


Figure 5: Change in Q-values in an episode. Four lines show the change in Q-values for the actions, "catch", "wait", "move up", and "move down". If the action is selected greedily, the action with the maximum Q-value is chosen.

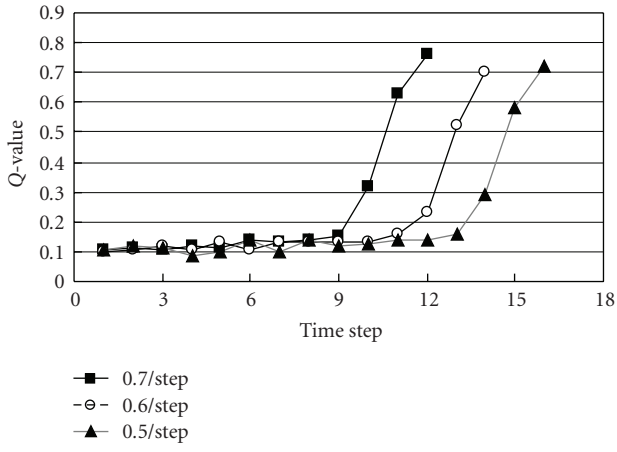


Figure 6: Change in the Q-value for "catch" action for three velocities. The timing for the increase in value differs, but the position of the object at that timing is almost the same in all cases.

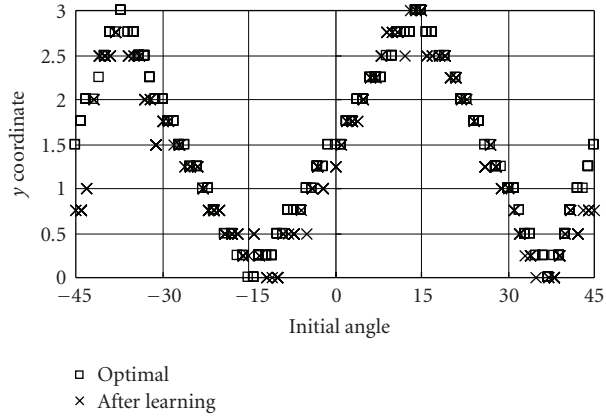


Figure 7: Positions where the agent catches the object when the initial object direction of motion varies (with no invisibility area).

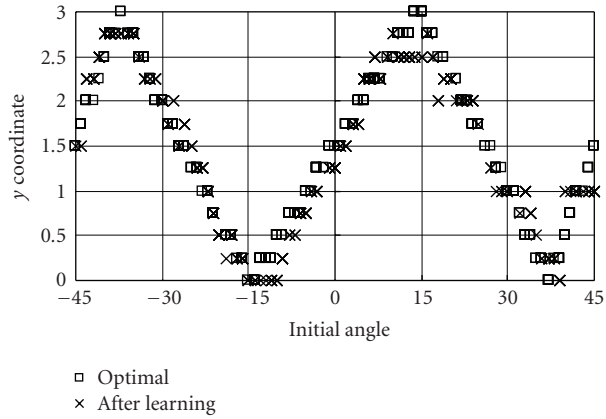


Figure 8: Positions where the agent catches the object when the initial object direction of motion varies (with the maximum invisibility area).

at around $y = 3.0$ in the case of the maximum invisibility area.

In this task, when the object bounces off a wall, its velocity is reduced to 80% of the previous value. Even if the object is in the invisibility area, it bounces and its velocity is reduced. If the agent cannot consider this property, it cannot catch the object at an appropriate timing. The authors examine whether the agent can really consider the reduction in the object velocity due to the bounce in the invisibility area, as follows.

If the velocity in the x direction is constant, the optimal time step for each catch action is equal unless the object bounces. The number of bounces increases with the increasing initial angle from the x axis, and this decreases the velocity. Then the initial angle is varied at an interval of 1°, with a constant velocity of 0.5/step in the x direction, and the time step when the agent catches the object is observed. In Figure 9, the optimal time step for catch actions is also plotted together with the observed results. In Figure 10, the optimal time step is plotted assuming that the object velocity is not reduced by the bounce in the invisibility area.

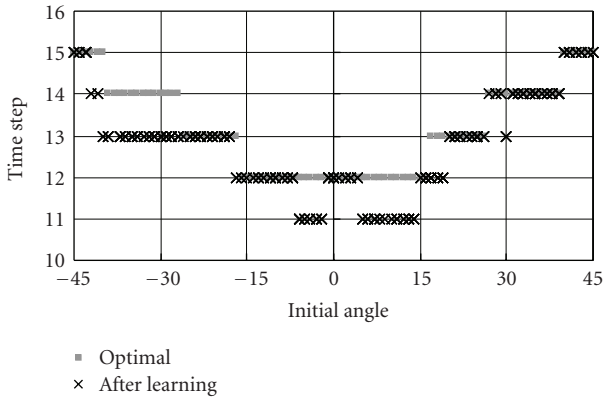


Figure 9: Catch timing when the initial motion of the object varies. The optimal catch timing is also plotted.

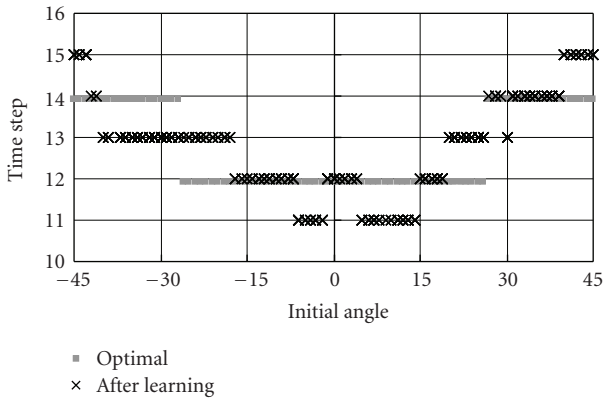


Figure 10: Catch timing when the initial motion of the object varies. The optimal catch timing under the assumption that the velocity is not reduced at the time of bounce in the invisibility area is also plotted to show whether the agent can consider the reduction in velocity due to the bounce by comparing it with Figure 9.

If the agent considers the reduction in velocity due to the bounce, the plots are more similar in Figure 9 than in Figure 10. In Figure 9, the time step is more similar to the optimal. However, the agent does not always catch the object at the optimal time step. The agent tends to catch the object a little earlier than the optimal.

These results indicate that through reinforcement learning alone, the agent can predict the necessary information such as the position or timing of the catch, considering the bounce, by using information before the object disappears.

4.2. Investigation of Hidden Neurons. In this section, the authors examine how the agent predicts the catch timing before the object enters the invisibility area, and the agent catches the object using the predicted information after some time lag. It appears that many neurons influence each other in a complex way, and prediction and catch are realized. In other words, one neuron apparently represents several pieces of information simultaneously and one piece of information is represented by several neurons. That is the same as our brain—a massively parallel system. Although it is difficult to clearly explain the mechanism, the authors try to elucidate it. In most graphs shown in the following figures, three lines represent changes in the output of the hidden neurons for each of the three cases shown in Figure 6. Some important connection weights from the neuron are indicated in the box marked "weight".

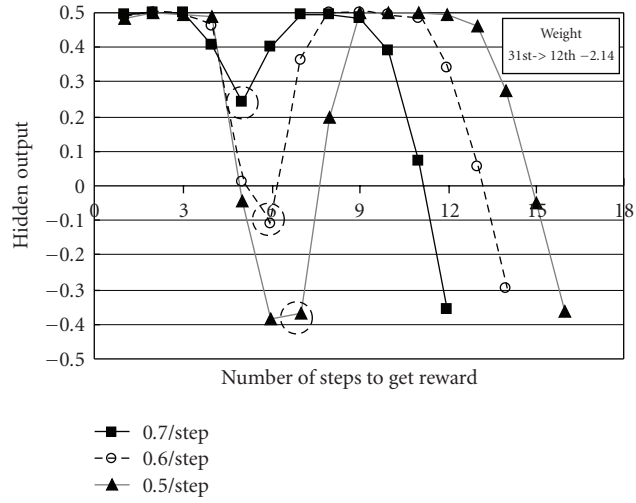


Figure 11: Change in output of the 31st hidden neuron for the three velocities. Broken circles indicate the timing just before the object disappears. Important connection weight value from this neuron is given in a box. In this case, the connection weight from the 31st neuron to the 12th neuron is −2.14.

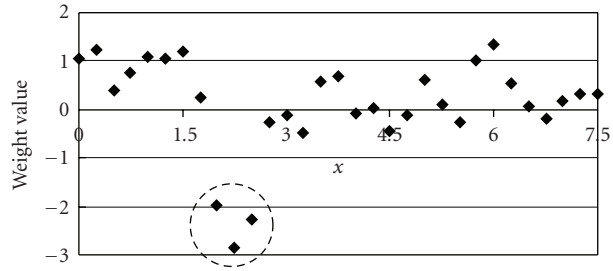


Figure 12: Connection weights to the 31st neuron from the input neurons that are responsible for the x location of the object. Lateral axis indicates the most responsible point of each input neuron. For example, the connection weight from the input neuron that takes the maximum value at $x = 2.25$ is approximately −2.8.

As shown in Figure 11, the output of the 31st hidden neuron decreases at around $2.0 < x < 2.5$, that is, a little before the area where the agent may not see the object. The broken circle is set on the timing just before the object disappears at $x = 3.0$. Figure 12 shows the connection weight of the 31st neuron from the input neurons that are responsible for the x location of the object. It can be seen that the 31st neuron has a large negative connection weight from three input neurons, each of which responds when the object exists at around $x = 2.0$, 2.25, or 2.5. This suggests that the negative connection weights decrease the output of this neuron.

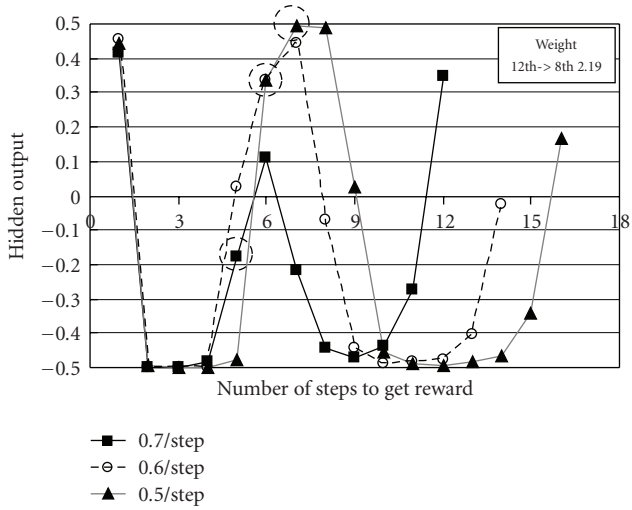


Figure 13: Output change of the 12th hidden neuron for the three velocities. Broken circles indicate the timing just before the object disappears.

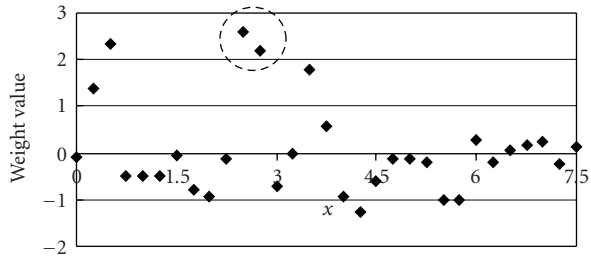


Figure 14: Connection weights to the 12th neuron from the input neurons responsible for the x location of the object. Lateral axis indicates the most responsible point of each input neuron.

As shown in Figure 13, the output of the 12th hidden neuron increases just before the object comes into the area where it may be invisible. The timing when the output of 12th neuron increases is slightly later than that of the decrease in output of the 31st neuron. The 12th neuron is connected from the 31st neuron with a large negative weight. Figure 14 shows the connection weights of the 12th hidden neuron from the input neurons contributing to the representation of the x coordinate of the object. This neuron has a large positive connection weight from the input neurons, which respond just before the object enters the area where it is possibly invisible. These two types of connections, from the 31st neuron and from some inputs, might increase the output of the 12th neuron.

When the pattern of increase in the output of the 12th hidden neuron is observed, it is clear that the output increases as the velocity of the object in the x direction decreases. When the x velocity of the object is smaller, it stays in the specific area for a longer period before entering the area where it might be invisible. This suggests that both the 12th and 31st neurons play a role in detecting the x velocity of the object, and the 12th neuron represents the velocity of the object just before it enters the area where it might

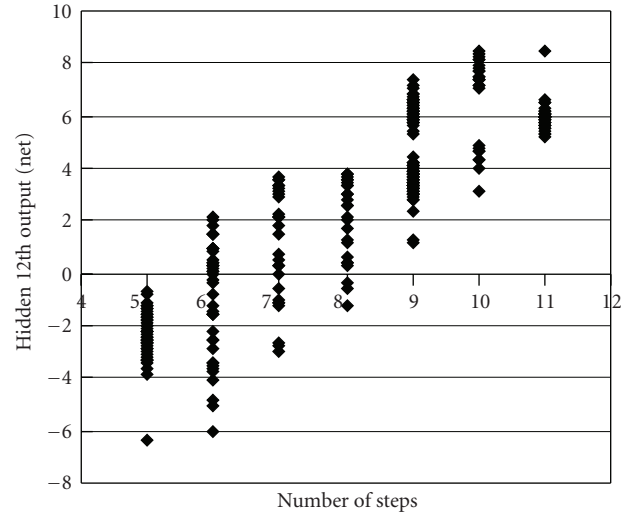


Figure 15: Correlation between the number of steps until catching the object in the invisibility area and the output of the 12th hidden neuron. The output varies according to the direction of motion and velocity of the object, although the number of steps until the catch is the same.

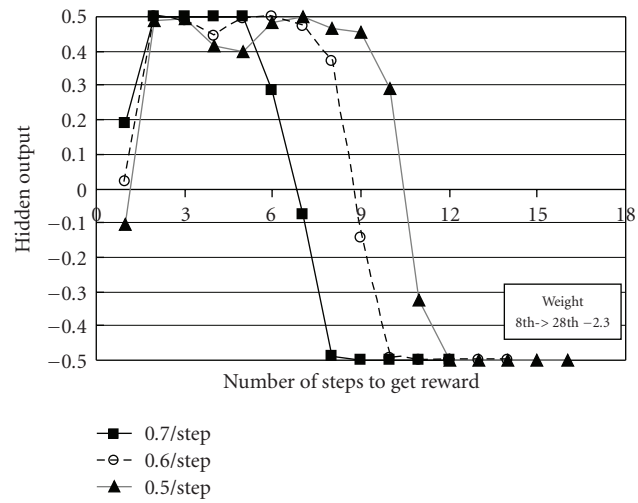


Figure 16: Output change of the 8th hidden neuron for the three velocities.

be invisible. The 12th neuron must represent the predicted catch timing and contribute toward catching the object at the appropriate timing even though the invisible area is wide.

Figure 15 shows the relation between the output of the 12th neuron just before the object passes the line of $x = 3.0$ and the number of steps from $x = 3.0$ to the catching of the object. Although they have some correlation, there is no one-to-one correspondence. As mentioned previously, the representation is distributed and one neuron represents several pieces of information simultaneously. In this case, this neuron represents not only the predicted catch timing but also some other information.

This 12th neuron has a large positive connection weight to the 8th neuron, whose output is shown in Figure 16.

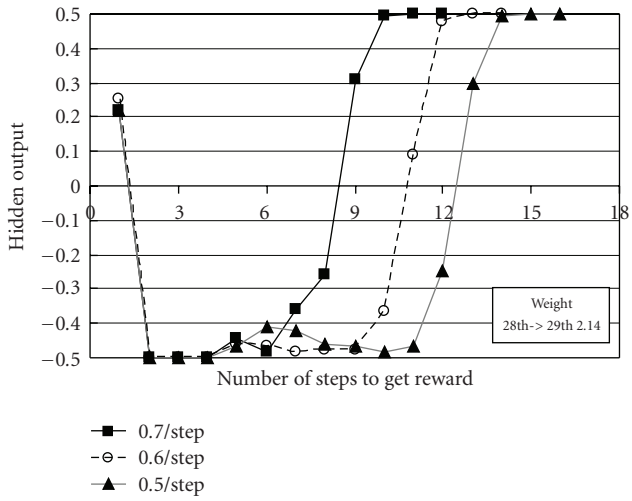


Figure 17: Output change of the 28th hidden neuron for the three velocities.

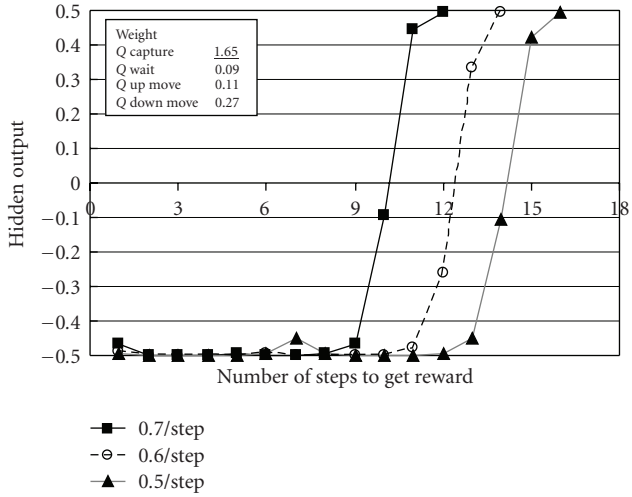


Figure 18: Output change of the 29th hidden neuron for the three velocities

Therefore, the delay of the timing of the decrease in the output of the 12th neuron causes a delay in the timing when the 8th neuron's output decreases. It is interesting that the difference of hidden output due to the object velocity in the 31st and 12th hidden neurons causes the delay of the signal in the 8th hidden neuron.

Next, the 28th neuron (Figure 17), which has a negative connection from the 8th neuron, responds later. Furthermore, the 28th neuron has a positive connection to the 29th neuron; the response of the 29th neuron (Figure 18) follows the response of the 28th neuron. The 29th neuron has a large positive connection to the output neuron, which represents the Q-value of the catch action whose response is shown in Figure 6. Thus, the response of the 12th neuron is relayed through some hidden neurons to the output neuron of the Q-value for the catch action. This relay of hidden neurons realizes the sudden increase in the Q-value for the catch action, as shown in Figure 6.

4.3. Consideration. The important point in this study is that the authors have not provided any knowledge in advance about the following items; the agent has learned them autonomously through reinforcement learning alone by reward and punishment.

(1) The velocity in the x direction of the object is useful for performing the catch action at the appropriate timing.

(2) The velocity in the x direction of the object can be detected by the inputs that respond to the existence of the object around a specific x coordinate.

(3) The way in which the detected velocity information can be related to the catch timing: especially, the detected velocity value is transformed to the delay of the signal.

(4) The information can be conveyed through a relay of hidden neurons.

Actually, more information than that discussed is considered in the recurrent network, but due to the parallelism of the processing system, it is difficult to understand its exact mechanism. Thus, it might also be difficult to understand the exact mechanism of the human brain.

5. Conclusion

In this paper, the authors proposed that the prediction function can emerge through reinforcement learning alone, using a recurrent neural network. This prediction function includes not only the way of predicting the target information but also the extraction of prediction target among many pieces of information available and the prediction of an appropriate timing. It was shown that through learning, the prediction function emerged—an agent could achieve a task in which prediction was necessary. Furthermore, the recurrent neural network extracts the necessary information for prediction from many input signals, relays the predicted information among some hidden neurons, and finally, enables the catching of the object at an appropriate timing. However, since the processing system is parallel, one neuron does not represent one piece of information explicitly, thus making it difficult to understand the processing of the network.

Acknowledgment

This research was supported by JSPS Grant-in-Aid for Scientific Research no. 19300070.

References

[1] J. Schmidhuber, "Temporal-difference-driven learning in recurrent networks," in *Parallel Processing in Neural Systems and Computers*, R. Eckmiller, G. Hartmann, and G. Hauske, Eds., pp. 209–212, North-Holland, Amsterdam, The Netherlands, 1990.

[2] J. Schmidhuber, "Reinforcement learning in Markovian and non-Markovian environments," in *Advances in Neural Information Processing Systems 3, (NIPS '91)*, D. S. Lippman, J. E. Moody, and D. S. Touretzky, Eds., pp. 500–506, Morgan Kaufmann, Denver, Colo, USA, 1991.

[3] L.-J. Lin and T. M. Mitchell, "Reinforcement learning with hidden states, from animals to animats 2," in *Proceedings of the 2nd International Conference on Simulation of Adaptive Behavior*, pp. 271–280, MIT Press, Honolulu, Hawai, USA, 1993.

[4] J. Tani, "Learning to generate articulated behavior through the bottom-up and the top-down interaction processes," *Neural Networks*, vol. 16, no. 1, pp. 11–23, 2003.

[5] M. L. Littman, R. S. Sutton, and S. Singh, "Predictive representations of state," in *Advances in Neural Information Processing Systems*, vol. 14, pp. 1555–1561, MIT Press, 2002.

[6] R. S. Sutton and B. Tanner, "Temporal-difference networks," in *Advances in Neural Information Processing Systems*, vol. 17, pp. 1377–1384, MIT Press, 2005.

[7] J. Schmidhuber, "A possibility for implementing curiosity and boredom in model-building neural controllers," in *Proceedings of the 1st International Conference on Simulation of Adaptive Behavior on From Animals to Animats*, J. A. Meyer and S. W. Wilson, Eds., pp. 222–227, MIT Press/Bradford Books, Paris, France, 1993.

[8] M. B. Ring, "CHILD: a first step towards continual learning," *Machine Learning*, vol. 28, no. 1, pp. 77–105, 1997.

[9] J. Schmidhuber, "Exploring the predictable," in *Advances in Evolutionary Computing*, S. Ghoshr and S. Tsutsui, Eds., pp. 579–612, Springer, London, UK, 2002.

[10] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 265–286, 2007.

[11] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, no. 1–3, pp. 139–159, 1991.

[12] P. McCracken, M. Bowling, and S. Tsutsui, "Online discovery and learning of predictive state representations," in *Advances in Neural Information Processing Systems 18*, pp. 875–882, 2006.

[13] K. Shibata and T. Kawano, "Acquisition of flexible image recognition by coupling of reinforcement learning and a neural network," *SICE Journal of Control, Measurement, and System Integration*, vol. 2, no. 2, pp. 122–129, 2009.

[14] A. Onat, H. Kita, and Y. Nishikawa, "Q-Learning with recurrent neural networks as a controller for the inverted Pendulum problem," in *Proceedings of the 5th International Conference on Neural Information Processing (ICONIP '98)*, pp. 837–840, Kitakyushu, Japan, October 1998.

[15] D. Aberdeen and J. Baxter, "Scaling internal-state policy-gradient methods for POMDPs," in *Proceedings of the International Conference on Machine Learning*, pp. 3–10, Las Vegas, Nev, USA, 2002.

[16] B. Bakker, V. Zhumatiy, G. Gruener, and J. Schmidhuber, "A robot that reinforcement-learns to identify and memorize important previous observations," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '03)*, pp. 430–435, Las Vegas, Nev, USA, 2003.

[17] H. Utsunomiya and K. Shibata, "Contextual behaviors and internal representations acquired by reinforcement learning with a recurrent neural network in a continuous state and action space task," in *Advances in Neural Information Processing Systems*, vol. 5507 of *Lecture Notes in Computer Science*, pp. 970–978, 2009.

[18] C. J. C. H. Watkins, "Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[19] D. Rumelhart, G. Hinton, and R.. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, D. Rumelhart, J. McClelland, and R. Williams, Eds., vol. 1, pp. 318–362, MIT Press, Cambridge, Mass, USA, 1986.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 217867, 15 pages
doi:10.1155/2010/217867

Research Article

Iterative Learning without Reinforcement or Reward for Multijoint Movements: A Revisit of Bernstein's DOF Problem on Dexterity

Suguru Arimoto,[1,2] Masahiro Sekimoto,[1] and Kenji Tahara[3]

[1] *Research Organization of Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan*
[2] *RIKEN-TRI Collaboration Center for Human-Interactive Robot Research, Nagoya, Aichi 463-0003, Japan*
[3] *Organization for the Promotion of Advanced Research, Kyushu University, Fukuoka 819-0395, Japan*

Correspondence should be addressed to Suguru Arimoto, arimoto@fc.ritsumei.ac.jp

Received 5 November 2009; Accepted 17 May 2010

Academic Editor: Noriyasu Homma

A robot designed to mimic a human becomes kinematically redundant and its total degrees of freedom becomes larger than the number of physical variables required for describing a given task. Kinematic redundancy may contribute to enhancement of dexterity and versatility but it incurs a problem of ill-posedness of inverse kinematics from the task space to the joint space. This ill-posedness was originally found by Bernstein, who tried to unveil the secret of the central nervous system and how nicely it coordinates a skeletomotor system with many DOFs interacting in complex ways. In the history of robotics research, such ill-posedness has not yet been resolved directly but circumvented by introducing an artificial performance index and determining uniquely an inverse kinematics solution by minimization. This paper tackles such Bernstein's problem and proposes a new method for resolving the ill-posedness in a natural way without invoking any artificial index. First, given a curve on a horizontal plane for a redundant robot arm whose endpoint is imposed to trace the curve, the existence of a unique ideal joint trajectory is proved. Second, such a uniquely determined motion can be acquired eventually as a joint control signal through iterative learning without reinforcement or reward.

1. Introduction

Almost a quarter century ago, "robotics" was defined by Professor Brady at the first International Conference of Robotics Research [1] as "the intelligent connection of perception to action." After a great deal of researches on developments of industrial robots and their applications, a variety of research projects on "humanoid" have attracted many roboticists during the past decade and nowadays robots that can walk with a bipedal mode are not peculiar. Nevertheless, the present state of the art of humanoid still lacks dexterity in fulfillment of ordinary tasks that human encounter in their everyday life. More than a half century ago preceding the birth of "humanoid," Bernstein [2, 3] noted that dexterity of human body movements resides in involvement of surplus degrees of freedom of limb joints but this incurs the ill-posedness of inverse kinematics. This was introduced to the robotics community through the famous textbook [4] in page 303 in such a statement as

"*The study of human biological motor control mechanisms led the Russian psychologist Bernstein to question how the brain could control a system with so many different degrees of freedom interacting in such a complex fashion. Many of these same complexities are also present in robotic systems and limit our ability to use multifingered hands and other robotic systems to their full advantage.*"

Actually, this was originally quoted from Hinton's article [5] in which he summarized what Bernstein challenged in the following way.

(a) What can we infer about the code that the brain uses to communicate with the periphery, and what does that tell us about how the computation is organized?, (b) If the brain knew just what movements it wanted the body to make, could it figure out what to tell the muscles in order to make

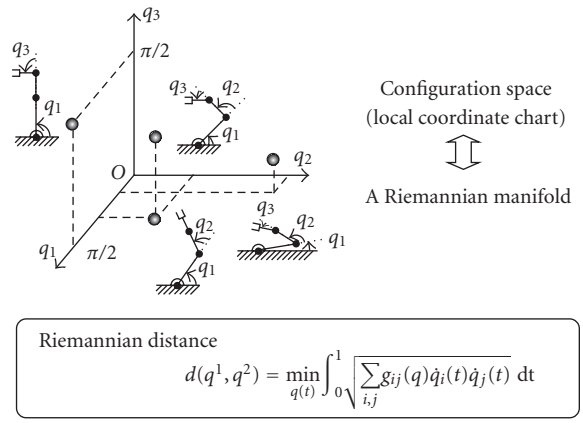


FIGURE 1: The set of all possible postures of a planar robot arm with three joints can be regarded as a Riemannian manifold with Riemannian metric $g_{ij}(q)$ that constitutes the inertia matrix.

it happen? (c) How is it possible to coordinate a system with so many degrees of freedom that interact in such complex ways? (d) How does the brain make sensible choices among the myriad possibilities for movement that the body offers?

This paper discusses what are the fundamentals of biomimetic control by focusing Bernstein's DOF problem and shows one way for solving such difficult problems from the standpoint of Riemannian geometry. It is shown in the case of multijoint reaching with DOF redundancy that, given a starting posture in the robot configuration space (or a Riemannian manifold as a set of all postures) and a target endpoint in the task space, there exists a unique orbit of joint motion, provided that the gravity term is compensated in a feedforward manner. Then, it is shown that such an ideal joint motion can be acquired through the iterative learning control without introducing any kind of performance index or reinforcement. In the second illustrative example, a handwriting motion with DOF redundancy is analyzed from the viewpoint of Riemannian geometry under the situation that writing with a ball pencil is imposed to trace an arbitrary smooth curve of C^∞-class on an arbitrary smooth surface in the three-dimensional Euclidean space. Even in this case there exists a unique joint motion in the base Riemannian manifold with DOF redundancy and it can be acquired through repeated exercises of handwriting motion, that is, an ILC scheme without introducing any artificial performance index. In conclusion, an ideal multijoint motion can be acquired through repeated exercises of motion even under the existence of redundancy in DOF, irrelevantly to any kind of reinforcement with the aid of some sort of reward [6, 7].

2. Riemannian Manifold and Euler's Equation

It is widely known among roboticists that kinematics and planning of multijoint robots are treataed in the configuration space regarded as an n-dimensional numerical space $\mathbf{R}^n$ [8, 9]. On the other hand, Arnold [10] pointed out the importance of Riemannian geometry in the analysis of mechanical systems and shown that the dynamics of motion of a double pendulum can be described by an orbit on a two-dimensional torus T^2 that is regarded as $T^2 = S^1 \times S^1$, where S^1 denotes a unit circle. In line with this notion, an n-DOF robot arm can be treated on an n-dimensional Rimeannian manifold like an n-dimensional torus T^n, and the stability problems of PD feedback with damping shaping [11] were retreated in a Riemannian-geometric manner [12, 13]. More recently, the author and his group showed that, given a robot arm, the set of all possible postures can be regarded as a Riemannian manifold with the Riemannian metric that constitutes the inertia matrix [14, 15] (see Figure 1). Thus, an orbit of motion as a geodesic solution to the Euler equation can be regarded as an inertia-induced motion without affection of damping and gravity forces [16].

It is well known as in a text book that motion of a robot manipulator as a serially connected rigid-body system is governed by an Euler-Lagrange equation shown in the form (see [17])

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q,\dot{q})\right\}\dot{q} + g(q) = u, \tag{1}$$

where $q = (q_1, \ldots, q_n)^{\mathrm{T}}$ denotes the vector of joint angles, $G(q) = (g_{ij})$ does the $n \times n$ inertia matrix, u a control toque vector, $g(q) = \partial P(q)/\partial q$ with a scalar function $P(q)$ called the potential, and $S(q, \dot{q})$ a skew-symmetric matrix $S = (S_{ij})$ defined as

$$S_{ij} = \frac{1}{2}\left\{\frac{\partial}{\partial q_j}\left(\sum_{k=1}^{n}\dot{q}_k g_{ik}\right) - \frac{\partial}{\partial q_i}\left(\sum_{k=1}^{n}\dot{q}_k g_{jk}\right)\right\}. \tag{2}$$

If we consider a control torque that can exactly compensate the gravity term, that is,

$$u = g(q), \tag{3}$$

then substitution of (3) into (1) yields

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q,\dot{q})\right\}\dot{q} = 0, \tag{4}$$

which is considered to be an ideal equation of motion without affection of gravity and joint damping forces like a robot arm on an artificial satellite in space. It is pointed out in the recent papers [14, 16] that (4) is equivalently written in the form

$$g_{ik}(q)\ddot{q}_i + \Gamma_{ikj}(q)\dot{q}_j\dot{q}_i = 0, \quad k = 1, \ldots, n, \tag{5}$$

where Γ_{ikj} denotes the Christoffel's symbol of the second kind and the symbol of summation with respect to i and j in (5) is omitted by obeying the Einstein's rule in differential geometry [18, 19]. Equation (5) is also expressed equivalently in the form

$$\ddot{q}_k + \Gamma^k_{ij}\dot{q}_i\dot{q}_j = 0, \quad k = 1, \ldots, n, \tag{6}$$

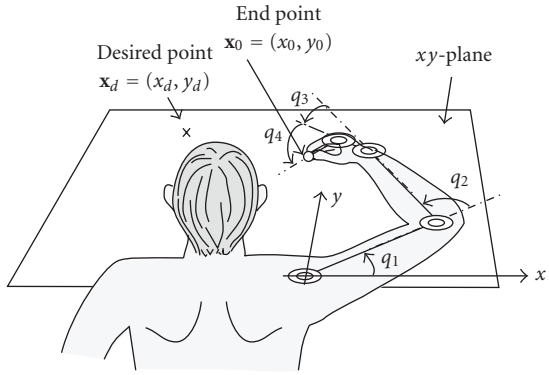


FIGURE 2: "Reaching" by means of a surplus DOF system of hand-arm dynamics.

which is called the Euler equation. In (5) and (6), Γ_{ikj} and Γ_{ij}^{k} (i.e., called the Christoffel's symbol of the first kind) are defined as follows:

$$\Gamma_{ikj} = \frac{1}{2}\left(\frac{\partial g_{jk}}{\partial q_i} + \frac{\partial g_{ik}}{\partial q_j} - \frac{\partial g_{ij}}{\partial q_k}\right),$$
$$\Gamma_{ij}^{k} = \frac{1}{2}\sum_{l=1}^{n} g^{lk}\left(\frac{\partial g_{jl}}{\partial q_i} + \frac{\partial g_{il}}{\partial q_j} - \frac{\partial g_{ij}}{\partial q_l}\right) = \frac{1}{2}\sum_{l=1}^{n} g^{lk}\Gamma_{ilj}, \tag{7}$$

where (g^{lk}) denotes the inverse of $G(=(g_{ij}))$. The equivalence relation of (5) to (4) is shown in the previous paper [15].

3. Multijoint Movements with DOF Redundancy

Let us now consider motion of a redundant planar robot arm whose endpoint is free to move in the horizontal plane as shown in Figure 2. The Lagrange equation of motion of the whole arm-hand system depicted in Figure 2 is expressed as

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q,\dot{q})\right\}\dot{q} = u, \tag{8}$$

where $q = (q_1, \ldots, q_4)^{\mathrm{T}}$ and each rotational axis of the four joints (shoulder, elbow, wrist, and index finger MP (metacarpophalangeal) joint) is in the direction perpendicular to the xy-plane. Given a robot arm posture q in the configuration space $\mathbf{R}^4$ or in the 4-dim. base manifold $\{M, g_{ij}\}$, the endpoint position $\mathbf{x}$ can be determined by the forward kinematics. A vector-valued function $\mathbf{x}(q) = (x(q), y(q))$ of C^{∞}-class. However, given an endpoint position $\mathbf{x}_d$ in $\mathbf{R}^4$, there arises an infinite number of inverses that realize $\mathbf{x}(q_d) = \mathbf{x}_d$ and thereby the problem for obtaining inverse kinematics from the 2-dimensional Euclidean space $\mathbf{E}^2$ to the 4-dimansional configuration space $\mathbf{R}^4$ becomes ill-posed.

A variety of ideas for solving such ill-posedness of inverse kinematics for redundant robotic systems with excess DOFs has been proposed in the area of robotics, based upon the use of the form

$$\dot{q}_d(t) = J^{+}(q(t))\dot{\mathbf{x}}_d + (I - J^{+}(q)J(q))\mathbf{v}, \tag{9}$$

where $\mathbf{v}$ should be computed so as to optimize a certain performance index related to joint position variables (for example, manipulability index [20], obstacle avoidance [21], etc.). In equation (9), $J(q)$ stands for the Jacobian matrix of $\mathbf{x}$ in q, that is, $J(q) = \partial \mathrm{x}/\partial q^{\mathrm{T}}$, and $J^{+}(q)$ denotes the pseudo-inverse of $J(q)$. The original idea of use of the pseudo-inverse $J^{+}(q)$ is due to Whitney [22]. Once a desired joint velocity $\dot{q}_d(t)$ is planned, it is claimed that the computed torque method can be applied for determining the control input that must generate the whole joint motion of the robot. This is called "inverse kinematics approach". Another idea of direct generation of a control signal called "inverse dynamics approach" is based upon a form of control input

$$u = G(q)J^{+}(q)\{\ddot{\mathbf{x}}_d - \dot{J}(q)\dot{q}\} + g(q,\dot{q}) + (I - J^{+}(q)J(q))\mathbf{v}, \tag{10}$$

where $\mathbf{v}$ is computed so as to optimize a certain performance index related to velocity variables (e.g., kinetic energy [23], torque [24], energy dissipation [25], etc.). In (10), $g(q,\dot{q})$ means compensation for the remaining nonlinear function including centrifugal and Coriolis forces and the gravity effect. In the physiological literature, main concerns are focussed on the question why human skilled multijoint reaching movements exhibit typical characteristics that (1) endpoint trajectory becomes a quasistraight line and less variable throughout repetitions, (2) velocity profiles of the endpoint velocity becomes bell-shaped, though (3) joint trajectories are rather variable trials-by-trials [26]. Then, a variety of cost functions for derivation of such properties of point-to-point reaching movements has been proposed, among which a quadratic function of endpoint jerk (rate of acceleration) was the first [27] and successively a cost function based on joint torques was introduced [28] for planning not only an endpoint trajectory but also joint trajectories. However, in the physiological literature, there is a dearth of papers that attempted to directly deal with reaching movements with redundant joints, though the importance of Bernstein's DOF problem [2] has been widely known among physiologists.

Differently from the traditional approaches, a simpler control method for multijoint reaching movements was proposed very recently in our previous papers [29–31] and shown to be effective in both cases of human and robotic arms with redundant DOFs. In those papers, only planar motions confined to a horizontal plane are treated and therefore the control signal is free from gravity with a simple form (see Figure 3)

$$u = -C\dot{q} - J^{\mathrm{T}}(q)k\Delta\mathbf{x}, \tag{11}$$

where C denotes a diagonal positive definite matrix as follows:

$$C = \mathrm{diag}(c_1, \ldots, c_n). \tag{12}$$

Notwithstanding this simpler form than (10), once damping factors C and single stiffness parameter k are chosen carefully, it is shown that it generates smooth reaching

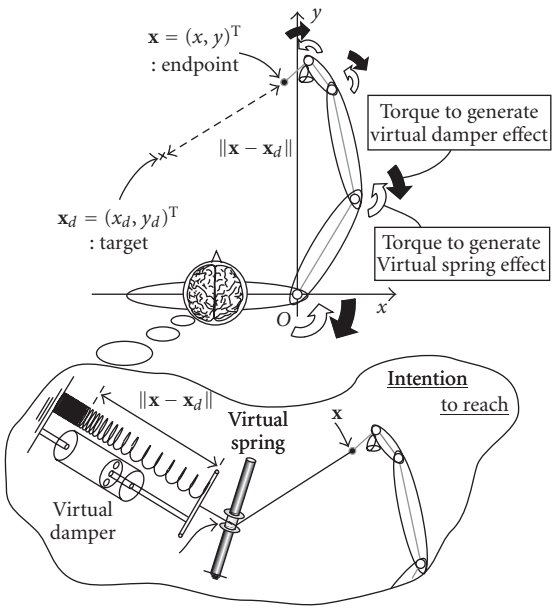


Figure 3: Virtual spring-damper hypothesis.

movements by realizing a quasistraight endpoint trajectory, bell-shaped velocity profiles, and double-peaked acceleration profiles typically seen in case of human skilled multijoint movements [26, 32]. Therefore, the method can get rid of undesirable fluctuations of the endpoint trajectory tracking as pointed out in a recent elaborate work [33]. Such fluctuations in task space tracking caused by using the computed torque under uncertainty in link parameters become more noticeable in cases of robots with redundant DOFs. In contrast, in the use of control defined in (11), there is no need of planning any desired endpoint trajectory. However, all these treatments are restricted to planar motions as well as in most of the previous papers on multijoint reaching movements. In addition, another disadvantage is that choice of damping factors recommended in [29] is not fit to the scale of coefficients of viscosity of human muscles [34, 35].

To reduce damping factors in general, another control method based upon "Virtual Spring-Damper Hypothesis" was suggested in [31], which in the case of planar motions without affection of the gravity is expressed by the form

$$\begin{aligned} u &= -C_0\dot{q} - J^{\mathrm{T}}(q)\left(\zeta\sqrt{k}\dot{\mathbf{x}} + k\Delta\mathbf{x}\right) \\ &= -C_0\dot{q} - \zeta\sqrt{k}J^{\mathrm{T}}(q)J(q)\dot{q} - J^{\mathrm{T}}(q)k\Delta\mathbf{x}, \end{aligned} \tag{13}$$

where C_0 is chosen as follows:

$$C_0 = \zeta_0\,\mathrm{diag}(c_1, \ldots, c_n), \tag{14}$$

together with positive constant $0 < \zeta_0 < 1.0$. The effectiveness of this control signal particularly in the case of middle-range reaching was demonstrated through computer simulation and its performance was compared with that of the control of (11). Theoretical verification of the effectiveness of this spring-damper hypothesis was also presented, on the basis of an energy conservation law like a Lyapunov-like relation

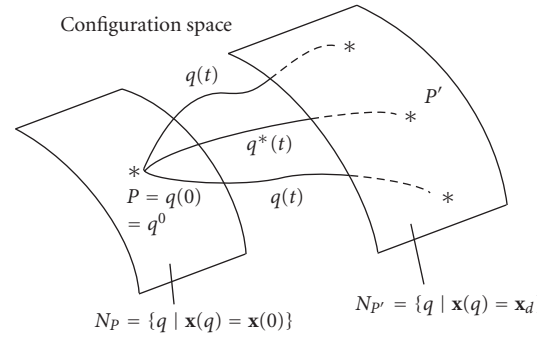


Figure 4: A set of all postures that have the same endpoint position $\mathbf{x}_d$ constitutes a two-dimensional Riemannian submanifold.

obtained by substituting (13) into (8) and taking the inner product of this resulted closed-loop dynamics and $\dot{q}$ as follows:

$$\begin{aligned} &G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G} + S + C_0\right\}\dot{q} + J^{\mathrm{T}}(q)\left\{\zeta\sqrt{k}\dot{\mathbf{x}} + k\Delta\mathbf{x}\right\} = 0, \\ &\frac{\mathrm{d}}{\mathrm{d}t}\left\{\frac{1}{2}\dot{q}^{\mathrm{T}}G(q)\dot{q} + \frac{k}{2}\|\Delta\mathbf{x}\|^2\right\} = -\dot{q}^{\mathrm{T}}C_0\dot{q} - \zeta\sqrt{k}\|\dot{\mathbf{x}}\|^2. \end{aligned} \tag{15}$$

These results suggest that skilled multijoint movements can be generated even in the case of robot arms with redundant DOFs without construction of any inverse dynamics through "error-feedback learning" as claimed in a physiological journal [36] for modifying Equilibrium-Point hypothesis [35], End-Point hypothesis [37], and Virtual Trajectory hypothesis [37].

4. Existence of Desired Joint-Motion

More recently in the paper [16], an interesting result is found that the endpoint trajectory of a solution to the closed-loop dynamics for a given starting posture resembles considerably the endpoint trajectory of a geodesic solution to the Euler equation of (5) or (6) starting from the same given posture to a certain different posture with the prescribed endpoint $\mathbf{x}_d$. The existence of such a geodesic solution to (6) is ascertained by considering a two-dimensional Riemannian submanifold that is defined by the set of all postures q satisfying $\mathbf{x}(q) = \mathbf{x}_d$, that is (see Figure 4),

$$N_{P'} = \{q \mid \mathbf{x}(q) = P'(= \mathbf{x}_d)\}. \tag{16}$$

Similarly, define another submanifold

$$N_P = \{q \mid \mathbf{x}(q) = P \ \ (= \mathbf{x}(0))\}. \tag{17}$$

For a given endpoint position $\mathbf{x}_d$ in $\mathbf{E}^2$, $N_{P'}$ constitutes a two-dimensional submanifold of the base manifold $\{M, g_{ij}\}$, that is called the equilibrium-point manifold or simply the EP-manifold in this paper. Denote by $q(t)$ any smooth orbit of motion of the robot starting at $t = a$ from the same posture $q(t = a) = q^0$ and reaching the submanifold $N_{P'}$ at $t = b$ so that it satisfies $\mathbf{x}(q(b)) = \mathbf{x}_d$. Then, consider the infimum

$$d(q^0, N_{P'}) = \inf_{q(t)} \int_a^b \sqrt{\sum_{i,j} g_{ij}(q)\dot{q}_i(t)\dot{q}_j(t)}\,\mathrm{d}t \tag{18}$$

over the set of all such possible orbits of robot motion connecting q^0 and $N_{P'}$. If $\mathbf{x}_d$ is not so distant from $\mathbf{x}(0)$ (= $\mathbf{x}(q^0)$), it is reasonable from the text books of Riemannian geometry [18, 19] that there exists a unique optimal orbit $q^*(t)$, $t \in [a, b]$, that minimizes the right-hand side of (18). Then, the quantity $d(q^0, N_{P'})$ is entitled to be called the Riemannian distance from q^0 to the submanifold $N_{P'}$ in the base manifold $\{M, g_{ij}\}$. It is well known that the optimal orbit $q^*(t)$ must satisfy the Euler equation (6) for $t \in [a, b]$ and, moreover, it must satisfy the following equation:

$$\{I_4 - J^+(q^*(t))J(q^*(t))\}\dot{q}^*(t) = 0, \tag{19}$$

where $J(q) = \partial \mathbf{x}(q)/\partial q^{\mathrm{T}}$, the Jacobian matrix of $\mathbf{x}(q)$ with respect to q. In other words, $\dot{q}^*(t)$ must belong to the image space of $J^{\mathrm{T}}(q^*(t))$ at any instant t in $[a, b]$. That is, $\dot{q}^*(t)$ does not have any component in the kernel space of J^{T}.

Consider now an endpoint trajectory tracking problem for a redundant multijoint arm of Figure 2 in the case that a desired endpoint motion is given as a curve $\mathbf{x}_d(t) : I = [0, T] \to \mathbf{E}^2$. The control task is to maneuver the robot to let its endpoint trace the given trajectory $\mathbf{x}_d(t)$ in $\mathbf{E}^2$ through a task space control signal

$$u = J^{\mathrm{T}}(q)v \tag{20}$$

provided that the robot dynamics is governed by the Lagrange equation

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q)\dot{q} + S(q, \dot{q})\right\}\dot{q} + C_0\dot{q} = J^{\mathrm{T}}(q)v, \tag{21}$$

where C_0 is a 4×4 positive definite damping coefficient matrix. We assume that the initial posture of the robot at $t = 0$ is given by $q(0) = q^0$ and motion of the robot starts from the still state, that is, $\dot{q}(0) = 0$. The first problem is to find an adequate control signal $v(t)$, $t \in [0, T]$, so that the solution to the Lagrange equation of (21) starting from $q(0) = q^0$ and $\dot{q}(0) = 0$ satisfies $\mathbf{x}(q(t)) = \mathbf{x}_d(t)$ for $t \in [0, T]$. In order to find a solution to this problem, we decompose any solution trajectory of joint velocity $\dot{q}(t)$ in such a way that

$$\dot{q} = (P, W)\begin{pmatrix}\dot{\mathbf{x}}\\ \dot{\eta}\end{pmatrix}, \tag{22}$$

where $\mathbf{x} = \mathbf{x}(q)$, $\dot{\mathbf{x}} = J(q)\dot{q}$, $\dot{\eta}$ is a 2×1 vector and P is the 4×2-matrix defined by

$$P = J^{\mathrm{T}}(q)\left(J(q)J^{\mathrm{T}}(q)\right)^{-1} = J^+(q) \tag{23}$$

and W is a 4×2-matrix whose column vectors w_1 and w_2 are orthogonal to $J^{\mathrm{T}}(q)$ (w_i belongs to the kernel space of $J^{\mathrm{T}}(q)$) and satisfy $\|w_1\| = \|w_2\| = 1$ and $w_1^{\mathrm{T}}w_2 = 0$. Then, if we define

$$\begin{aligned} &Q = (P, W), \qquad Q^{-1} = \begin{pmatrix}P^{\mathrm{T}}\\ W^{\mathrm{T}}\end{pmatrix} = Q^{\mathrm{T}},\\ &G_1 = Q^{\mathrm{T}}GQ, \qquad C_1 = Q^{\mathrm{T}}C_0Q,\\ &S_1 = Q^{\mathrm{T}}SQ - \frac{1}{2}\dot{Q}^{\mathrm{T}}GQ + \frac{1}{2}Q^{\mathrm{T}}G\dot{Q}, \end{aligned} \tag{24}$$

then, by substituting (22) into (21) and multiplying the resultant equation by the transpose of Q from the left, we have

$$G_1\begin{pmatrix}\ddot{\mathbf{x}}\\ \ddot{\eta}\end{pmatrix} + \frac{1}{2}\{\dot{G}_1 + S_1 + C_1\}\begin{pmatrix}\dot{\mathbf{x}}\\ \dot{\eta}\end{pmatrix} = \begin{pmatrix}\mathbf{v}\\ 0\end{pmatrix}. \tag{25}$$

Note that S_1 is again skew-symmetric. For convenience let us define

$$B_1 = \frac{1}{2}\dot{G}_1 + S_1 + C_1 \tag{26}$$

and decompose G_1 and B_1 in such a way that

$$G_1 = \begin{pmatrix}G_{11} & G_{12}\\ G_{12}^{\mathrm{T}} & G_{22}\end{pmatrix}, \qquad B_1 = \begin{pmatrix}B_{11} & B_{12}\\ B_{21} & B_{22}\end{pmatrix}, \tag{27}$$

where all G_{ij} and B_{ij} are of 2×2-matrix. Then, it follows from (25) that

$$G_{22}\ddot{\eta} + B_{22}\dot{\eta} = -G_{12}^{\mathrm{T}}\ddot{\mathbf{x}} - B_{21}\dot{\mathbf{x}}. \tag{28}$$

This equation means that if $\dot{\mathbf{x}}(t)$ is set as $\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_d(t)$ and $q_d(t)$ is also given then $\dot{\eta}(t)$ can be determined uniquely from solving the differential equation of (28) as an initial-value problem by setting $\dot{\eta}(0) = 0$. It should be remarked at this stage that the given curve $\mathbf{x}_d(t)$ is of C^∞-class described in terms of time parameter t in $\mathbf{E}^2$, that is, it is an $\mathbf{E}^2$-valued function of t with the initial value $\mathbf{x}_d(0) = \mathbf{x}(q^0)$, and it has the continuous time derivative $\dot{\mathbf{x}}_d(t)$ with the initial value $\dot{\mathbf{x}}_d(0) = 0$. Now, multiplying (28) by G_{22}^{-1} and accompanying this with (22) by setting $\dot{\mathbf{x}} = \dot{\mathbf{x}}_d$ and $\ddot{\mathbf{x}} = \ddot{\mathbf{x}}_d$, we have

$$\dot{q}_d = P(q_d)\dot{\mathbf{x}}_d + W(q_d)\dot{\eta}, \tag{29}$$

$$\begin{aligned} \ddot{\eta} = &-G_{22}^{-1}(q_d)\{B_{22}(q_d, \dot{q}_d)\dot{\eta} + G_{12}(q_d)\ddot{\mathbf{x}}_d\}\\ &+ G_{22}^{-1}(q_d)B_{21}(q_d, \dot{q}_d)\dot{\mathbf{x}}_d. \end{aligned} \tag{30}$$

This couple implies a set of six simultaneous differential equations of 1st order concerning six variables $\{q_d, \dot{\eta}\}$ though the right hand side of (30) contains $\dot{q}_d$ and hence it is an implicit function expression. Fortunately, it is possible to obtain an explicit expression of the six simultaneous differential equation by putting

$$\begin{aligned} B_{22} &= B_{22}(q_d, P\dot{\mathbf{x}}_d + W\dot{\eta}),\\ B_{21} &= B_{21}(q_d, P\dot{\mathbf{x}}_d + W\dot{\eta}). \end{aligned} \tag{31}$$

Hence, the right hand sides of (29) and (30) are nonlinear in q_d and $\dot{\eta}$, but they are Lipschitz continuous in q_d and $\dot{\eta}$ locally. Therefore, for given $\dot{\mathbf{x}}_d$ and $\ddot{\mathbf{x}}_d$ there exists a unique solution $\{q_d(t), \dot{\eta}(t)\}$ for an interval $t \in [0, a]$ with some $a > 0$ satisfying $q_d(0) = q^0$ and $\dot{\eta}(0) = 0$, where q^0 signifies an initial posture satisfying $\mathbf{x}(q^0) = \mathbf{x}_d(0)$. This fact was already discussed in our previous paper [38]. In this paper, we now prove the unique existence of the solution to the pair of 1st-order differential equations of (29) and (30) over the time interval $[0, T]$, provided that C_0 is not so small in comprison

with the scale of $G(q)$ and both the quantities of $\dot{\mathbf{x}}_d$ and $\ddot{\mathbf{x}}_d$ are within an adequate physical scale.

First, we analyze (30) or (28) by rewriting them in a more detailed manner as follows:

$$G_{22}\ddot{\eta} + \left\{\frac{1}{2}\dot{G}_{22} + S_{22}\right\}\dot{\eta} + C_{22}\dot{\eta} = -G_{12}^{\mathrm{T}}\ddot{\mathbf{x}}_d - \overline{B}_{21}\dot{\mathbf{x}}_d - C_{21}\dot{\mathbf{x}}_d, \tag{32}$$

where

$$G_{22} = W^{\mathrm{T}}GW, \qquad C_{22} = W^{\mathrm{T}}C_1W, \qquad G_{12}^{\mathrm{T}} = W^{\mathrm{T}}GP,$$
$$\overline{B}_{21} = \frac{1}{2}\dot{G}_{21} + S_{21}, \qquad C_{21} = W^{\mathrm{T}}C_1P \tag{33}$$

and S_{22} denote the submatrix (s_{ij}) of S_1 for $i, j = 3$ or 4, and $\overline{B}_{21}$ plus C_{21} constitutes B_{21}. Note that $\overline{B}_{21}$ is linear and homogeneous in $\dot{\eta}$ but G_{12} are irrelevant to $\dot{\eta}$. In this paper, we restrict our consideration to the case that overall movements of the robot is confined to a single chart of the base manifold in which at any posture in the chart the Jacobian matrix is nondegenerate and therefore there exists a positive constant σ_0 such that $J(q)J^{\mathrm{T}}(q) \geq \sigma_0 I_2$ inside the chart. Under this condition, we analyze the following relation obtained by taking the inner product of (32) and $\dot{\eta}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{2}\dot{\eta}^{\mathrm{T}}G_{22}\dot{\eta}\right) = -\dot{\eta}^{\mathrm{T}}C_{22}\dot{\eta} - \dot{\eta}^{\mathrm{T}}\overline{B}_{21}\dot{\mathbf{x}}_d - \dot{\eta}^{\mathrm{T}}\left\{G_{12}^{\mathrm{T}}\ddot{\mathbf{x}}_d + C_{21}\dot{\mathbf{x}}_d\right\}. \tag{34}$$

The second term of the right hand side is quadratic in $\dot{\eta}$ and therefore there exists a constant β_0 such that

$$\left\|\dot{\eta}^{\mathrm{T}}\overline{B}_{21}\dot{\mathbf{x}}_d\right\| \leq \beta_0\|\dot{\eta}\|^2 \tag{35}$$

and β_0 depend on the maximum magnitude of $\dot{\mathbf{x}}_d$ and $1/\sigma_0$. The third term of the right hand side is bounded from the above in the following way

$$\left\|-\dot{\eta}^{\mathrm{T}}\left\{G_{12}^{\mathrm{T}}\ddot{\mathbf{x}}_d + C_{21}\dot{\mathbf{x}}_d\right\}\right\| \leq \dot{\eta}^{\mathrm{T}}W^{\mathrm{T}}GW\dot{\eta} + \frac{1}{2}\ddot{\mathbf{x}}_dP^{\mathrm{T}}GP\ddot{\mathbf{x}}_d + \frac{1}{2}\dot{\mathbf{x}}_d^{\mathrm{T}}P^{\mathrm{T}}C_0G^{-1}C_0P\dot{\mathbf{x}}_d. \tag{36}$$

As discussed previously in [39], the damping matrix C_1 can be chosen to be of the order of $G^{1/2}$ and further so as to satisfy

$$C_0 > 2\beta_0 I_4, \quad C_0 > 2G. \tag{37}$$

Then, (34) is reduced to the inequality relation

$$\frac{\mathrm{d}}{\mathrm{d}t}\eta_d(t) \leq -\gamma_0\eta_d(t) + \xi_d(t), \tag{38}$$

where we put with some positive constant γ_0

$$\eta_d(t) = \frac{1}{2}\dot{\eta}^{\mathrm{T}}(t)G_{22}(t)\dot{\eta}^{\mathrm{T}}(t),$$
$$\xi_d(t) = \frac{1}{2}\ddot{\mathbf{x}}_d^{\mathrm{T}}(t)P^{\mathrm{T}}GP\ddot{\mathbf{x}}_d(t) + \frac{1}{2}\dot{\mathbf{x}}_d^{\mathrm{T}}(t)P^{\mathrm{T}}C_0G^{-1}C_0P\dot{\mathbf{x}}_d(t). \tag{39}$$

Clearly, since $\eta_d(0) = 0$, (38) implies

$$\eta_d(t) \leq \int_0^t e^{-\gamma_0(t-\tau)}\xi_d(\tau)\mathrm{d}\tau \tag{40}$$

which concludes that $\dot{\eta}(t)$ is uniformly bounded in $t \in [0, T]$.

Once $\dot{q}_d(t)$ and $\eta(t)$ are obtained for the given $\dot{\mathbf{x}}_d(t)$ and $\ddot{\mathbf{x}}_d(t)$, the desired input signal $\mathbf{v}_d(t)$ in the image space of $J(q_d)$ is obtained by setting $q(t) = q_d(t)$, $\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_d(t)$, $\ddot{\mathbf{x}}(t) = \ddot{\mathbf{x}}_d(t)$ in (25).

5. Iterative Learning Control in the Task Space

Given a desired trajectory of the endpoint in the task space $\mathbf{E}^2$ for a redundant robot arm, there exists a unique trajectory of robot motion in the joint space for a specified initial posture. In particular, it is shown in the previous section that there is uniquely a control signal in the task space that maneuvers the robot through the transpose of the Jacobian matrix to realize the endpoint tracking. However, such a control signal can not be obtained in any analytical form. Nevertheless, it is possible for us to acquire such a desired control signal by using a simple iterative learning control scheme, provided that the endpoint trajectory in the task space can be measured by visual sensing.

At the kth trial of iterative learning, the control signal for the dynamics of (21) is designed in the form

$$\nu_k = -J^{\mathrm{T}}(q_k)\left\{\kappa\Delta\mathbf{x}_k(t) + \zeta_1\sqrt{\kappa}\Delta\dot{\mathbf{x}}_k(t)\right\} + \nu_{k-1} - \Phi\Delta\dot{\mathbf{x}}_{k-1}(t), \tag{41}$$

where $J(q_k)$ means $\partial\mathbf{x}/\partial q$ at $\mathbf{x} = \mathbf{x}(q_k)$, and

$$\Delta\mathbf{x}_k(t) = \mathbf{x}(q_k(t)) - \mathbf{x}_d(t),$$
$$\Delta\dot{\mathbf{x}}_k(t) = \dot{\mathbf{x}}(q_k(t)) - \dot{\mathbf{x}}_d(t). \tag{42}$$

The first term of the right hand side of (41) signifies the inner task-space PD feedback, ν_{k-1} denotes the previous control signal at the $(k-1)$th trial, and Φ is an adequate positive definite constant matrix. At the first trial, usually we set $\nu_0(t) = 0$ for $t \in [0, T]$. At the second trial $\nu_1(t)$ must contain erroneous terms. Fortunately, without knowing the desired ideal control $\nu_d(t)$, it is possible to expect that $\Delta\mathbf{x}_i(t) \to 0$ and $\Delta\dot{\mathbf{x}}_k \to 0$ for $t \in [0, T]$ as $k \to \infty$. We give an illustrative example of numerical simulation conducted for the 4-DOF robot arm shown in Figure 2 with physical parameters given in Table 1. The values for length, mass, and inertia moment of the first link correspond to those of an upper arm of average human adult (male), and the values for the second link do to those of a lower arm. The third link corresponds to a human palm and the fourth an index finger. The desired task is to write a handwritten character "α" on the xy-plane. More explicitly, the endpoint trajectory is given by the equation

$$\mathbf{x}_d(t) = \begin{bmatrix}0.00\\0.30\end{bmatrix} + \begin{bmatrix}0.075\cos\omega(t)\\0.100\cos 1.5\omega(t)\end{bmatrix}, \tag{43}$$

Table 1: Physical parameters of the 4-DOF robot arm.

Link number	i	1	2	3	4
Length [m]	l_i	0.2800	0.2800	0.09500	0.09000
Center of mass [m]	l_{gi}	0.1400	0.1400	0.04750	0.04500
Cylinder radius [m]	r_i	0.04000	0.03500	N/A	0.009500
Cuboid height [m]	h_i	N/A	N/A	0.08500	N/A
Cuboid depth [m]	d_i	N/A	N/A	0.03000	N/A
Mass [kg]	m_i	1.407	1.078	0.2423	0.02552
Inertia moment [kg m^2]	I_{giz}	9.758×10^{-3}	7.370×10^{-3}	2.004×10^{-4}	1.780×10^{-5}

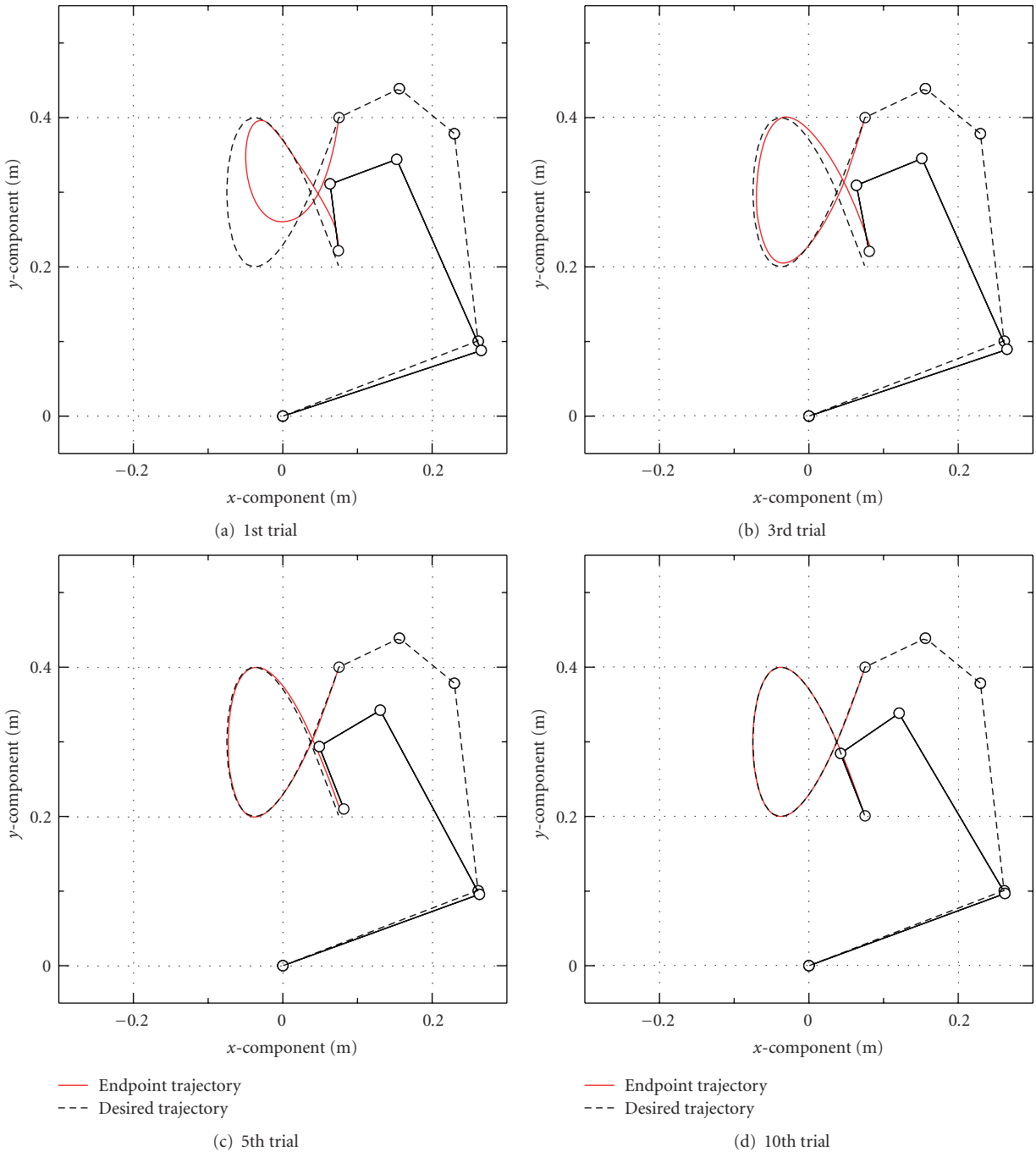


Figure 5: Endpoint trajectories and the initial and final postures of the arm.

Table 2: Initial values and gain settings in the case of the 4-DOF planar robot arm.

(a)

Terminal time	T	2.0 [s]
Initial posture	$q_1(0)$	21.02 [deg]
	$q_2(0)$	75.47 [deg]
	$q_3(0)$	44.20 [deg]
	$q_4(0)$	64.87 [deg]

(b)

Gains	k	10 [N/m]
	ζ_0	0.5 [−]
	ζ_1	2.5 [−]
	Φ	10.0 [−]
	c_1	0.862 [Nms]
	c_2	0.569 [Nms]
	c_3	0.129 [Nms]
	c_4	0.0356 [Nms]

where $T = 2.0$ [s] and

$$\omega(t) = 2.0\pi\left\{-15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5 + 10\left(\frac{t}{T}\right)^3\right\}. \quad (44)$$

The initial posture of the arm is given in Table 2. Based on these data, the system of differential equations

$$\begin{aligned} G(q_k)\ddot{q}_k + \left\{\frac{1}{2}\dot{G}(q_k) + S(q_k, \dot{q}_k) + C_0\right\}\dot{q}_k \\ + J^{\mathrm{T}}(q_k)\{\kappa\Delta\mathbf{x}_k + \zeta_1\sqrt{\kappa}\Delta\dot{\mathbf{x}}_k\} = J^{\mathrm{T}}(q_k)\mathbf{v}_k \end{aligned} \quad (45)$$

is numerically solved by using the Runge-Kutta method. In Figure 5 we show endpoint trajectories at 1st, 3rd, 5th, and 10th trials and their corresponding postures at $t = 0$ and $t = T$. On the other side, we are able to obtain the desired control signal numerically by numerically solving a couple of 1st-order differential equations of (29) and (30). Based upon knowing physical data given in Table 1, the initial posture of $q(0)$ given in Table 2, and the specified endpoint trajectory of (43), we obtain the desired ideal control signal shown in Figure 6. It is quite interesting to know that, through simulations of the iterative learning, calculated control signals $v_k(t)$, $t \in [0, T]$, approach the desired one as the trial number k increases as shown in Figure 6. When $k = 10$, the trajectory of control signal $\mathbf{v}_{10}(t)$ almost coincides with the ideal one $\mathbf{v}_d(t)$, that uniquely exists just in the image space of the Jacobian matrix $J(q_d)$ with $\mathbf{x}(q_d) = \mathbf{x}_d$ for all $t \in [0, T]$.

It should be remarked that the desired endpoint trajectory $\mathbf{x}(t)$ in the case of multijoint reaching for the robot arm of Figure 2 is obtained by solving the Euler equation of (4) or (5) as a two-point boundary-value problem when the initial posture $q(0) = q^0$ at $t = 0$ is given and the terminal condition at $t = T$ is partially specified so as to move the endpoint of the arm to meet $\mathbf{x}(q) = \mathbf{x}_d$ at $t = T$ and pass it away. This is the problem to find a curved orbit of the endpoint connecting two given points $\mathbf{x}(q(0)) = \mathbf{x}^0$ and $\mathbf{x}(q(T)) = \mathbf{x}_d$ by selecting an adequate initial joint velocity $\dot{q}(0)$ that is nonzero. However, the orbit $\mathbf{x}(t)$ of the endpoint can be represented by a curve $c(s)$ on $\mathbf{E}^2$ with the aid of length parameters. In the case of middle-range reaching, the profile of this endpoint geodesic curve $c(s)$ quite resembles those of human-like multijoint reaching characterized typically by a quasistraight line movement of the endpoint starting from a fixed still state.

6. Extension to the Case of Existence of Effect of Gravity

Most of the previous results in Sections 4 and 5 can be extended to the case that robot dynamics is subject to the effect of gravity. In such a robot with redundancy in DOF, robot dynamics is expressed by the Lagrange equation:

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q, \dot{q}) + C_0\right\}\dot{q} + g(q) = J^{\mathrm{T}}(q)\mathbf{u}, \quad (46)$$

where $g(q)$ stands for the gravity term that can be regarded as a gradient vector of a potential function $U(q)$ with respect to q, that is, $g(q) = \partial U(q)/\partial q$. Let us denote again the endpoint position by $\mathbf{x}(q)$ in the m-dimensional Euclidean space $\mathbf{E}^m$. Then, we split $g(q)$ into

$$\begin{aligned} g(q) &= J^+(q)J(q)g(q) + (I_n - J^+(q)J(q))g(q) \\ &= g_1(q) + g_2(q). \end{aligned} \quad (47)$$

That is, $g_1(q)$ is a component of $g(q)$ in the image space of $J(q)$ and $g_2(q)$ is that of $g(q)$ in the kernel space of $J(q)$, that is orthogonally complement to the image space. In accordance with the split of the term $g(q)$, let us choose an $n \times (n - m)$-matrix $W(q) = (w_1, \ldots, w_{n-m})$ with n-dimensional unit column vectors w_i $(i = 1, \ldots, n - m)$ that are mutually orthogonal and satisfy $W(q)J^{\mathrm{T}}(q) = 0$ (i.e., $W(q)J^+(q) = 0$). Then, in a similar way to (22), we define

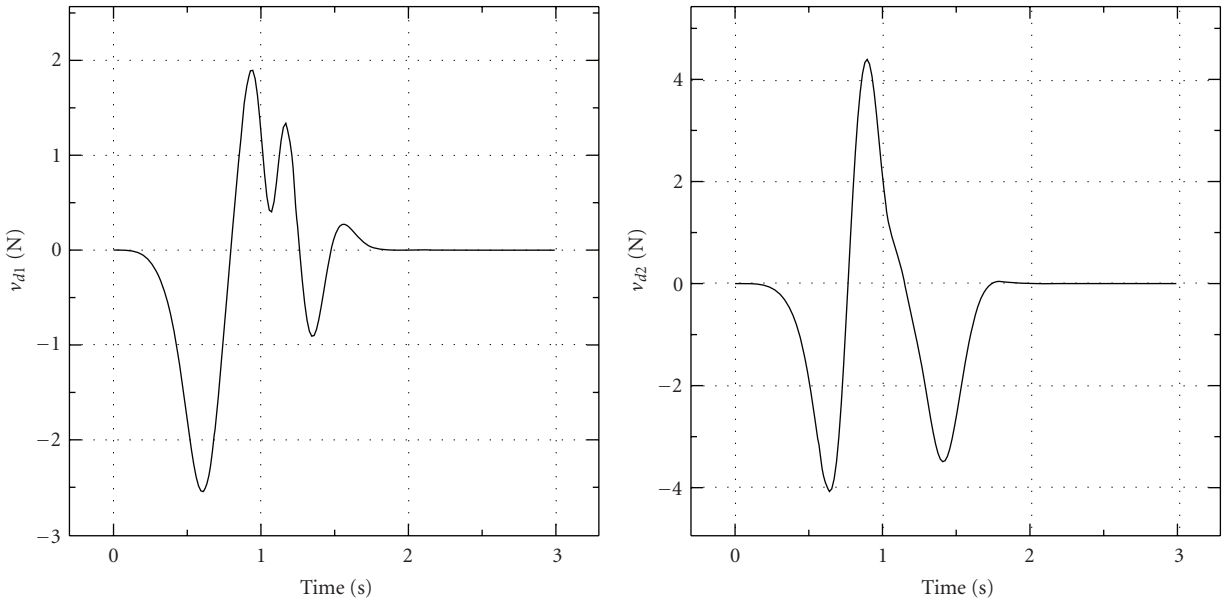
$$Q(q) = (J^+(q), W(q)) \quad (48)$$

which leads to

$$Q^{-1}(q) = \begin{pmatrix} J(q) \\ W^{\mathrm{T}}(q) \end{pmatrix}. \quad (49)$$

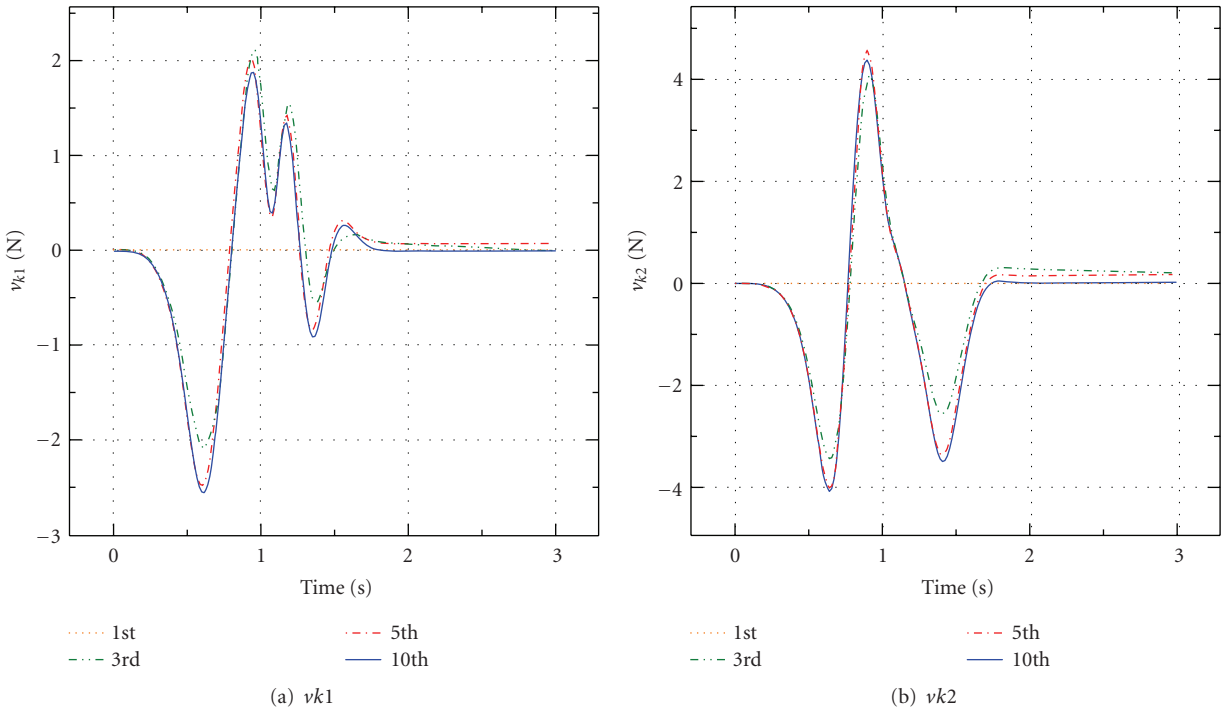
On the other hand, for a given desired endpoint trajectory $\mathbf{x}_d(t)$ for $t \in [0, T]$, we consider the control signal with the task space PD feedback

$$\mathbf{u} = -\{\kappa\Delta\mathbf{x} + \zeta_1\sqrt{\kappa}\Delta\dot{\mathbf{x}}\} + \mathbf{v}, \quad (50)$$



(a) $vd1$ (b) $vd2$

FIGURE 6: Transient responses of the ILC term $\mathbf{v}_d = (v_{d1}, v_{d2})^{\mathrm{T}}$.



(a) $vk1$ (b) $vk2$

FIGURE 7: Transient responses of the ILC term $\mathbf{v}_k = (v_{k1}, v_{k2})^{\mathrm{T}}$.

where $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_d$. Substituting (50) into (46) yields the closed-loop dynamics

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q,\dot{q}) + C_0\right\}\dot{q} + g_1(q) + g_2(q) + J^{\mathrm{T}}(q)\left\{\kappa\Delta\mathbf{x} + \zeta_1\sqrt{\kappa}\Delta\dot{\mathbf{x}}\right\} = J^{\mathrm{T}}(q)\mathbf{v}, \tag{51}$$

where $\mathbf{v}$ expresses a feedforward task space control signal. Then, consider the transform of $\dot{q}$ to $(\dot{\mathbf{x}}^{\mathrm{T}}, \dot{\eta}^{\mathrm{T}})^{\mathrm{T}}$ with an $(n - m)$-dimensional vector in such a form that

$$\dot{q} = (J^{+}(q), W(q))\begin{pmatrix}\dot{\mathbf{x}}\\ \dot{\eta}\end{pmatrix} = Q(q)\begin{pmatrix}\dot{\mathbf{x}}\\ \dot{\eta}\end{pmatrix}. \tag{52}$$

In order that the equation of (51) with a desired control signal $\mathbf{v}_d(t)$ has a unique solution $q_d(t)$ and $\dot{q}_d(t)$ so that its endpoint trajectory $\mathbf{x}(t)$ is coincident with $\mathbf{x}_d(t)$, and $\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_d(t)$, the following equations must be satisfied:

$$G_d\begin{pmatrix}\ddot{\mathbf{x}}_d\\ \ddot{\eta}\end{pmatrix} + \left\{\frac{1}{2}\dot{G}_d + S_d + C_d\right\}\begin{pmatrix}\dot{\mathbf{x}}_d\\ \dot{\eta}\end{pmatrix} + \begin{pmatrix}(J^+(q_d))^{\mathrm{T}} g_d\\ W^{\mathrm{T}}(q_d) g_d\end{pmatrix} = \begin{pmatrix}\mathbf{v}_d\\ 0\end{pmatrix}, \tag{53}$$

where $g_d = g(q_d)$ and

$$G_d = Q_d^{\mathrm{T}} G(q_d) Q_d, \qquad C_d = Q_d^{\mathrm{T}} C_0 Q_d,$$
$$S_d = Q_d^{\mathrm{T}} S Q_d - \frac{1}{2}\dot{Q}_d^{\mathrm{T}} G(q_d) Q_d + \frac{1}{2} Q_d^{\mathrm{T}} G(q_d)\dot{Q}_d \tag{54}$$

with $Q_d = Q(q_d)$. The latter $(n-m)$-dimensional component of (51) can be expressed by

$$G_{22}\ddot{\eta}_d + \left\{\frac{1}{2}\dot{G}_{22} + S_{22} + C_{22}\right\}\dot{\eta} + G_{12}^{\mathrm{T}}\ddot{\mathbf{x}}_d + B_{21}\dot{\mathbf{x}}_d + W_d^{\mathrm{T}} g_d = 0, \tag{55}$$

where G_d and $B_{\mathrm{d}} = (1/2)\dot{G}_d + S_d + C_d$ are decomposed into

$$G_d = \begin{pmatrix}G_{11} & G_{12}\\ G_{12}^{\mathrm{T}} & G_{22}\end{pmatrix}, \qquad B_d = \begin{pmatrix}B_{11} & B_{12}\\ B_{21} & B_{22}\end{pmatrix}, \tag{56}$$

and G_{22} and B_{22} are of $(n-m)\times(n-m)$-matrix, and G_{11} and B_{11} are of $m \times m$-matrix. The simultaneous differential equations (53) of 1st order together with

$$\dot{q}_d = J^+(q_d)\dot{\mathbf{x}}_d + W(q_d)\dot{\eta} \tag{57}$$

determine a unique solution $q_d(t)$ and $\dot{\eta}(t)$ for given $\mathbf{x}_d(t)$, $\dot{\mathbf{x}}_d(t)$, and $\ddot{\mathbf{x}}_d(t)$ with the initial conditions $q_d(0) = q^0$, $\mathbf{x}(q^0) = \mathbf{x}_d(0)$, and $\dot{\eta}(0) = 0$.

7. ILC for Handwriting

All the considerations in the previous sections can be extended to the case of a handwriting robot whose last link is a ball-point pen constrained on a hypersurface $\varphi(\mathbf{x}) = 0$, where $\mathbf{x} = (x, y, z)^{\mathrm{T}}$ in $\mathbf{E}^3$ and $\varphi(\mathbf{x})$ is a scalar function of C^∞-class (see Figure 8). First, we consider a pure mathematical problem of finding a geodesic curve by ignoring the effect of gravity and any joint damping. In this case, let us consider an open connected area $\overline{S}$ on the hypersurface as shown in Figure 8, on which any point $P(= \mathbf{x})$ satisfies the equality $\varphi(\mathbf{x}) = 0$. Then, we denote by F a local coordinate chart defined by $F = \{q \in (M, g_{ij}) \mid \mathbf{x}(q) \in \overline{S}\}$ and assume that F is connected and at any $q \in F$ the Jacobian matrix $J(\mathbf{x}(q)) = \partial\mathbf{x}(q)/\partial q^{\mathrm{T}}$ is nondegenerate. Then, given a point P on $\overline{S}$ with the cartesian coordinates $\mathbf{x}_P$, the set of all q such that $\mathbf{x}(q) = \mathbf{x}_P$ and $q \in F$ constitutes a single-dimensional submanifold N_P (see Figure 9). Hence, for another given point P' on

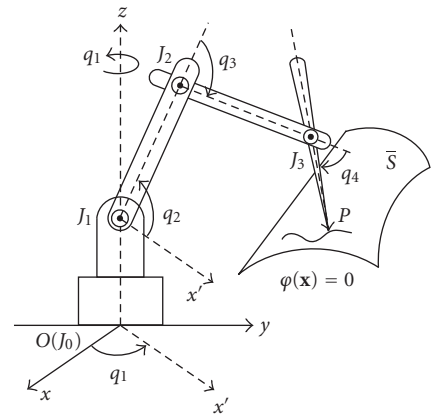


Figure 8: A handwriting robot with four DOFs whose endpoint $P(= (x, y, z))$ is constrained on a hypersurface $\varphi(\mathbf{x}) = 0$, where $\varphi(\mathbf{x})$ is a scalar function of C^∞-class.

$\overline{S}$, it is possible to consider an orbit in the submanifold F (equivalently, $\phi(F)$ in the configuration space) starting from q^0, lying on $N(S)$, and reaching some point lying on $N_{P'}$ that is another single-dimensional submanifold defined by the set of all q satisfying $\mathbf{x}(q) = \mathbf{x}_{P'}$ and $q \in F$. Thus, it is reasonable to suppose that there exists an optimal orbit $q^*(t)$ that gives minimization of the Riemannian distance from q^0 to $N_{P'}$ such that

$$d(q^0, N_{P'}) = \inf_{q(t)} \int_0^1 \sqrt{g_{ij}(q(t))\dot{q}_i(t)\dot{q}_j(t)}\,\mathrm{d}t = \int_0^1 \sqrt{g_{ij}(c(t))\dot{c}_i(t)\dot{c}_j(t)}\,\mathrm{d}t, \tag{58}$$

where the infimum is taken over all the orbits lying on $N(S)$ (see Figure 9), starting from q^0 and reaching $N_{P'}$, where $q^*(t)$ is rewritten by $c(t)$. It is reasonable to conclude that the optimal orbit $c(t)$ in (58) is a solution to the Euler equation under the constraint $\varphi(\mathbf{x}(c)) = 0$:

$$G(c(t)) + \left\{\frac{1}{2}\dot{G}(c) + S(c, \dot{c})\right\}\dot{c}(t) = -\lambda J_c^{\mathrm{T}}(\mathbf{x}(c(t)))\frac{\partial\varphi}{\partial\mathbf{x}^{\mathrm{T}}}, \tag{59}$$

where λ denotes a Lagrange multiplier. In other words, the path of $c(t)$ from q^0 to some point on $N_{P'}$ can be called the geodesic on the Riemannian submanifold $N(S)$ induced naturally from the constraint $\varphi(\mathbf{x}(q)) = 0$.

Next, consider the full dynamics of the handwriting robot with 4 DOFs shown in Figure 8 by taking into account the effect of gravity forces and damping torques at joints (see [38]). The dynamics is described by

$$G(q)\ddot{q} + \left\{\frac{1}{2}\dot{G}(q) + S(q, \dot{q})\right\}\dot{q} + C_0\dot{q} + g(q) = -\lambda J^{\mathrm{T}}(q)\frac{\partial\varphi(\mathbf{x})}{\partial\mathbf{x}^{\mathrm{T}}} + u, \tag{60}$$

where $g(q) = \partial U(q)/\partial q$, $U(q)$ denotes the gravity potential, C_0 a positive definite damping matrix, and $J(q) = \partial\mathbf{x}(q)/\partial q^{\mathrm{T}}$. It should be remarked that $\partial\varphi/\partial\mathbf{x}$ stands for a vector that

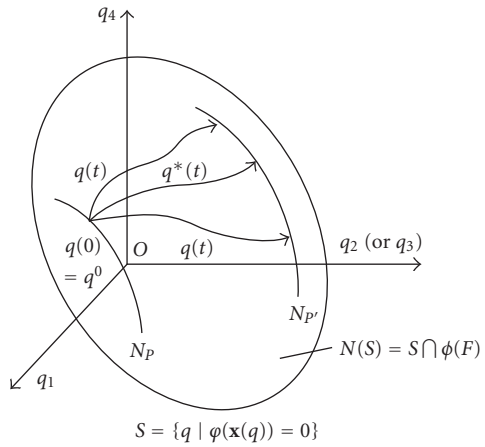


FIGURE 9: A geodesic curve starting from q^0 lying on N_P and reaching some point on $N_{P'}$. Here, ϕ is a homomorphism mapping F in M to $\phi(F)$ in R^4.

originates at the position $\mathbf{x}(q)$ in $\mathbf{E}^3$ and is normal to the surface $\varphi(\mathbf{x}) = 0$ which the ball pen contacts with. Now we introduce a length parameter s in $\mathbf{E}^3$ by the quantity

$$\begin{aligned} s(t) &= \int_0^t \|\dot{\mathbf{x}}(\tau)\| \mathrm{d}\tau \\ &= \int_0^t \left\| \dot{q}^{\mathrm{T}}(\tau) J(q(\tau)) J^{\mathrm{T}}(q(\tau)) \dot{q}(\tau) \right\| \mathrm{d}\tau \end{aligned} \tag{61}$$

and define the unit normal at the contact point as follows:

$$\mathbf{n}(s) = \frac{\partial \varphi}{\partial \mathbf{x}} \left\| \frac{\partial \varphi}{\partial \mathbf{x}} \right\|^{-1}. \tag{62}$$

Then, by rewriting $f = \lambda \|\partial\varphi/\partial\mathbf{x}\|$, (60) can be rewritten into

$$\begin{aligned} G(q)\ddot{q} + \left\{ \frac{1}{2}\dot{G}(q) + S(q,\dot{q}) \right\} \dot{q} + C_0 \dot{q} + g(q) \\ = -f J^{\mathrm{T}}(q)\mathbf{n}(s) + u. \end{aligned} \tag{63}$$

Note that the inner product of $\dot{q}$ and the right-hand side of (63) vanish. In other words, $\dot{\mathbf{x}}(= J(q)\dot{q})$ is orthogonal to $\mathbf{n}(s)$ at the contact point in $\mathbf{E}^3$. Under the assumption that both the endpoint position $\mathbf{x}(t)$ and the velocity $\dot{\mathbf{x}}(t)$ are measured in real time by visual sensing and the Jacobian matrix $J(q(t))$ is calculated from the measurement data of $\mathbf{x}(t)$ and $q(t)$ in real time, suppose that the control signal must be constructed through the Jacobian transpose in the form

$$u = J^{\mathrm{T}}(q)v. \tag{64}$$

For a given desired endpoint trajectory $\mathbf{x}_d(t)$, $t \in [0, T]$, together with $\dot{\mathbf{x}}_d(t)$ and $\ddot{\mathbf{x}}_d(t)$ and a given desired pressing force $f_d(t)$, we are concerned with the problem to find a desired control signal $v_d(t)$ of (64) that maneuvers the robot to make the endpoint of the last link (ball pen) trace $\mathbf{x}(q(t)) = \mathbf{x}_d(t)$ on the hypersurface with the pressing force f_d in the direction normal to the surface. To show the existence of a control signal in the image space of $J(q)$, we express it in a decomposed form such that

$$u = J^{\mathrm{T}}(q)\{v_n \mathbf{n}(s) + v_b \mathbf{b}(s) + v_e \mathbf{e}(s)\}, \tag{65}$$

where $\mathbf{b}(s)$ signifies the unit vector tangent to the surface in the direction of $-\dot{\mathbf{x}}(t)$ and $\mathbf{e}(s) = \mathbf{n}(s) \times \mathbf{b}(s)$. Note that substituting (65) into (63) yields

$$\begin{aligned} G(q)\ddot{q} + \left\{ \frac{1}{2}\dot{G} + S \right\} \dot{q} + C_0 \dot{q} + (I_4 - J^{+}J) g(q) \\ = -J^{\mathrm{T}} \Pi(s) \left\{ \begin{pmatrix} v_n - f \\ v_b \\ v_n \end{pmatrix} + \begin{pmatrix} g_n \\ g_b \\ g_e \end{pmatrix} \right\}, \end{aligned} \tag{66}$$

where

$$\begin{aligned} &\Pi(s) = (\mathbf{n}(s), \mathbf{b}(s), \mathbf{e}(s)), \\ &\begin{pmatrix} g_n \\ g_b \\ g_e \end{pmatrix} = \begin{pmatrix} g^{\mathrm{T}}(q) J^{+} \mathbf{n}(s) \\ g^{\mathrm{T}}(q) J^{+} \mathbf{b}(s) \\ g^{\mathrm{T}}(q) J^{+} \mathbf{e}(s) \end{pmatrix} = \Pi^{\mathrm{T}}(s) (J^{+})^{\mathrm{T}} g(q). \end{aligned} \tag{67}$$

Note that $\Pi(s)$ is an orthogonal matrix belonging to $SO(3)$, $\mathbf{g} = (g_n, g_b, g_e)^{\mathrm{T}}$ is a vector in $\mathbf{E}^3$, and a component of $g(q)$ to

$$\dot{q} = Q(q) \begin{pmatrix} \dot{\xi} \\ \dot{\eta} \\ \dot{\gamma} \end{pmatrix}, \tag{68}$$

$$Q(g) = (J^{+}(q)\mathbf{n}(s), W(q), J^{+}(q)(\mathbf{b}(s), \mathbf{e}(s))), \tag{69}$$

where $W(q)$ is the 4×1 unit vector satisfying $J(q)W(q) = 0$, $\dot{\xi}$ and $\dot{\eta}$ are a scalar and express a 2×1 velocity vector such that $(\mathbf{b}(s), \mathbf{e}(s))\dot{\gamma}(s) = \dot{\mathbf{x}}(t)$ at the contact point between the tip of the ball pen and the surface. Since $\dot{\xi} = 0$ that implies that the velocity of the tip of the pen in the direction $\mathbf{n}(s)$ normal to the surface is zero, substituting (68) into (66) and multiplying (66) by $Q^{\mathrm{T}}(q)$ yield

$$\alpha(\dot{\eta}, \ddot{\eta}, \dot{\gamma}, \ddot{\gamma}) = -(v_n - f + g_n), \tag{70}$$

$$\begin{aligned} G_{11}\ddot{\eta} + \left\{ \frac{1}{2}\dot{G}_{11} + S_{11} \right\} \dot{\eta} + W^{\mathrm{T}} C_0 W \dot{\eta} + W^{\mathrm{T}} g(q) \\ + G_{12}\ddot{\gamma} + B_{12}\dot{\gamma} = 0, \end{aligned} \tag{71}$$

$$\begin{aligned} G_{22}\ddot{\gamma} + \left\{ \frac{1}{2}\dot{G}_{22} + S_{22} \right\} \dot{\gamma} + C_{22}\dot{\gamma} \\ + G_{12}^{\mathrm{T}}\ddot{\eta} + B_{21}\dot{\eta} = -v - \begin{pmatrix} g_b \\ g_e \end{pmatrix}, \end{aligned} \tag{72}$$

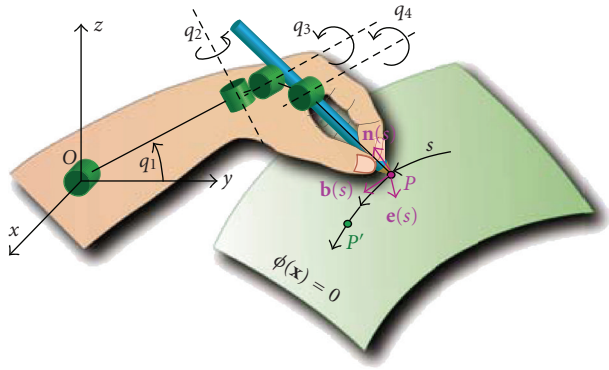


Figure 10: Handwriting on a hypersurface $\varphi(\mathbf{x}) = 0$.

where G_{11} is of 1×1, G_{22} is of 2×2, and

$$\overline{G} = P^{\mathrm{T}} G P, \quad P = (W, J^{+}(\mathbf{b}, \mathbf{e})),$$
$$\overline{G} = \begin{pmatrix} G_{11} & G_{12} \\ G_{12}^{\mathrm{T}} & G_{22} \end{pmatrix}, \qquad \overline{B} = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}, \tag{73}$$
$$\overline{B} = \frac{1}{2}\dot{\overline{G}} + \overline{S} + P^{\mathrm{T}} C_0 P,$$
$$\overline{S} = P^{\mathrm{T}} S P - \frac{1}{2}\dot{P}^{\mathrm{T}} G P + \frac{1}{2} P^{\mathrm{T}} G \dot{P}.$$

In (70), α is a function of C^{∞}-class with respect to, $\dot{\eta}, \ddot{\eta}, \dot{y}, \ddot{y}$, and q. Thus, for a given desired trajectory of the tip of the ball pen, $\dot{y}(t) = \dot{\mathbf{x}}_d(t)$ and $\ddot{y}(t) = \ddot{\mathbf{x}}_d(t)$, it is possible to prove the existence of q_d and $\dot{\eta}(= \dot{\eta}_d)$ that satisfies

$$\dot{q}_d = P(q_d) \begin{pmatrix} \dot{\eta}_d \\ \dot{\mathbf{x}}_d \end{pmatrix},$$
$$G_{11}\ddot{\eta}_d + \left\{ \frac{1}{2}\dot{G}_{11} + S_{11} \right\} \dot{\eta}_d + C_{11}\dot{\eta}_d \tag{74}$$
$$+ W^{\mathrm{T}} g(q_d) + G_{12}\ddot{\mathbf{x}}_d + B_{12}\dot{\mathbf{x}}_d = 0,$$

where $C_{11} = W^{\mathrm{T}} C_0 W$. The two equations (74) are the set of five simultaneous differential equations of the 1st order in five variables of $\dot{\eta}_d$ and q_d. Once the desired trajectory of joint motion $q_d(t)$ with its first and second derivatives is determined together with the motion $\dot{\eta}_d(t)$ of excess-DOF, the desired control signal v_d is given by (72) by putting $\dot{y} = \dot{\mathbf{x}}_d$, $q = q_d$, and $\dot{\eta} = \dot{\eta}_d$. Moreover, if a desired pressing force f_d is specified, the desired control signal in the direction normal to the surface can be constructed through (70) such that

$$v_n = f_d - g_n(q_d) - \alpha(\dot{\eta}_d, \ddot{\eta}_d, \dot{\mathbf{x}}_d, \ddot{\mathbf{x}}_d). \tag{75}$$

8. What Should the Central Processing Unit Learn?

Given a smooth curve $\gamma(s(t))$ on a hypersurface $\varphi(\mathbf{x}) = 0$ over time interval $t \in [0, T]$, it is shown in the previous section that there exists uniquely a control signal $v(t)$ lying in the image space that makes the endpoint of the robot trace the curve $\gamma(s(t))$ on the hypersurface. Irrespective of the redundancy of DOF, once a starting posture of the robot is specified in the Euclidean space $\mathbf{E}^3$ with its endpoint position $P = \mathbf{x}(0)$ on the hypersurface $\varphi(\mathbf{x}) = 0$ as shown in Figure 10, the orbit $q_d(t)$ that realizes $\mathbf{x}(q_d(t)) = \mathbf{x}_d(t)$ itself is determined uniquely correspondingly to the uniquely existing control signal $v_d(t)$ (see Figure 10). Then, there arises a problem whether it is possible to acquire such a desired control signal without knowing the details of the robot dynamics. In order to discuss the problem, let us consider a learning control scheme that is expressed by the following form of the control signal (see Figure 11):

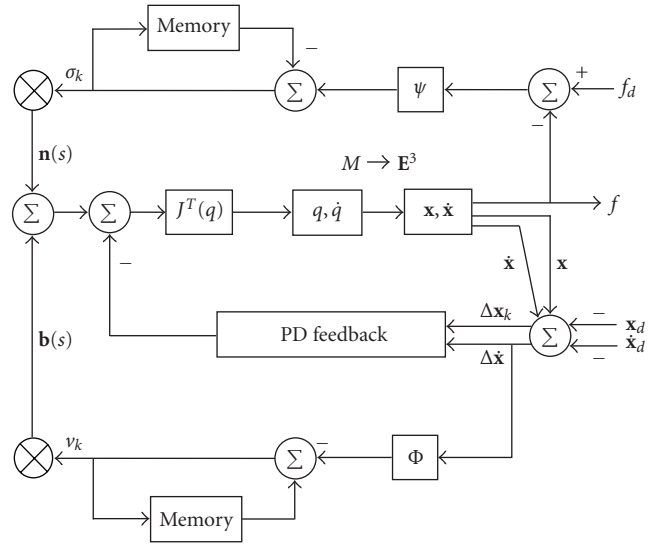


Figure 11: Iterative learning control scheme for robotic handwriting.

$$u_k = -J^{\mathrm{T}}(q)\{\kappa(\mathbf{x} - \mathbf{x}_d) + \zeta\sqrt{\kappa}(\dot{\mathbf{x}} - \dot{\mathbf{x}}_d)\}$$
$$- J^{\mathrm{T}}(q)\{\mathbf{v}_k + \sigma_k \mathbf{n}(s)\}, \tag{76}$$

where $\kappa > 0$ stands for a constant expressing the stiffness parameter of position feedback and $\zeta > 0$ a constant lying in interval $[0.5, 2.0]$ of $\mathbf{R}^1$. The first term of the right-hand side expresses the task space ($\mathbf{E}^3$) PD feedback for a given set of position and velocity $(\mathbf{x}_d, \dot{\mathbf{x}}_d)$ and the second term stands for the learning control signal at the kth operation that is constructed as follows:

$$\mathbf{v}_{k+1} = \mathbf{v}_k - \Phi\Delta\dot{\mathbf{x}}_k,$$
$$\sigma_{k+1} = \sigma_k + \psi\Delta f_k, \tag{77}$$

where ψ is an appropriate constant such that $0 < \psi < 1$ and Φ is a positive-definite 3×3 constant matrix. In the case of ordinary handwriting when the constraint surface is a horizontal plane, it is shown in the previous paper in [38] that the learning update law for the control signal based on (76) makes the motion of the endpoint trace the given curve $\mathbf{x}_d(t)$ on the plane and at the same time the joint motions

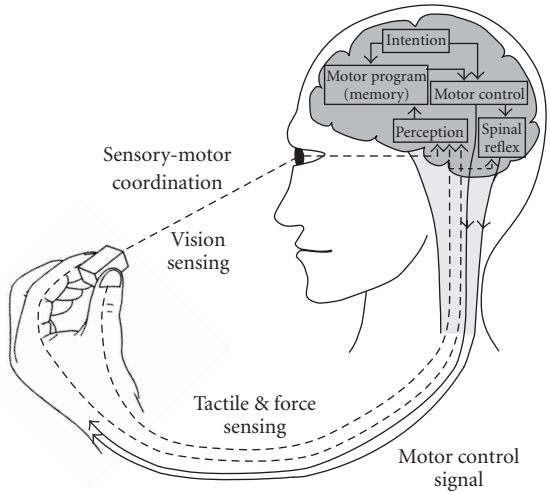


Figure 12: Sensory-motor coordination.

$q_k(t)$ and $\dot{q}_k(t)$ converge to the ideal trajectories $q_d(t)$ and $\dot{q}_d(t)$ as $k \to \infty$. The proof of convergence of the control signals $\mathbf{v}_k$ to the desired one $\mathbf{v}_d$ as $k \to \infty$ is also given in [39] on the basis of the "passivity" of the input $\mathbf{v}$ and the output $\dot{\mathbf{x}}$ of the system dynamics of (63) provided that u is given through (64) and the gravity effect $g(q)$ is carefully compensated. Some numerical simulation results are also presented in [39].

Returning to Bernstein's problem discussed in the introductory section, we are fortunately able to quote Latash's commentary [40] to the book originally written by Bernstein in Russian cited as [2]. Latash says,

Bernstein's definition of degrees of freedom relied heavily on the analysis of kinematic degrees of freedom of various systems. In particular, when analyzing the human upper limb, Bernstein considered all possible orthogonal axes of rotation in a joint as independent degrees of freedom, which were later summed over the joints. The apparent redundancy of the joints of the human arm in comparison to the three-dimensional space where we happen to live and where movements take place led Bernstein to his famous formulation that the essence of motor control is the elimination of the redundant degrees of freedom. The beauty and brevity of this formulation is stunning. A skeptic may want to decide whether control can always be reduced to the elimination of redundant degrees of freedom. In other words, I am suggesting that Bernstein's famous definition may not always be correct (blasphemy!!!)

Even in the case of simple humanlike multijoint point-to-point reaching, Riemannian geometry suggests a good reason to think that the central nervous system is functioning according to a natural law of Newton's mechanics, that is, the law of inertia for multijoint mechanisms. Once a geodesic curve $q^*(t)$ connecting the given initial posture q^0 and the equilibrium submanifold $N_{P'}$ (see (18)) is determined with some initial velocity $\dot{q}^*(0)$, then the three-dimensional orbit $\mathbf{x}(q^*(t))$ for $t \in [0, T]$ is obtained. This orbit can be rewritten into a three-dimensional curve $\gamma(s)$ with the aid of length parameter s for a corresponding interval $s \in [a, b]$. Then this curve can be spelled out by a desired trajectory given as a movement of the tip of the arm in $\mathbf{E}^3$ in the form of $\mathbf{x}_d(t) = \gamma(s(t))$ for $t \in [0, T]$ with the aid of a scale change of "time" through a monotonously increasing function $s(t)$. Once a desired orbit $\mathbf{x}_d(t)$ of motion of a tip of the arm is given in $\mathbf{E}^3$ and at the same time an initial posture of the arm is chosen, the trajectory of motion in the joint space is uniquely determined without eliminating any excess of the system's DOF. However, it is important to note that, even if there arises a small change of the initial posture, say $q^0 + \delta q$, it is possible to obtain the same desired orbit $\mathbf{x}_d(t)$ in $\mathbf{E}^3$ but the trajectory $q_d(t)$ of joints differes slightly from that obtained when the initial posture is set as $q(0) = q^0$ and also the desired control signal $v_d(t)$ may differ. Another noteworthy characteristic of humanlike multijoint movements is called "variability", which was first pointed out by Bernstein [2]. Latash [32] observed that in the case of human skilled motion the grade of variability of each endpoint trajectory is quite low relative to variable profiles of joint responses at each trial of reaching. The mathematical arguments in the previous section suggest that the main part of variability in joint space is caused by small fluctuations of choice of an initial posture trial by trial and the others may be noise.

We do not discuss the importance of redundancy of the muscles involved in a specific motion of multijoint reaching. The central nervous system surely uses its own means of communication with the muscles, which is not analogous to the language of joint kinematics that specifies the trajectories in individual joint. Instead of overcoming the problem of redundancy of the muscles, we implicitly assume in this paper that a desired torque of individual joint is finally generated by a total of forces of all the muscles involved in movement of the joint. This premise may not be justified by any reason, but it is important to quote Jackson's observation as a neurophysiologist [41] (see Figure 12):

"*To speak figuratively, the central nervous system knows nothing of muscles, it only knows movements.*"

"*The highest centres represent all parts of the body, literally all parts supplied by nerves.*"

9. Conclusions

This paper discusses difficult problems of control for humanlike robots with redundant DOFs from the standpoint of Riemannian geometry. Irrespective of joint redundancy, we have shown that there is no necessity to resolve the inverse kinematics problem by introducing any artificial performance index and optimizing it in both the cases of (1) multijoint reaching movement with excess joints, and (2) handwriting through iterative learning control. The most important conclusion is that Riemannian geometry does not care about the redundancy of joints through which rigid links are connected in series but it is powerful in establishing a correspondence between working-point trajectories in the external three-dimensional space ($\mathbf{E}^3$) and movement trajectories in a space of joint angles (a configuration space $\mathbf{R}^n$ or a Riemannian manifold $\{M, g_{ij}\}$ as a set of whole possible postures).

References

[1] M. Brady, *Robotics Research: The First International Symposium*, MIT Press, Cambridge, Mass, USA, 1948.

[2] N. A. Bernstein, "On dexterity and its development," in *Dexterity and Its Development*, M. L. Latash and M. T. Turvey, Eds., pp. 3–275, Lawrence Erlbaum Associates, Mahway, NJ, USA, 1996.

[3] N. A. Bernstein, *The Coordination and Regulation of Movements*, Pergamon, London, UK, 1967.

[4] R. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton, Fla, USA, 1994.

[5] G. Hinton, "Some computational solutions to Bernstein's problems," in *Human Motor Actions: Bernstein Reassessed*, H. T. A. Whiting, Ed., pp. 413–418, Elsevier, Amsterdam, The Netherlands, 1984.

[6] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 13, no. 5, pp. 834–846, 1983.

[7] J. Morimoto and K. Doya, "Reinforcement learning of dynamic motor sequence: learning to stand up," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1721–1726, Victoria, Canada, October 1998.

[8] B. Sicilliano and O. Khatib, Eds., *Springer Handbook of Robotics*, Springer, New York, NY, USA, 2008.

[9] J. C. Latombe, Ed., *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, Mass, USA, 1991.

[10] V. I. Arnold, Ed., *Methematical Methods of Classical Mechanics*, Springer, New York, NY, USA, 2nd edition, 1989, translated by K. Vogtmann and A. Weinstein.

[11] M. Takegaki and S. Arimoto, "A new feedback method for dynamic control of manipulators," *Journal of Dynamic Systems, Measurement and Control*, vol. 103, no. 2, pp. 119–125, 1981.

[12] F. Bullo and A. D. Lewis, Eds., *Geometric Control of Mechanical Systems: Modeling, Analysis, and Design for Simple Mechanical Control Systems*, Springer, New York, NY, USA, 2000.

[13] W. M. Oliva, Ed., *Geometric Mechanics*, vol. 1798 of *Lecture Notes in Mathematics*, Springer, Berlin, Germany, 2002.

[14] S. Arimoto, M. Yoshida, M. Sekimoto, and K. Tahara, "A riemannian-geometry approach for dynamics and control of object manipulation under constraints," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1683–1690, Kobe, Japan, May 2009.

[15] S. Arimoto, "Modeling and control of multi-body mechanical systems part 1: a Riemannian geometry approach," *International Journal of Factory Automation, Robotics and Soft Computing*, no. 2, pp. 108–122, 2009.

[16] M. Sekimoto, S. Arimoto, S. Kawamura, and J.-H. Bae, "Skilled-motion plannings of multi-body systems based upon Riemannian distance," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1233–1238, Pasadena, Calif, USA, May 2008.

[17] S. Arimoto, *Control Theory of Nonlinear Mechanical Systems: A Passivity-Based and Circuit-Theoretic Approach*, Oxford University Press, Oxford, UK, 1996.

[18] J. Jost, *Riemannian Geometry and Geometric Analysis*, Springer, Berlin, Germany, 2002.

[19] S. Gallot, D. Hulin, and J. Lafontaine, *Rimennian Geometry*, Springer, Berlin, Germany, 2004.

[20] T. Yoshikawa, "Manipulability of robotic mechanisms," *International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[21] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*, Addison-Wesley, Reading, Mass, USA, 1991.

[22] D. E. Whitney, "Resolved motion rate control of manipulators and human protheses," *IEEE Transactions on Man-Machine Systems*, vol. 10, no. 2, pp. 47–53, 1969.

[23] O. Khatib, "A unified approach for motion and force control of robot manipulators: the operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[24] J. M. Hollerbach and K. C. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE Journal of Robotics and Automation*, vol. 3, no. 4, pp. 308–316, 1987.

[25] V. Potkonjak, S. Tzafestas, D. Kostic, G. Djordjevic, and M. Rasic, "The handwriting problem," *IEEE Robotics and Automation Magazine*, vol. 10, no. 1, pp. 35–46, 2003.

[26] P. Morasso, "Spatial control of arm movements," *Experimental Brain Research*, vol. 42, no. 2, pp. 223–227, 1981.

[27] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[28] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multijoint arm movement: minimum torque-change model," *Biological Cybernetics*, vol. 61, no. 2, pp. 89–101, 1989.

[29] S. Arimoto, M. Sekimoto, H. Hashiguchi, and R. Ozawa, "Natural resolution of ill-posedness of inverse kinematics for redundant robots: a challenge to Bernstein's degrees-of-freedom problem," *Advanced Robotics*, vol. 19, no. 4, pp. 401–434, 2005.

[30] S. Arimoto, H. Hashiguchi, M. Sekimoto, and R. Ozawa, "Generation of natural motions for redundant multi-joint system: a differential-geometric approach based upon the principle of least actions," *Journal of Robotic Systems*, vol. 22, no. 11, pp. 583–605, 2005.

[31] S. Arimoto and M. Sekimoto, "Human-like movements of robotic arms with redundant DOFs: virtual spring-damper hypothesis to tackle the Bernstein problem," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '06)*, vol. May 2006, pp. 1860–1866, Orlando, Fla, USA, 2006.

[32] M. L. Latash, *Neurophysiological Basis of Movement*, Human Kinetics, New York, NY, USA, 1998.

[33] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Comparative experiments on task space control with redundancy resolution," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1575–1582, Edmonton, Canada, August 2005.

[34] J. M. Wintess and L. Stark, "Muscle models: what is gained and what is lost by varying model complexty," *Biological Cybernetics*, vol. 55, no. 6, pp. 403–420, 1987.

[35] E. Bizzi, N. Hogan, F. A. Mussa-Ivaldi, and S. Giszter, "Does the nervous systems use equilibrium-point control to guide single and multiple joint movements," *Behavioral and Brain Sciences*, vol. 15, no. 4, pp. 603–613, 1992.

[36] M. Kawato and H. Gomi, "A computational model of four regions of the cerebellum based on feedback-error learning," *Biological Cybernetics*, vol. 68, no. 2, pp. 95–103, 1992.

[37] N. Hogan, "The mechanics of multi-joint posture and movement control," *Biological Cybernetics*, vol. 52, no. 5, pp. 315–331, 1985.

[38] S. Arimoto, H. Hashiguchi, and R. Ozawa, "Simple control method coping with a kinematically ill-posed inverse problem of redundant robots: analysis in case of a handwriting robot," *Asian Journal of Control*, vol. 7, no. 2, pp. 112–123, 2005.

[39] S. Arimoto, M. Sekimoto, and S. Kawamura, "Task-space iterative learning for redundant robotic systems: existence of a task-space control and convergence of learning," *SICE Journal of Control, Measurement, and System Integration*, vol. 1, no. 4, pp. 312–319, 2008.

[40] M. L. Latash, "The Bernstein problem: how does the central nervous system make its choices," in *Dexterity and Its Development*, pp. 227–303, Lawrence Erlbaum Associates, Mahway, NJ, USA, 1996.

[41] M. L. Jackson, "Observations on the physiology and pathology of hemi-chorea," in *John Hughlings Jackson: Selected Writings*, J. Taylor, G. Holmes, and F. M. R. Walshe, Eds., vol. 2, p. 400, Hodder and Stoughton, London, UK, 1932.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 307293, 9 pages
doi:10.1155/2010/307293

Research Article

Parameterless-Growing-SOM and Its Application to a Voice Instruction Learning System

Takashi Kuremoto, Takahito Komoto, Kunikazu Kobayashi, and Masanao Obayashi

Graduate School of Science and Engineering, Yamaguchi University, Tokiwadai 2-16-1, Ube, Yamaguchi 755-8611, Japan

Correspondence should be addressed to Takashi Kuremoto, wu@yamaguchi-u.ac.jp

Received 5 January 2010; Revised 22 April 2010; Accepted 21 June 2010

Academic Editor: Ivo Bukovsky

An improved self-organizing map (SOM), parameterless-growing-SOM (PL-G-SOM), is proposed in this paper. To overcome problems existed in traditional SOM (Kohonen, 1982), kinds of structure-growing-SOMs or parameter-adjusting-SOMs have been invented and usually separately. Here, we combine the idea of growing SOMs (Bauer and Villmann, 1997; Dittenbach et al. 2000) and a parameterless SOM (Berglund and Sitte, 2006) together to be a novel SOM named PL-G-SOM to realize additional learning, optimal neighborhood preservation, and automatic tuning of parameters. The improved SOM is applied to construct a voice instruction learning system for partner robots adopting a simple reinforcement learning algorithm. User's instructions of voices are classified by the PL-G-SOM at first, then robots choose an expected action according to a stochastic policy. The policy is adjusted by the reward/punishment given by the user of the robot. A feeling map is also designed to express learning degrees of voice instructions. Learning and additional learning experiments used instructions in multiple languages including Japanese, English, Chinese, and Malaysian confirmed the effectiveness of our proposed system.

1. Introduction

Kohonen's self-organizing map (SOM) is a kind of a neural network which maps a high-dimensional input onto a regular low-dimensional grid orderly by unsupervised learning schemes [1–4]. Because of its simple algorithm and powerful performance, SOM has been developed and applied widely to the fields of pattern recognition, signal processing, intelligent control, and so on [5–15]. In a website of SOM library [6], more than 7,000 papers concern with this technique are collected.

Generally, SOM algorithm maps an n-dimension feature data in an input space $\mathbf{x}(x_1, x_2 \ldots, x_n)$ to a unit i in a low-dimensional output space with connections $\mathbf{m}_i(m_1, m_2 \ldots, m_n)$ by a simple rule using Euclidean distance, winner-takes-all,

$$c = \arg\min_i(\|\mathbf{x} - \mathbf{m}_i\|), \tag{1}$$

that is, a high-dimensional input is corresponded to a most suitable unit i with position c, best-match-unit (BMU) on the output map. For all inputs and initial connections with random values, a competitive learning rule enhances that the input data with similar features keep closely on the visualized topological output map

$$\Delta\mathbf{m}_i = \alpha h_{ci}(\mathbf{x} - \mathbf{m}_i), \tag{2}$$

where α is learning rate and h_{ci} is a neighborhood function

$$\mathbf{h}_{ci} = \exp\left(-\frac{\|c_i - c\|^2}{2\sigma^2}\right). \tag{3}$$

Here, c_i, c denote the positions of an arbitrary unit on the output map and BMU, respectively, $i = 1, 2, \ldots, k \leq N \times M$, σ is a constant. Obviously, $\mathbf{h}_{ci}(\mathbf{x}) \geq 0$, $\mathbf{h}_{ci}(0) = 1$, and $\mathbf{h}_{ci}(\infty) = 0$.

In fact, the size of the output space $N \times M$ in the original SOM is fixed in advance, and parameters such as learning rate α and the scale of neighborhood σ are often determined empirically. These constraints result in 2 kinds of problems

in technical applications [6–14]:

(1) the fixed size of output map prevents additional learning when new feature data are presented and BMU*s* are difficult to be found on the trained output map;

(2) annealing schemes for tuning the learning rate and the neighborhood size are necessary to improve the operation rate of output map; however, it usually increases computational load to realize the annealing.

Variations of SOM with growing structures are proposed to solve the first problem [7–10]. The basic idea of these kinds of SOM is to set the output feature map with a small size initially, for example, 2 units, then insert rows/columns into the map in training, where/when a most visited BMU exists [7, 10] or the deviation of the distance between the units on input layer and output map [8, 9]. We proposed another kind of method to solve the lack of units by using a memory layer to store matured units on the feature map during training process and release the matured units to be initialized, that is, the units come to available to be reused [12, 13]. When a feature data set is input to the learning system, the process searches corresponding BMU on memory layer at first, feature map which is produced by SOM just become to an intermediate map, so we called it transient SOM (T-SOM).

To solve the second problem, there have been also various approaches such as reducing learning rate (α in (2)) and neighborhood size (σ in (3)) linearly, that is, multiplying attenuation coefficients, calculating the neighborhood size in the input space, or using Kalman filters to find BMU on the output space [6]. Berglund and Sitte proposed a low-cost parameterless SOM algorithm (PLSOM) recently which uses the fitting error between the input and the map only to decide the annealing schemes [11].

In this paper, we combine the idea of growing SOM algorithm and the method of PLSOM to construct a novel SOM names parameterless-growing-SOM (PL-G-SOM) to tackle both problems of SOM described above. This new PL-G-SOM increases its structure adapting to the input data, and anneals parameters to realize sensitive clustering on the output space automatically. We also adopt PL-G-SOM into a voice instruction learning system where it serves as an automatic classifier of input features as well as T-SOM has been applied to a hand image instruction learning system [12, 13] and a voice instruction learning system [14].

The rest of this paper is organized as follows. Section 2 presents the details of PL-G-SOM. Section 3 shows a voice instruction learning system using PL-G-SOM. In Section 4, instruction learning experiments with 4 languages were reported to confirm the ability of learning and additional learning of the proposed system. Section 5 is the conclusion.

2. A New SOM: PL-G-SOM

2.1. Growing of Output Map. To construct a growing SOM which is more sensitive to larger categories of input data comparing with the SOM with fixed size in advance, different criteria have been proposed. Fritzke chose to insert a new row/column adjacent to a most often visited BMU in his Growing Grid [7]. The reason for this criterion of map enlargement is that the earlier map may be considered as a coarse one and likelihood BMUs need to raise their resolution to deal with the change of input. Meanwhile, Bauer and Villmann suggested adding units in the direction or even new dimension of the largest error between input data and the output map in their GSOM [8, 9]. However, the process of enlarging the output map either in Growing Grid or GSOM is similar and it is shown in Figure 1. In fact, when a new row/column needs to be inserted to the neighbor of a BMU c, for example, in the middle of c and f, the weights

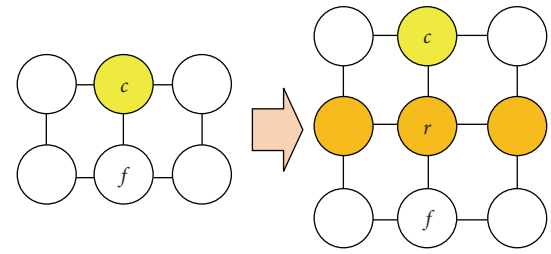


Figure 1: Insert a row/column into the feature map. Unit c is a BMU, and f is the farthest unit among the neighbors of c, r the inserted row/column.

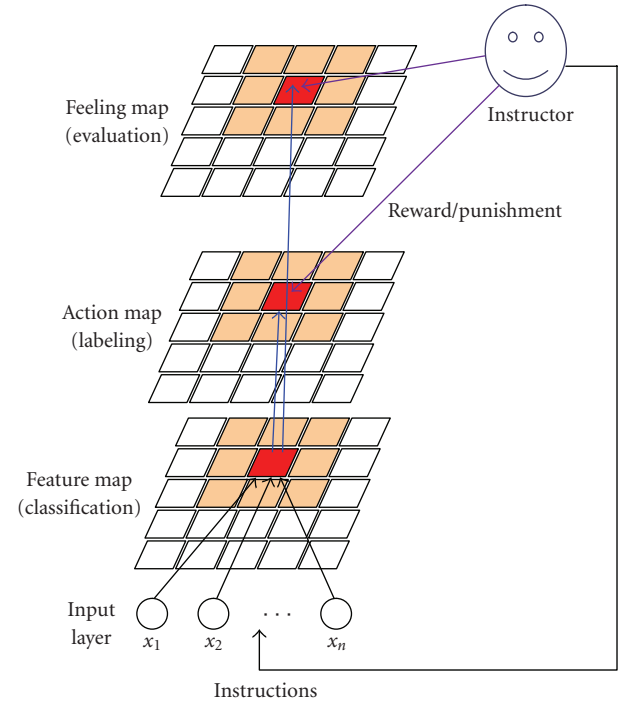


Figure 2: The structure of a voice instruction learning system using PL-G-SOM. It is similar to the system using T-SOM in [12–14], however, instead of memory layer of BMU in T-SOM, each map grows with training. Annealing schemes of their neighborhood size and learning rates are given by PL-G-SOM.

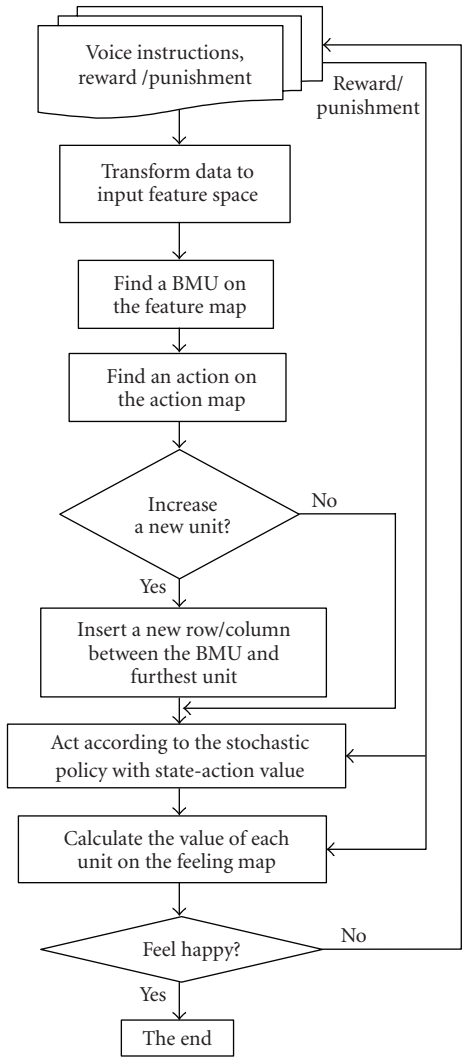


Figure 3: Flow chart of the proposed voice instruction learning system processing.

of connections between input and new nodes take average values of c and f:

$$\mathbf{m}_r = 0.5\left(\mathbf{m}_c + \mathbf{m}_f\right), \tag{4}$$

and so do them of r's neighbors

$$\mathbf{m}_{r\pm l} = 0.5\left(\mathbf{m}_{c\pm l} + \mathbf{m}_{f\pm l}\right), \tag{5}$$

where $l = 1, 2, \ldots, N$ or M. Unit f is chosen which has a largest Euclidean distance from the BMU c among the neighbors of c, and after this process, the map size changes to $N \times (M+1)$, or $(N+1) \times M$.

We use the same growing process here however, a new criterion to choose the BMU is proposed by concerning with a reinforcement learning algorithm when SOM is adopted into a human-machine interaction learning system. The detail will be given in Section 3.

2.2. Annealing of Parameters. To decide the learning rate and the size of neighborhood function, we adopt the method of PLSOM proposed by Berglund and Sitte [11]. Either the learning rate $\alpha = \varepsilon(t)$ or the neighborhood size $\sigma(t)$ is calculated by the distance between input and the BMU:

$$\begin{aligned}
\varepsilon(\mathbf{t}) &= \frac{\|x(\mathbf{t}) - m_c(\mathbf{t})\|^2}{r(\mathbf{t})},\\
r(t) &= \max\left(\|x(t) - m_c(t)\|^2, r(t-1)\right),\\
r(0) &= \|x(0) - m_c(0)\|^2,\\
\mathbf{h}_{\mathbf{ci}}^{\varepsilon}(\mathbf{t}) &= \exp\left(-\frac{\|\mathbf{c_i} - \mathbf{c}\|^2}{(\sigma(\varepsilon(\mathbf{t})))^2}\right),\\
\sigma(\varepsilon(\mathbf{t})) &= \sigma_{\max} \cdot \varepsilon(\mathbf{t}), \quad \sigma(\varepsilon(\mathbf{t})) \geq \sigma_{\min},
\end{aligned} \tag{6}$$

where $\sigma_{\max}$, $\sigma_{\min}$ are positive parameters, for example, the value may be the size of the map and 1.0, respectively.

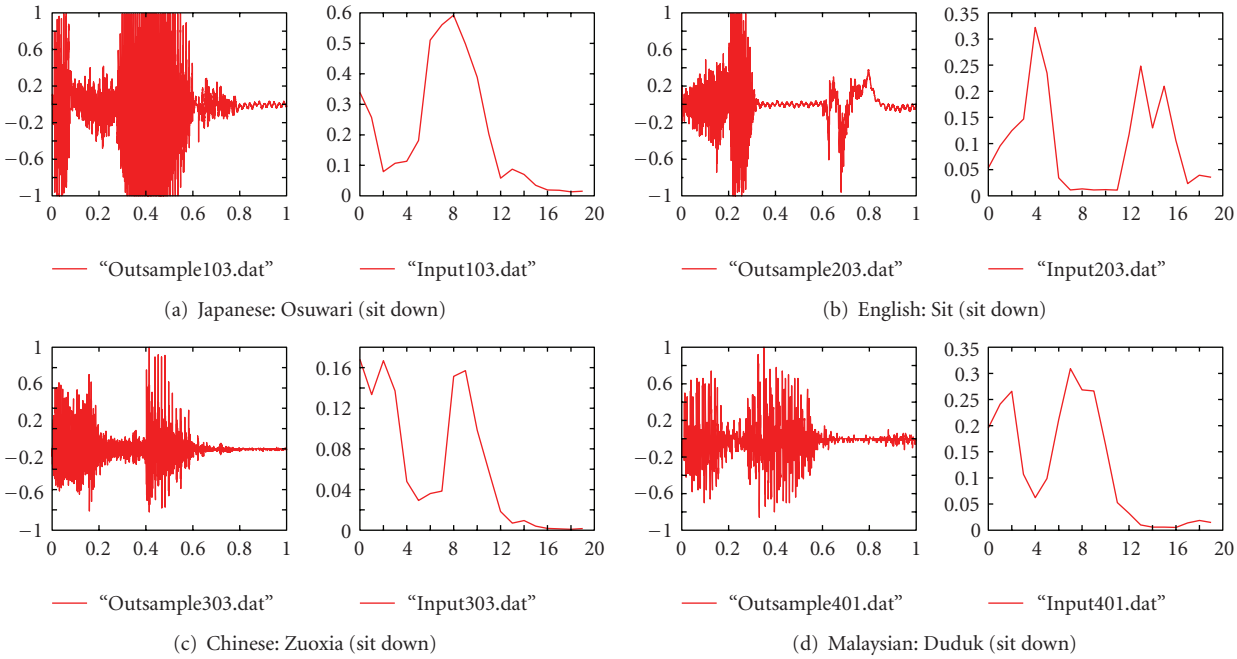
The competitive learning rule of the connections between input and output units that is, (2), can be changed to an online learning algorithm

$$\begin{aligned}
m_i(\mathbf{t}+1) &= m_i(\mathbf{t}) + \Delta m_i(\mathbf{t})\\
&= m_i(\mathbf{t}) + \varepsilon(\mathbf{t})\mathbf{h}_{\mathbf{ci}}^{\varepsilon}(\mathbf{t})(x(\mathbf{t}) - m_i(\mathbf{t})).
\end{aligned} \tag{7}$$

3. A Voice Instruction Learning System Using PL-G-SOM

A voice instruction learning system is supposed as an internal model of an autonomous robot which performs kinds of available actions when external signal in voices is presented at first and learns to output requested actions using the reward or punishment from the instructor. So the system supports the robot to keep learning and additional learning abilities. For example, a robot with the voice instruction learning system is able to "understand" human's instructions in different languages, or a pet robot like "AIBO" [16] comes easily to used to change a new owner.

3.1. The Structure. To realize the human-machine interaction, an internal model of autonomous robot is constructed as shown in Figure 2. The structure is similar to a learning system using Transient-SOM (T-SOM) which is proposed in our previous work [12–14]. In [12, 13], a hand image instruction learning system which has 5 layers including Input Layer, Feature Map, Action Map, Feeling Map, and Memory Layer is composed with SOM algorithm and reinforcement learning rules. Instructions to the robot are presented by kinds of shapes of human's hand, and robots categorize them, that is, image signals in an 80-dimensional space with SOM, and the instructions are labeled with a series of autonomous actions according to a stochastic policy. Instructor observes the action of the robot and provides reward/punishment of the action to robot, so the action policy of the robot is able to be modified to cooperate with the instructions of hand images. For online learning and additional learning, T-SOM adopted a memory layer which



(a) Japanese: Osuwari (sit down)

(b) English: Sit (sit down)

(c) Chinese: Zuoxia (sit down)

(d) Malaysian: Duduk (sit down)

FIGURE 4: An instruction (*sit down*) features input to robot in different languages. Left: sound waves; right: normalized features in 20-dimentional space.

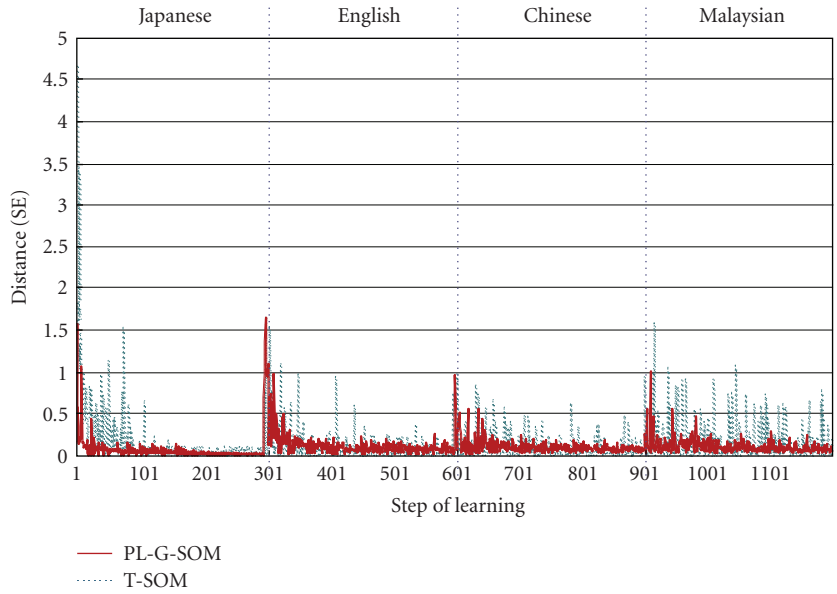


FIGURE 5: Comparison of the Euclidean distance (SE: squared error) between input and BMUs in learning (the first 300 iterations)/additional learning process between T-SOM [12–14] (broken lines) and PL-T-SOM (solid lines).

stores "matured" BMUs, and input features are matched with units on the memory layer before executing SOM on feature map. We also adopted annealing plan to decide the size of neighborhood and learning rate into T-SOM, and a voice instruction learning system using the improved T-SOM named PL-T-SOM was developed in [14]. However, a problem that exists in T-SOM is that its memory layer stores only the value of matured units without the topology of the feature map. Even if the memory layer could remember the topology of the feature map trained online, the new topology would not be able to be established on it. For this reason, we propose a new voice instruction learning system using PL-G-SOM given in Section 2 instead of T-SOM.

In Figure 2, Feature Map is the basic growing SOM and the size of Action Map and Feeling Map growing with the Feature Map too. In fact, instructions given by voice data are transformed into feature vectors of input space (layer) at first, the PL-G-SOM algorithm is then executed

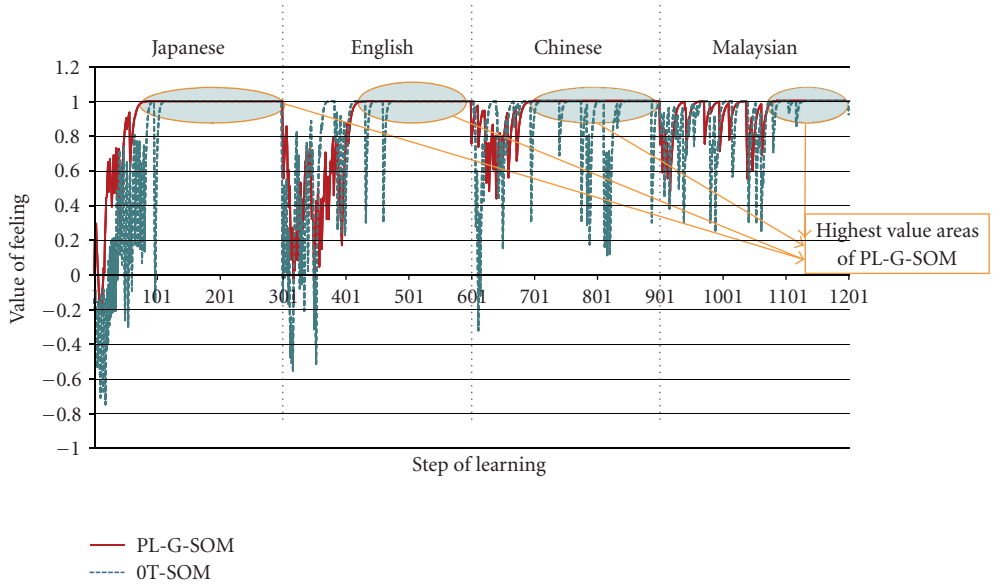


Figure 6: Feeling values rose to the maximum happiness 1.0 according to training iterations. PL-G-SOM proposed here (solid lines) showed faster and longer convergence than T-SOM in [12–14] (dash lines).

on the Feature Map, and the rules of growing given by (4) and (5) (Figure 1) are also applied to increase Action Map and Feeling Map. Action Map is composed by those units which correspond to the units on Feature Map; that is, each unit on Action Map represents each kind of features of input data. The units on Action Map are labeled by a reinforcement learning algorithm given by Section 3.2 to limit each feature to adaptive actions of the robot. Feeling Map has the same distribution of units as which of Action Map. The action number that comes from Action Map is furnished with a feeling value which expresses the degree of the action mastered by the robot. The details of Feeling Map are described in Section 3.3.

3.2. Reinforcement Learning Algorithm. The value of units on Action Map is given by a value function of state and action, that is, (8), where $Q(s_t, a_t(i))$ has the value of selected action $a_t(i)$ when the robot is in the state s_t, and $Q(s_0, a_0(i))$ = random numbers initially:

$$Q_{t+1}(s_{t+1}, a_{t+1}(i)) = Q_t(s_t, a_t(i)) \pm r, \tag{8}$$

where $\pm r$ is the empirical value of reward (+)/punishment (−) given by the instructor, for example, a positive constant when the robot acted correctly according to its policy function and a negative constant oppositely.

Now suppose that there are $N \times M$ units that exist on Action Map; that is, $N \times M$ states exist in the environment of Markov decision process (MDP), each unit has K actions to be selected available, then a reinforcement learning (RL) algorithm [17] can be used to label the classes of the states which are units on Action Map yielded by the Feature Map. According to (8), a Q-value table can be established as shown in Table 1.

For each state s_t that is, presented voice instruction, robot intends to select a valuable action $a_t(i)$ according to a stochastic action policy π given by Gibbs distribution (Boltzmann distribution) as shown in

$$\pi_t(a_t(i) \mid s_t) = \frac{e^{Q_t(s_t, a_t(i))/T}}{\sum_{j \in A} e^{Q_t(s_t, a_t(j))/T}}. \tag{9}$$

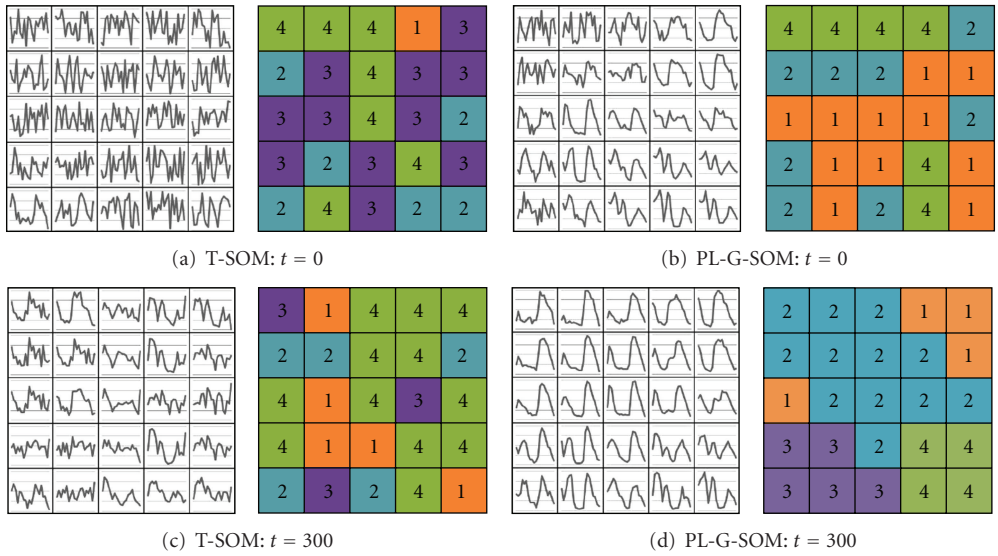
Here, T is a positive parameter named temperature [17], higher T causes an active exploration of actions (each action is selected under a similar probability), and lower T gives a greedy selection of the action with higher Q value oppositely.

We propose to use $Q(s_t, a_t(i))$ as a criterion of growing the size of Feature Map, Action Map, and Feeling Map. In fact, when the robot chooses an action with high $Q(s_t, a_t(i))$ but instructor judges that it is wrong, then a new row/column is inserted nearby the s_t, that is, the BMU c. The growing process is described in Section 2.1.

Table 1: Q-value table. Each unit of Action Map has a $Q_t(s_t, a_t(i))$ value corresponding to an action.

s_t	$a_t(1)$	$a_t(2)$	$a_t(i)$	$a_t(K)$
1	6	2	−8	0
2	10	1	0	1
⋯				
$N \times M$	0	2	7	2

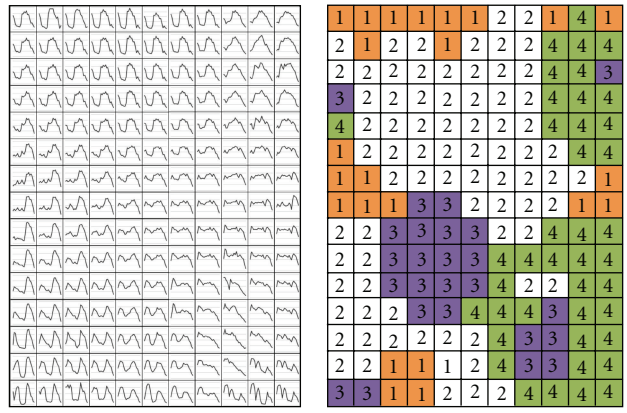
3.3. Feeling Map. To express the degree of how a voice instruction is learned by robot, a Feeling Map which has the same number of units with Action Map is designed (Figure 2). The distance from input pattern to BMU of Feature Map and the reward from instructor are used to calculate feeling values which is normalized in [−1.0, 1.0] where high positive value means happiness and 0.0 is the



(a) T-SOM: $t = 0$ (b) PL-G-SOM: $t = 0$

(c) T-SOM: $t = 300$ (d) PL-G-SOM: $t = 300$

FIGURE 7: Change of each map during learning process. Left maps show Feature Maps and Right maps show Action Maps in (a)–(d). Comparing the learning result showed by (c) and (d), it is easy to find that except of the action "1", PL-G-SOM showed to be more effective in gathering the similar input on its Action Map as neighbored output comparing with T-SOM.



(a) Feature Map of PL-G-SOM ($t = 1200$) (b) Action Map of PL-G-SOM ($t = 1200$)

FIGURE 8: Results of feature classification and instruction learning/additional learning using PL-G-SOM. The sizes of maps grew to 11×15 (165 units) when they began with 5×5 (25 units) in the experiment.

initial value of each unit; negative values express sadness. The learning algorithm which was also used in [12–14] is given by

$$F_{t+1}(i) = F_t(i) \pm aC - bD_i, \tag{10}$$

where $F(i)$ notes the feeling value of unit i on the Feeling Map (zero initially), C notes the continue times of reward or punishment, D_i is the Euclidean distance (squared error) between the unit on Feature Map corresponding to i, and the input data, a, b are constants and $0 < a < 1$, $0 < b \ll 1$.

4. Experiments

4.1. Descriptions. Learning and additional learning experiments were performed using the system with PL-G-SOM proposed in Section 3 and the system with T-SOM in [12–14].

Four kinds of voice instructions were used in experiments: *sit down*, *lie down*, *stand up*, and *walk*. Instructions in Japanese were used to training the system. Additional learning using voice instructions given by other languages was executed after training using the Japanese. Three kinds of languages: English, Chinese, and Malaysian were used to confirm additional learning ability of the system. The voices were recorded in a normal room by 3 males who pronounced each instruction 3 times. So, there were 3 samples of one instruction used for each kind of languages while 4 actions with 48 samples.

Sound waves were preprocessed by normalization and noise elimination, and windowed by 20 intervals to yield

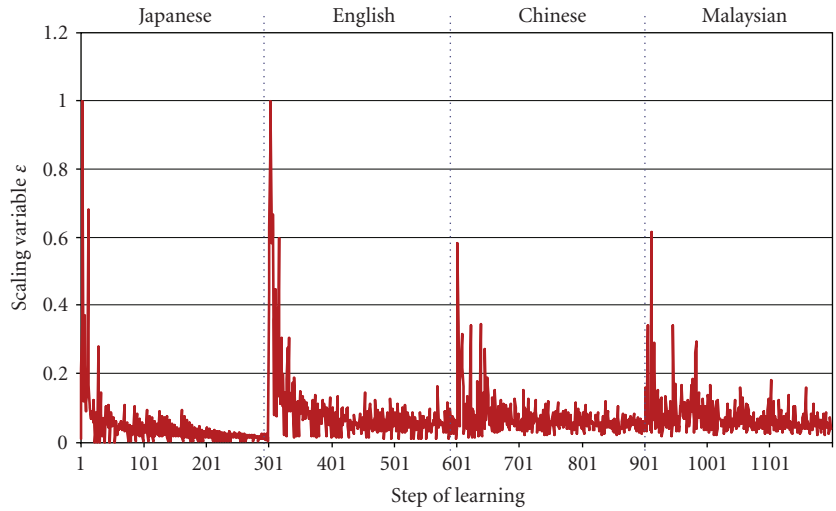


Figure 9: The change of scaling variable ε of PL-G-SOM in the learning/additional learning process.

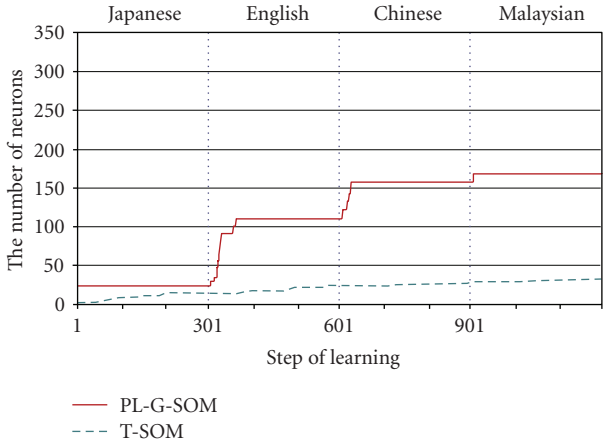


Figure 10: Comparison of recognition error (SE: squared error between input and BMU) in learning process between T-SOM and PL-T-SOM.

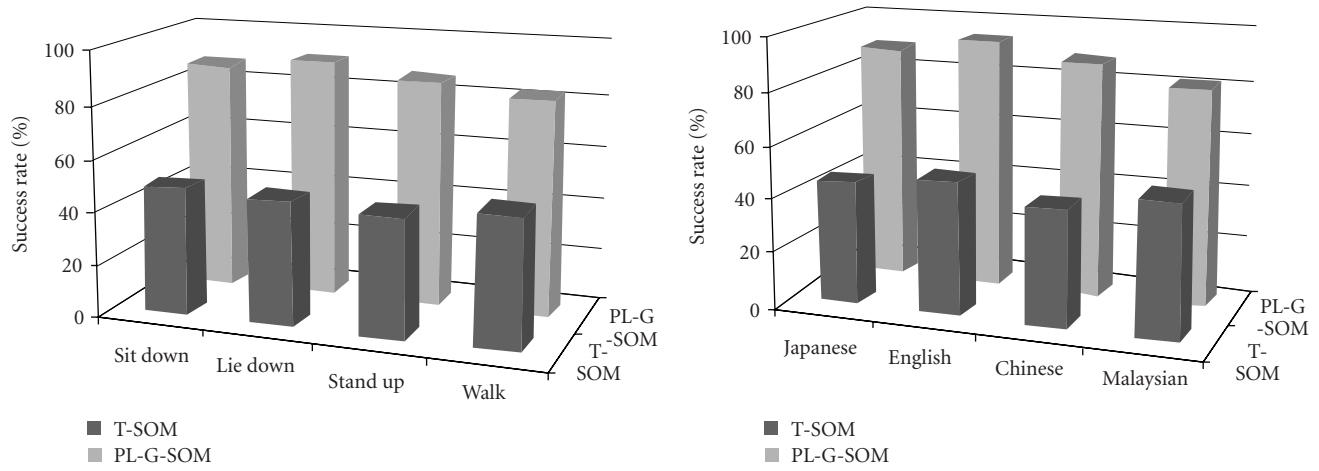
Table 2: Parameters used in the experiments.

Description	Symbol	Quantity
Size of Feature Map of T-SOM	$N \times M$	5×5
Size of initial PL-G-SOM	$N \times M$	5×5
Iteration times for one instruction	t	300×4
Temperature	T	1.0
Number of instructions (actions)	$a(i)$	4
Maximum/Minimum neighborhood in PL-G-SOM	$\sigma_{\max}, \sigma_{\min}$	$N \times M/2$, 0.7
Reward for one action selected	r	10.0
Parameters of Feeling Map	a, b	0.2, 0.05
Number of samples	—	48
Sampling rate	—	8 KHz
Sample size	—	8 bit
Channel	—	monaural

20 feature vectors of input space. Figure 4 shows an example of instruction "*sit down*" pronounced in Japanese ("Osuwari"), English ("Sit"), Chinese ("Zuoxia"), and Malaysian ("Duduk"). Parameters used in the experiments are shown in Table 2.

4.2. Results and Analyses. Either T-SOM or PL-G-SOM realized 100% recognition rates for 4 actions in different languages after learning and additional learning. However, PL-G-SOM showed faster and better convergence than T-SOM when the Euclidean distance (SE: squared error) between input and BMUs (Figure 5). This means that the classification to the input pattern was executed more efficiently by PL-G-SOM. Furthermore, the feeling values which express instruction recognition rate showed more obviously that correct actions of robot corresponding to instructions in voices were acquired more quickly and stably (Figure 6). Figure 7 shows the internal states of Feature Map (left) and Action Map (right) changing in training. The curves in each unit on Feature Map in Figure 7 express values of $\mathbf{m}_i$ $(m_1, m_2 \ldots, mn)$, $i = 1, 2, \ldots, N \times M$. Numbers with different colors on the Action Map express the different actions which were classified (labeled) by the reinforcement learning process described in Section 3.

Figures 7(a) and 7(b) show the initial states of T-SOM and PL-G-SOM, where random numbers were used. Figures 7(c) and 7(d) are the results of learning using Japanese instructions. Comparing with T-SOM, PL-G-SOM showed more effective on the topology formation of actions; that is, numbers of actions on the Action Map clustered more clearly. After additional learning, that is, using English, Chinese, and



(a) Recognition rates of different actions with noisy voices (in 4 languages)

(b) Recognition rates of different languages with noisy voices (for 4 kinds of actions)

Figure 11: Comparison of recognition rates between T-SOM and PL-T-SOM using voice instructions with 10% noises.

Table 3: Recognition rates of different actions with noises.

Language	Method	Success rate (10% noises)	Success rate (20% noises)	Success rate (30% noises)
Japanese	T-SOM	45.9	40.9	35.0
	PL-G-SOM	88.5	57.8	46.6
English	T-SOM	50.8	45.3	27.0
	PL-G-SOM	93.2	57.6	39.9
Chinese	T-SOM	44.5	41.1	32.5
	PL-G-SOM	88.3	60.1	43.9
Malaysian	T-SOM	51.0	40.0	37.1
	PL-G-SOM	81.9	54.9	47.2
Average	T-SOM	48.0	41.6	33.1
	PL-G-SOM	86.7	57.6	44.4

Malaysian 300 times, respectively, the size of Feature Map and Action map of PL-G-SOM grew from 25 (5×5) units to 165 (11×15) (Figure 8).

The scaling variable ε used in PL-G-SOM ((6)-(7)) changed with the training, and by Figure 9, one can confirm that ε decreased eventually during learning at first; however, when a new kind of language was input, the scaling variable ε suddenly changed to be larger and repeated its annealing scheme. Figure 10 shows the increase of the number of units on Memory Layer of T-SOM and the increase of the number of units on PL-G-SOM. Both units grew with additional learning and the number of units on Memory Layer of T-SOM stopped at 33, meanwhile 140 units were inserted into PL-G-SOM each layer. To confirm the robustness of the two learning system, we also tested noisy samples.

Table 3 shows the results of recognition rates of different actions with 10%, 20%, and 30% noises added to the 48 voice samples (i.e., N% of data in 20 dimensions were replaced by random numbers between $[0.0, 1.0]$). The average rate of success actions using T-SOM and PL-G-SOM was 48.0% and 86.7, respectively, given by 10 times of executions. Table 4 shows the results of recognition rates of different languages with the respective noisy samples.

Figure 11 shows the comparison of recognition rates of T-SOM and PL-G-SOM when 10% noises existed in all 48 instruction samples.

The results using PL-G-SOM proposed here show advantages than those with conventional learning system in all cases. In fact, we also investigated the use of frequency features for recognition of different instructions, however, similar results were observed in the experiments.

5. Conclusion

PL-G-SOM, a novel self-organizing map, was proposed using a reinforcement learning algorithm and annealing schemes of parameters. Online learning and additional learning are available with PL-G-SOM, and it was adopted into a voice instruction learning system of autonomous robot instead of conventional T-SOM. Experiments results showed that the advantage of the new learning system is speed and noise robustness.

Table 4: Recognition rates of different languages with noises.

Instruction	Method	Success rate (10% noises)	Success rate (20% noises)	Success rate (30% noises)
Sit down	T-SOM	48.8	55.2	33.6
	PL-G-SOM	87.2	62.0	42.0
Lay down	T-SOM	47.2	34.0	36.4
	PL-G-SOM	90.8	67.2	56.0
Stand up	T-SOM	45.6	33.2	25.2
	PL-G-SOM	86.4	48.4	33.2
Walk	T-SOM	50.4	44.0	37.6
	PL-G-SOM	82.4	52.8	46.4
Average	T-SOM	48.0	41.6	33.1
	PL-G-SOM	86.7	57.6	44.4

Acknowledgment

This paper was supported by Grant-in-Aid for Scientific Research (JSPS nos. 20500207, 20500277).

References

[1] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[2] T. Kohonen, "Analysis of a simple self-organizing process," *Biological Cybernetics*, vol. 44, no. 2, pp. 135–140, 1982.

[3] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1–3, pp. 1–6, 1998.

[4] T. Kohonen, *Self-Organizing Maps*, Springer Series in Information Sciences, Springer, Berlin, Germany, 1995.

[5] M. Cottrell, J. C. Fort, and G. Pagès, "Theoretical aspects of the SOM algorithm," *Neurocomputing*, vol. 21, no. 1–3, pp. 119–138, 1998.

[6] Bibliography of SOM, http://www.cis.hut.fi/nnrc/refs/.

[7] B. Fritzke, "Growing grid—a self-organizing network with constant neighborhood range and adaptation strength," *Neural Processing Letters*, vol. 2, no. 5, pp. 9–13, 1995.

[8] H.-U. Bauer and T. Villmann, "Growing a hypercubical output space in a self-organizing feature map," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 218–226, 1997.

[9] T. Villmann and H.-U. Bauer, "Applications of the growing self-organizing map," *Neurocomputing*, vol. 21, no. 1–3, pp. 91–100, 1998.

[10] M. Dittenbach, D. Merkl, and A. Rauber, "Growing hierarchical self-organizing map," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '00)*, vol. 6, pp. 15–19, July 2000.

[11] E. Berglund and J. Sitte, "The parameterless self-organizing map algorithm," *IEEE Transactions on Neural Networks*, vol. 17, no. 2, pp. 305–316, 2006.

[12] T. Kuremoto, T. Hano, K. Kobayashi, and M. Obayashi, "For partner robots: a hand instruction learning system using transient-SOM," in *Proceedings of the 2nd International Conference on Natural Computation and the 3rd International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '06)*, pp. 403–414, 2006.

[13] T. Hano, T. Kuremoto, K. Kobayashi, and M. Obayashi, "A hand image instruction learning system using transient-SOM," *Transaction on Society of Instrument and Control Engineering*, vol. 43, no. 11, pp. 1004–1006, 2007.

[14] T. Kuremoto, T. Komoto, K. Kobayshi, and M. Obayashi, "A voice instruction learning system using PL-T-SOM," in *Proceedings of the 2nd International Congress on Image and Signal Processing (CISP '09)*, pp. 4294–4299, 2009.

[15] V. Moschou, D. Ververidis, and C. Kotropoulos, "Assessment of self-organizing map variants for clustering with application to redistribution of emotional speech patterns," *Neurocomputing*, vol. 71, no. 1–3, pp. 147–156, 2007.

[16] AIBO, http://www.jp.aibo.com/.

[17] S. S. Sutton and A. G. Barto, *Reinforcement Learning: An Instruction*, The MIT Press, London,UK, 1998.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 919306, 7 pages
doi:10.1155/2010/919306

Research Article

How Can Brain Learn to Control a Nonholonomic System?

Noriyasu Homma,[1] Shinpei Kato,[2] Takakuni Goto,[3] Ivo Bukovsky,[1,4] Ryuta Kawashima,[3] and Makoto Yoshizawa[1]

[1] *CyberScience Center, Tohoku University, 6-6-05 Aoba, Aramaki, Aoba-ku, Sendai 980-8579, Japan*
[2] *Graduate School of Engineering, Tohoku University, 6-6-05 Aoba, Aramaki, Aoba-ku, Sendai 980-8579, Japan*
[3] *Institute of Development, Aging and Cancer, Tohoku University, Seiryo-machi 4-1, Aoba-ku, Sendai 980-8575, Japan*
[4] *Faculty of Mechanical Engineering, Czech Technical University in Prague, Technicka 4, 166 07 Prague 6, Czech Republic*

Correspondence should be addressed to Noriyasu Homma, homma@ieee.org

Received 14 December 2009; Accepted 5 March 2010

Academic Editor: Zeng-Guang Hou

Humans can often conduct both linear and nonlinear control tasks after a sufficient number of trials, even if they initially do not have sufficient knowledge about the system's dynamics and the way to control it. Theoretically, it is well known that some nonlinear systems cannot be stabilized asymptotically by any linear controllers and we have reported by an f-MRI experiment that different types of information may be involved in linear and nonlinear control tasks, respectively, from a brain function mapping point of view. In this paper, from a controllability analysis, we still show a possibility that human may use a linear control scheme for such nonlinear control tasks by switching the linear controllers with a virtual constraint. It is suggested that the proposed virtual constraint can play an important role to overcome a limitation of the linear controllers and to mimic human control behavior.

1. Introduction

In the fields of control engineering, system engineering, and brain sciences, the excellent human's control abilities have been widely studied [1–6]. Among such studies, Wolpert and Kawato [7] have proposed a new model of human control mechanism called Modular Selection And Identification Control (MOSAIC) in which many elemental prediction and control system modules are combined in order to conduct the desired motion. Imamizu et al. [8] have evaluated the MOSAIC model using linear control tasks. They used a functional MRI (f-MRI) scanner to observe brain activities during the control tasks and reported plausibility of the model.

However, human ability is not limited to the linear control case. In fact, humans can often achieve difficult control tasks after a sufficient number of trials, even if they initially do not have sufficient knowledge about the system's dynamics and the way to control it. For example, Goto et al. [9] have reported manual control ability to operate a 2-link planar under actuated manipulator (2PUAM). The 2PUAM is a nonlinear system with a nonholonomic constraint and cannot be stabilized asymptotically by any linear controller. In this case, to control such nonholonomic manipulator, the operators have to plan the object's trajectory first and then control the system to follow the trajectory. To do so, the operator must use shape information of the manipulator while the operator does not need such shape information to control linear manipulators.

To verify this difference between linear and nonlinear control tasks, we have conducted another f-MRI experiment using the 2PUAM [10]. The results suggest that the difference can be observed through the brain activities. In fact, a specific area involving the shape information processing was activated with a significant difference compared to the linear task only when subjects control the 2PUAM.

In this paper, to further clarify the experimental results, a new model-based analysis is conducted. The model is a multiple model-based reinforcement learning (MMRL) [11] for nonlinear controls but it is a linear controller. Using the MMRL, we attempt to explain the f-MRI results of the human information processing for the 2PUAM control task from a control theory point of view.

2. 2-Link Planer Under Actuated Manipulator (2PUAM)

In this study, we use the 2PUAM shown in Figure 1 as the control object. The 2PUAM moves in a horizontal plane. Thus, the 2PUAM is free from gravity, and arbitrary positions are equilibrium points. Motor is mounted on the first joint. Then the dynamic equations of the 2PUAM are

$$\begin{aligned} M_{11}\ddot{\theta}_1 + M_{12}\ddot{\theta}_2 + C_1 &= T_1 + T_{fr1}, \\ M_{21}\ddot{\theta}_1 + M_{22}\ddot{\theta}_2 + C_2 &= T_2 + T_{fr2}, \end{aligned} \tag{1}$$

where

$$\begin{aligned} M_{11} &= (m_1 + m_2)l_2^2 + m_2 l_2^2 + 2m_2 l_1 l_2 \cos\theta_2, \\ M_{12} &= M_{21} = m_2 l_2^2 + m_2 l_1 l_2 \cos\theta_2, \\ M_{22} &= m_2 l_2^2, \\ C_1 &= -m_2 l_1 l_2 \dot{\theta}_1 \left(2\dot{\theta}_1 + \dot{\theta}_2\right) \sin\theta_2 + c_1\dot{\theta}, \\ C_2 &= m_2 l_1 l_2 \dot{\theta}_1^2 \sin\theta_2 + c_2\dot{\theta}_2. \end{aligned} \tag{2}$$

Here, as shown in Figure 1, m_1 and m_2 are the mass of the first and second links, l_1 and l_2 are the length, c_1 and c_2 are the resistance coefficient, respectively, θ_1 and θ_2 are angles, $\dot{\theta}_1$ and $\dot{\theta}_2$ are angular velocities, and $\ddot{\theta}_1$ and $\ddot{\theta}_2$ are angular accelerations of joints, respectively. The right-hand sides of (1) are the input and friction torques. In this study, we simplify the model as follows:

$$\begin{aligned} m_1 = m_2 = m, \qquad l_1 = l_2 = l, \\ c_1 = c_2 = 0, \qquad T_2 = T_{fr1} = T_{fr2} = 0. \end{aligned} \tag{3}$$

It is known that the 2PUAM is not controllable by the standard continuous feedback. This implies that reaching the 2PUAM and stopping it at some point is a difficult task. Although some indirect methods have been proposed to control the 2PUAM from the control engineering point of views [9], it is still unclear how human can control the nonlinear object such as the 2PUAM.

3. Summary of f-MRI Experiments

3.1. Control Tasks and Training. Here, human operator is required to feed an input torque $\tau \in [-1, 1]$ so that the end effector of the arm will be driven to the goal point and kept at the point. Since the position of the first joint is fixed, there are at least two objective positions of the second joint as shown in Figure 2: the upper position of the second joint p_1 and the lower p_2.

The manual control experiment was conducted by 6 neurologically normal subjects (19–24 years of age; six males) participated in the experiments. All subjects were right-handed. Informed written consent was obtained from each participant. They had no knowledge about the dynamic response of the system before the experiment. They observed the virtual manipulator's states (positions, angles, and velocities of the 2PUAM) through visual data displayed on an LCD

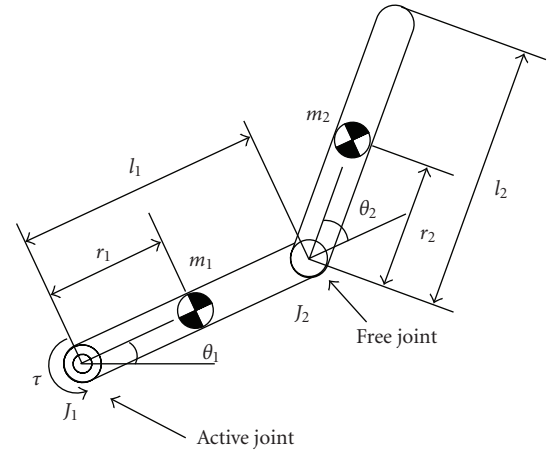


Figure 1: 2-link planner under actuated manipulator (2PUAM).

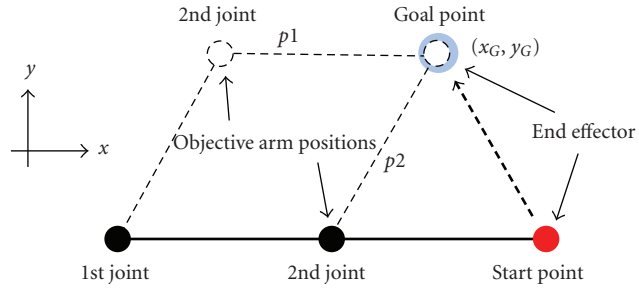


Figure 2: Positioning task.

monitor and fed the torque by using a joystick. The time limit was 30 seconds for each trial, and 300 training trials have been conducted outside of the MRI scanner before the following scanning sessions. The duration of 300 training trials depended on the subject's willingness and tiredness, but all the subjects completed them for 2-3 days.

For each trial, the performance index defined by the following equation was recorded:

$$J = \sum_{k=0}^{T} \sqrt{(x(k) - x_G)^2 + (y(k) - y_G)^2}. \tag{4}$$

Here $(x(k), y(k))$ denotes the position of the end effector at time $k\Delta t$ $(k = 0, 1, 2, \ldots, T)$, Δt is the interval of time step, (x_G, y_G) is the position of the goal point, and T is the maximum steps of each trial. The performance index J has been displayed on the monitor to guide subjects' criterion.

3.2. MRI Scanning Sessions. In scanning sessions, the trained subjects control two kinds of virtual manipulators whose shapes are projected on a screen in the MRI scanner as shown in Figure 3, by using an optical (magnet-free) joystick [10]. Both manipulators are the same in shape, but the first manipulator is the 2PUAM and the other is a manipulator which has a linear input and output relation (subjects can control it like a PC mouse). Controlling the 2PUAM is the main task while the linear control task is a comparative one,

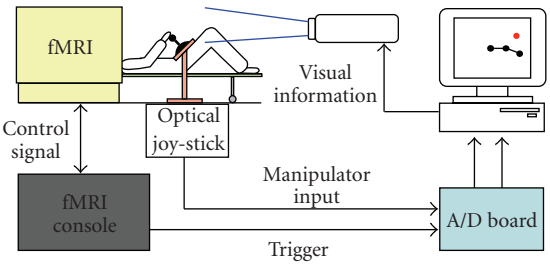


Figure 3: Experimental system.

and thus the former is called test trial and the latter is baseline trial, and their duration are called test and baseline periods, respectively. In other words, subjects can directly operate the coordinate of the end effector of the manipulator by the joystick in baseline trial. On the other hand, subjects operate only the torque of the first joint of the manipulator by the joystick and indirectly control the coordinate of the end effector in test trial. In each trial, subjects try to move the end effector of the manipulator to the goal point and keep it at the goal. The coordinates of the goal points are chosen randomly and displayed on the screen when trial starts.

MRI scanner (1.5T SIEMENS: Symphony 1.5T) was used to obtain blood oxygen level-dependant contrast functional images. Images weighted with the apparent transverse relaxation time were obtained with an echo-planner imaging sequence (repetition time: 3.0 s; echo time: 50 ms; flip angle: 90; field of view: 256 × 256 mm). High-resolution anatomical images of all subjects were also acquired with a T1-weighted sequence.

3.3. Results. Figure 4 indicates regions significantly more activated during test periods than baseline periods ($t > 5.89$, $P < .001$). Activities in the primary motor cortex, the somatosensory cortex, the somatosensory association cortex, the prefrontal cortex, the inferior temporal gyrus, and the fusiform gyrus were observed [10].

In the nonlinear control task, significant activities of the inferior temporal gyrus and the fusiform gyrus were observed. These areas are known to have an intimate involvement in recognizing the characteristic (color and shape) of the object. On the other hand, the prefrontal cortex is known as a region that receives the information that is necessary for action planning from both the temporal association area and the occipital association area and assembles complicated action plan. These suggest that in operating 2PUAM, subjects use the information about shape or position of 2PUAM and make trajectory planning of the positioning task based on that information. From a viewpoint of control theory, it may be worth to mention that such information is not needed for the linear control, but it is necessary for the nonholonomic systems control.

4. Controllability Analysis

The difference between linear and nonlinear control tasks observed through the significant brain activities can be a reflection of the human control mechanism that can cognize the target nonlinear dynamics and use an appropriate piece of information. However, this is not sufficient to conclude that human does not use any linear control scheme. Instead, the hypothesis proposed in this paper is that human can use a linear control scheme by switching linear controllers responsible for specific regions where the linear approximation can work well for the target nonlinear task. The following linear model can then be employed to verify the hypothesis.

4.1. Multiple Model-Based Reinforcement Learning (MMRL). MMRL has multiple modules that are pairs of prediction model which predicts future state of the controlled objects and reinforcement learning controller which learns the control output. "Responsibility signals" are calculated from the softmax function of the prediction errors. The prediction model which outputs the more accurate prediction has the larger responsibility signal. By weighting control signal and learning of the each module with responsibility signals, these modules are adapted to the corresponding specific situations, respectively.

In this study, we use a multiple linear quadratic controller (MLQC) by using multiple linear prediction and quadratic reward models as an efficient implementation of the MMRL [11]. Change in the state vector of the target system, $x \in R^4$, is given by

$$\dot{x}(t) = f(x(t), u(t)), \tag{5}$$

where x is the state vector of the system, and u is the control output. Each variable of the vector for the 2PUAM can be defined as

$$x_1 = \theta_1, \quad x_2 = \dot{\theta}_1, \quad x_3 = \theta_2, \quad x_4 = \dot{\theta}_2. \tag{6}$$

Linear prediction models of the MMRL can be represented as follows:

$$\hat{\dot{x}}_i(t) = f_i(x(t), u(t)) = A_i x(t) + B_i u(t), \quad i = 1, 2, \ldots, n. \tag{7}$$

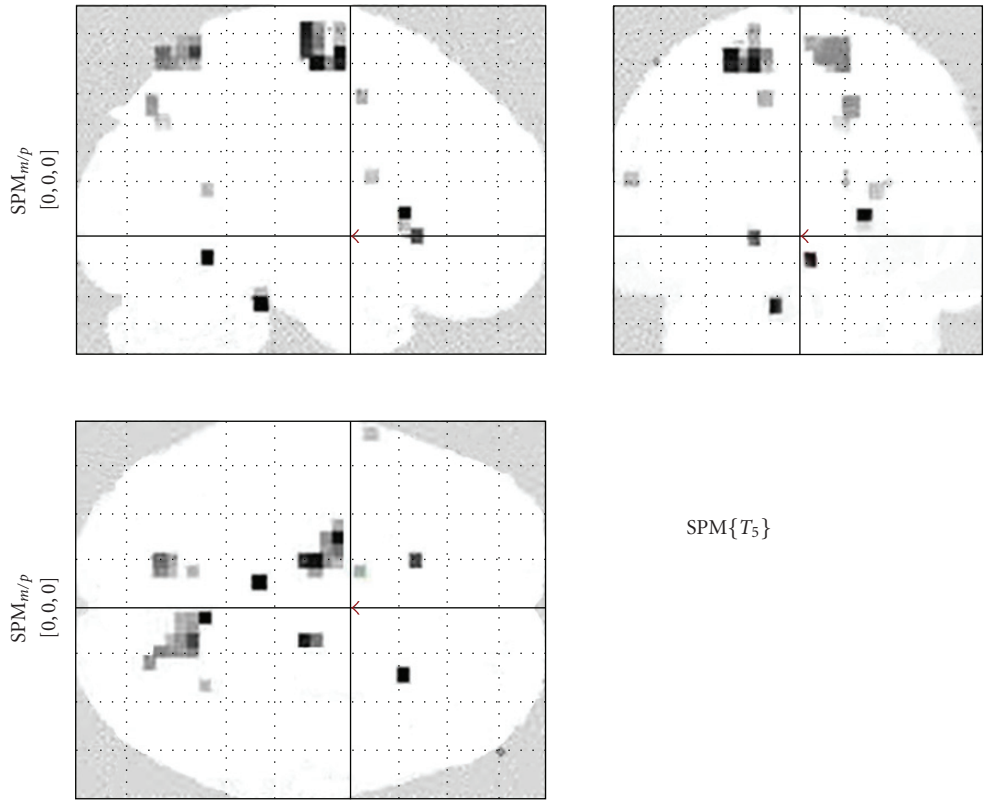
where n denotes the number of modules. State prediction $\hat{\dot{x}}$ is given by a weighting sum of prediction models with responsibility signals λ_i [11]:

$$\hat{\dot{x}}(t) = \sum_{i=1}^{n} \lambda_i(E_i(t))\hat{\dot{x}}_i(t) = \sum_{i=1}^{n} \lambda_i(E_i(t))(A_i x(t) + B_i u(t)), \tag{8}$$

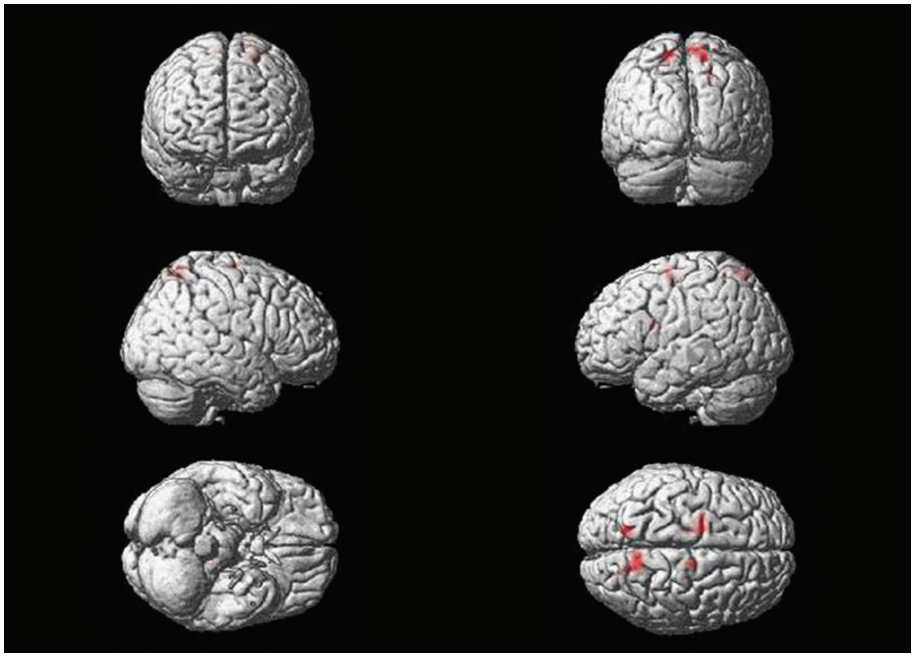
where $E_i(t)$ is a short-time average of the prediction error $\dot{x}(t) - \hat{\dot{x}}(t)$. Learning of each prediction model in (7), $i \in \{1, 2, \ldots, n\}$, is conducted by changing its parameter vector w_i consisting of all the elements of the matrices A_i and B_i as follows:

$$\dot{w}_i(t) = \eta_w \lambda_i(E_i(t)) \left(\frac{\partial f_i}{\partial w_i}\right)^T \left(\dot{x}(t) - \hat{\dot{x}}(t)\right), \tag{9}$$

where η_w (> 0) is an update coefficient. Schematic diagram of the multiple predictor-controller pair architecture is shown in Figure 5.



(a)



(b)

Figure 4: Significant activated areas of test period compared to baseline period (for more details, see [10]).

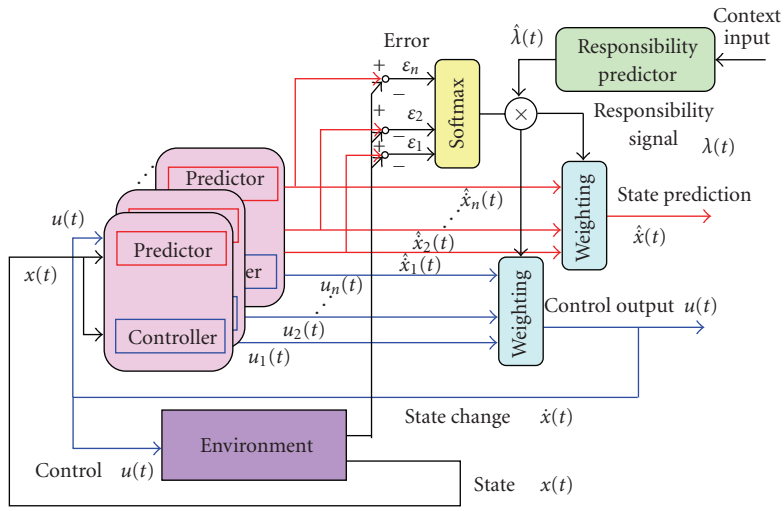


Figure 5: Schematic diagram of multiple predictor-controller pair architecture [11].

4.2. Controllability of 2PUAM. We obtain the following state equation of the 2PUAM from (1) and (6) under the condition in (3):

$$\begin{aligned}
\dot{x}_1 &= x_2,\\
\dot{x}_2 &= \frac{1}{2-\cos^2 x_3}\{(3+\cos x_3)x_2^2 \sin x_3\\
&\qquad +x_2 x_4 \sin x_3 + u/ml^2\},\\
\dot{x}_3 &= x_4,\\
\dot{x}_4 &= -(1+\cos x_3)\dot{x}_2 - x_2^2 \sin x_3.
\end{aligned} \tag{10}$$

Equation (10) can be described using the vector form as

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = f\left(\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix}, u\right). \tag{11}$$

Here, we define the microscopic fluctuation at some positions x_a in phase space as $\delta x = x - x_a$, $\delta u = u - u_a$. Calculating the Taylor series of (11) and ignoring the higher-order terms from the second order, we get

$$\delta\dot{x} = A\delta x + B\delta u. \tag{12}$$

Here matrices A and B are

$$A = \frac{\partial f}{\partial x} = \begin{pmatrix} 0 & 1 & 0 & 0\\ 0 & \frac{\partial \dot{x}_2}{\partial x_2} & \frac{\partial \dot{x}_2}{\partial x_3} & \frac{\partial \dot{x}_2}{\partial x_4}\\ 0 & 0 & 0 & 1\\ 0 & \frac{\partial \dot{x}_4}{\partial x_2} & \frac{\partial \dot{x}_4}{\partial x_3} & \frac{\partial \dot{x}_4}{\partial x_4}\end{pmatrix},$$

$$B = \frac{\partial f}{\partial u} = \frac{1}{aml^2}\begin{pmatrix}0\\ 1\\ 0\\ -(1+\cos x_3)\end{pmatrix}, \tag{13}$$

where

$$\begin{aligned}
\frac{\partial \dot{x}_2}{\partial x_2} &= \frac{(2x_2(3+\cos x_3)+x_4)\sin x_3}{a},\\
\frac{\partial \dot{x}_2}{\partial x_3} &= \frac{(6x_2+x_4)x_2\cos^2 x_3 - 2x_2^2}{a} + \frac{u\sin 2x_3}{ml^2 a},\\
\frac{\partial \dot{x}_2}{\partial x_4} &= \frac{x_2 \sin x_3}{a},\\
\frac{\partial \dot{x}_4}{\partial x_2} &= -(1+\cos x_3)\frac{\partial \dot{x}_2}{\partial x_2} - 2x_2 \sin x_3,\\
\frac{\partial \dot{x}_4}{\partial x_3} &= \dot{x}_2 \sin x_3 - (1+\cos x_3)\frac{\partial \dot{x}_2}{\partial x_3} - x_2^2 \cos x_3,\\
\frac{\partial \dot{x}_4}{\partial x_4} &= -(1+\cos x_3)\frac{x_2 \sin x_3}{a},\\
a &= 2-\cos^2 x_3.
\end{aligned} \tag{14}$$

Since the MMRL is a linear controller, it cannot control the 2PUAM. Indeed, the rank of the controllability matrix $U_c = [B\ AB\ A^2B\ A^3B]$ at any equilibrium position x_0 becomes 2, which is not the full rank, 4. That is, let us denote an equilibrium point by using arbitrary values θ_{10} and θ_{20}

$$x_0 = \begin{pmatrix}\theta_{10} & 0 & \theta_{20} & 0\end{pmatrix}^T \tag{15}$$

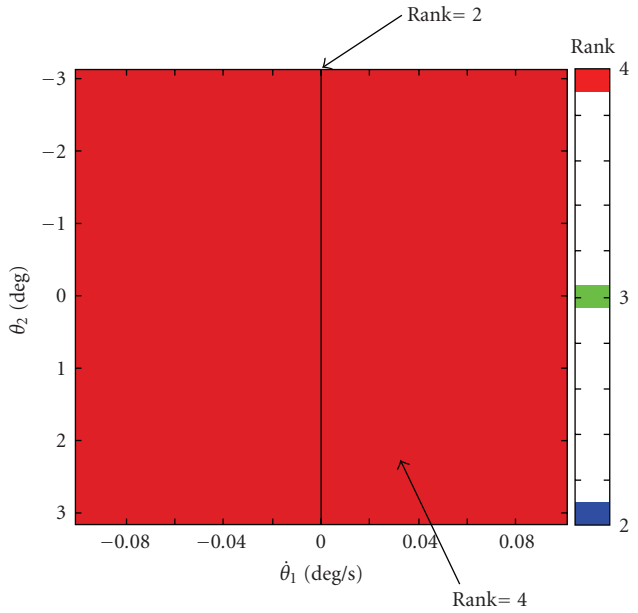


Figure 6: Rank of the controllability matrix U_c at some positions $x_s(\dot{\theta}_2 = 0, u = 0)$.

with no input $u = 0$. Then, from (14),

$$A = \left.\frac{\partial f}{\partial x}\right|_{u=0}^{x=x_0} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$
$$B = \left.\frac{\partial f}{\partial x}\right|_{u=0}^{x=x_0} = \frac{1}{a_0 m l^2} \begin{pmatrix} 0 \\ 1 \\ 0 \\ -(1+\cos\theta_{20}) \end{pmatrix}, \tag{16}$$

where $a_0 = 2 - \cos^2\theta_{20}$. In this case, the rank of the U_c is not equal to the full rank, 4, but is 2 as follows:

$$\text{rank}\,[U_c] = \text{rank}\,[B \;\; AB \;\; A^2B \;\; A^3B] = 2. \tag{17}$$

4.3. Possible Control Strategy. The rank of the controllability matrix can, however, be the full rank, 4, at some positions x_s in the phase space where angular velocity of the first joint is not exactly zero, $\dot{\theta}_1 \neq 0$, even if the $\dot{\theta}_1$ is very close to zero, as shown in Figure 6.

To verify the controllability, let us denote positions x_s in the phase space by using arbitrary values θ_{1s}, θ_{2s}, and $\dot{\theta}_{2s}$:

$$x_s = \left(\theta_{1s} \;\; \dot{\theta}_{1s} \;\; \theta_{2s} \;\; \dot{\theta}_{2s}\right)^T, \quad \dot{\theta}_{1s} \neq 0. \tag{18}$$

Then, from (14),

$$A_s = \left.\frac{\partial f}{\partial x}\right|_{u=u_s}^{x=x_s} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{\partial \dot{x}_2}{\partial x_2} & \frac{\partial \dot{x}_2}{\partial x_3} & \frac{\partial \dot{x}_2}{\partial x_4} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{\partial \dot{x}_4}{\partial x_2} & \frac{\partial \dot{x}_4}{\partial x_3} & \frac{\partial \dot{x}_4}{\partial x_4} \end{pmatrix},$$
$$B_s = \left.\frac{\partial f}{\partial x}\right|_{u=u_s}^{x=x_s} = \frac{1}{a_s m l^2} \begin{pmatrix} 0 \\ 1 \\ 0 \\ -(1+\cos\theta_{2s}) \end{pmatrix}, \tag{19}$$

where $a_s = 2 - \cos^2\theta_{2s}$. In this case, the rank of the controllability matrix is 4, the full rank:

$$\text{rank}\,[U_{cs}] = \text{rank}\left[B_s \;\; A_sB_s \;\; A_s^2B_s \;\; A_s^3B_s\right] = 4. \tag{20}$$

Thus, it is confirmed that 2PUAM cannot be stabilized at any equilibrium point, but if $\dot{\theta}_{1s} \neq 0$, the end effector of the 2PUAM can approach any geometrical point with any slow speed.

4.4. Discussions. The slow speed approach verified above might be an interesting result because, nevertheless human subjects can control the 2PUAM very well, it is often very hard even for human subjects to stop the 2PUAM completely in the manual control tasks [9, 10]. The linear controller can be responsible only for a small region in which the linear approximation can work well. Thus, by switching multiple linear models, there is a possibility to control the 2PUAM with the MMRL.

Different from (16), we assumed the condition in (19) where $\partial\dot{x}_i/\partial x_j \neq 0$, $i = 2, 4$, and $j = 2, 3, 4$ to make the rank of controllability matrix be the full rank. The condition implies a virtual constraint of the manipulator's shape (joint movement) because, for example, $\partial\dot{x}_2/\partial x_3 \neq 0$ implies the virtual existence of relation between $\ddot{\theta}_1$ and θ_2 that makes a constraint on the angular acceleration depended on the angle of the manipulator shape. Interestingly, if human subjects could feel and realize such virtual constraint on the 2PUAM control (joint movement), the control task can sometimes be achieved relatively easier [9].

According to the f-MRI results, subjects may use the shape and position information of the 2PUAM. The virtual constraint discussed above can further be created in order to make the control easier. Unfortunately, the controllability analysis does not prove this hypothesis, but there is a possibility that in operating the 2PUAM, subjects use the shape information to switch the controllers [10] and create the virtual constraint to make the control easier [9]. In this sense, MMRL can be regarded as a linear model for controlling the 2PUAM. Then, if human's superior learning ability to control the complex nonlinear system could be based on such linear control schemes, its implementation on a robot system might be easier than we expected.

5. Conclusions

In this paper, by using a controllability analysis, we have revised the previous f-MRI experimental results that reveal significant activation areas for the nonlinear control task compared to the linear one. Even the useful pieces of information for the linear task may be different from nonlinear ones, the analysis suggests some possibility to control the 2PUAM with a set of linear control models in a similar way by which human subjects can control it. In fact, to stop the 2PUAM at an equilibrium point completely seems very hard or almost impossible, but to approach there with any arbitrary slow speed seems an easier task. The additional information of shape and position of the 2PUAM can then be used for switching the linear controllers. Although the internal relation between the virtual constraint and the controllability of the nonlinear task is still unclear and should be clarified further from both computational and brain sciences point of views, the hypothesis proposed in this paper implies that it could be possible to design or realize robots with the human-level learning ability in an easier way.

References

[1] R. Coulom, "High-accuracy value-function approximation with neural networks applied to the acrobat," in *Proceedings of the 12th European Symposium on Artificial Neural Networks (ESANN '04)*, pp. 7–12, Bruges, Belgium, April 2004.

[2] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, no. 1, pp. 219–245, 2000.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Nuero-Dynamic Programming*, Athena Scientific, Belmont, Calif, USA, 1996.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, UK, 1998.

[5] E. Wiewiora, "Potential-based shaping and Q-value initialization are equivalent," *Journal of Artificial Intelligence Research*, vol. 19, pp. 205–208, 2003.

[6] A. Karniel and F. A. Mussa-Ivaldi, "Sequence, time, or state representation: how does the motor control system adapt to variable environments?" *Biological Cybernetics*, vol. 89, no. 1, pp. 10–21, 2003.

[7] D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, vol. 11, no. 7-8, pp. 1317–1329, 1998.

[8] H. Imamizu, T. Kuroda, T. Yoshioka, and M. Kawato, "Functional magnetic resonance imaging examination of two modular architectures for switching multiple internal models," *Journal of Neuroscience*, vol. 24, no. 5, pp. 1173–1181, 2004.

[9] T. Goto, N. Homma, M. Yoshizawa, and K. Abe, "An analysis of human learning process on manual control of complex systems," *Ergonomics*, vol. 42, no. 5, pp. 287–294, 2006 (Japanese).

[10] N. Homma, S. Kato, T. Goto, et al., "Human brain activities related to manual control of a nonholonomic system: an f-MRI study," *International Journal of Advanced Computer Engineering*, vol. 2, no. 2, pp. 129–133, 2009.

[11] K. Samejima, K. Katagiri, K. Doya, and M. Kawato, "Multiple model-based reinforcement learning of nonlinear control," *The Journal of The Institute of Electronics, Information and Communication Engineers*, vol. 84, no. 9, pp. 2092–2106, 2001 (Japanese).

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 139634, 11 pages
doi:10.1155/2010/139634

Research Article

Motion Intention Analysis-Based Coordinated Control for Amputee-Prosthesis Interaction

Fei Wang,[1,2] Shiguang Wen,[1] Chengdong Wu,[1] Yuzhong Zhang,[1] and Jincheng Li[1]

[1] *The institute of Artificial Intelligence & Robotics, College of Information Science & Engineering, Northeastern University, 110004 Shenyang, China*
[2] *State Key Laboratory of Robotics & Systems, Harbin Institute of Technology, 150001 Harbin, China*

Correspondence should be addressed to Fei Wang, wangfei@ise.neu.edu.cn

Received 1 November 2009; Revised 25 February 2010; Accepted 24 March 2010

Academic Editor: Zeng-Guang Hou

To study amputee-prosthesis (AP) interaction, a novel reconfigurable biped robot was designed and fabricated. In homogeneous configuration, two identical artificial legs (ALs) were used to simulate the symmetrical lower limbs of a healthy person. Linear inverted pendulum model combining with ZMP stability criterion was used to generate the gait trajectories of ALs. To acquire interjoint coordination for healthy gait, rate gyroscopes were mounted on CoGs of thigh and shank of both legs. By employing principal component analysis, the measured angular velocities were processed and the motion synergy was obtained in the final. Then, one of two ALs was replaced by a bionic leg (BL), and the biped robot was changed into heterogeneous configuration to simulate the AP coupling system. To realize symmetrical stable walking, master/slave coordinated control strategy is proposed. According to information acquired by gyroscopes, BL recognized the motion intention of AL and reconstructed its kinematic variables based on interjoint coordination. By employing iterative learning control, gait tracking of BL to AL was archived. Real environment robot walking experiments validated the correctness and effectiveness of the proposed scheme.

1. Introduction

Lower limb prosthesis is used to compensate the locomotion function for amputees in the field of biomedical rehabilitation. Conventional mechanical prosthesis has been criticized for difficulty in motion transformation, stiff-legged gait and poor mobility under complex condition. Intelligent prosthetic leg controlled by a microprocessing unit can realize the arbitrary gait precisely to coordinate with the sound leg of amputee [1]. It has been a challenging endeavor for interaction between amputee and prosthesis for their different structures, actuation manners, cognitive competence, and dynamic characters. To realize coordinated movement, prosthetic leg must be able to perceive the motion intention of amputee properly so as to actuate its joints accordingly when walking on different terrains with various cadences and stride length.

To guarantee the performance of prosthetic leg during development stage, a great amount of repetitive experiments that need amputee to participate entirely is necessary. It is not only costly but also painful to handicapped person, and even leads to accidental injury to amputee. Moreover, individual difference also makes it difficult to obtain the uniform and quantitative performance evaluation for prosthetic leg.

To solve problems mentioned above, a novel reconfigurable test-bed for prosthetic leg development is designed and fabricated by the Robotics Group at Northeastern University, China [2], which has two kinds of form called homogeneous configuration and heterogeneous configuration separately. The former is mainly used to study the symmetrical locomotion as many common biped robot systems [3, 4]. The latter, however, provides an ideal platform for the study of multiagent coordination, gait tracking and interaction for AP coupling system.

In this research, the heterogeneous configuration of test-bed is used to validate the master/slave dual-leg coordination strategy of the AP coupling system, motion intention recognition, and gait tracking based on iterative learning control.

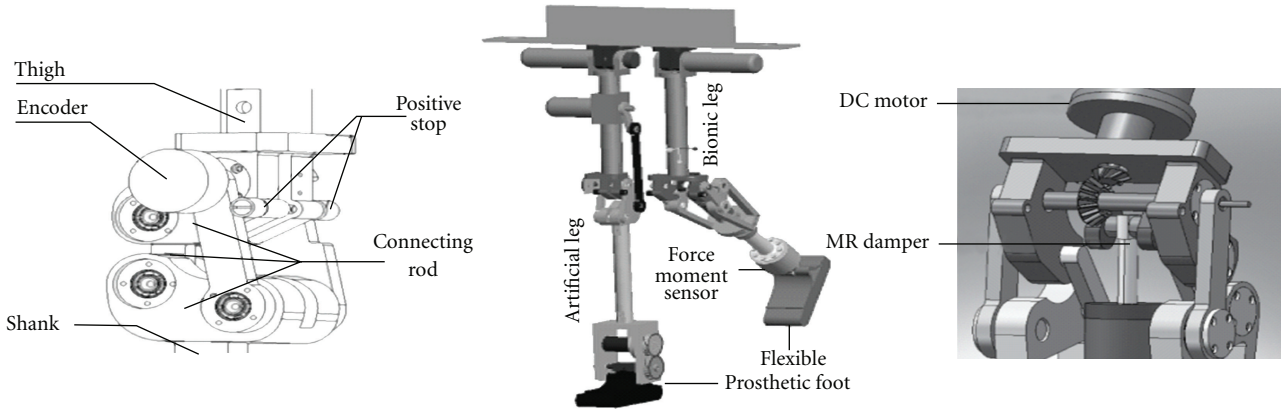


Figure 1: The test-bed in heterogeneous configuration and 4-bar knee joint mechanism.

In this architecture, the master is the sound leg of amputee, corresponds to AL in the test-bed. It generates the active motion that is planned by linear inverted pendulum model combining with ZMP stability criterion. The desired joint angle trajectory is then calculated by inverse kinematics of AL. The slave, however, is prosthetic leg, corresponds to the BL in the test-bed. It perceives the motion intention of the master, and controls itself making gait tracking to realize the coordination to the master.

2. Heterogeneous Biped Robot

Figure 1 shows the prototype of the test-bed in heterogeneous configuration that is mainly made up of two legs, one is AL used to simulate the sound leg of amputee and another is BL used to simulate stump with prosthetic leg. The two agents are heterogeneous at the aspect of mechanism, actuation manner, sensing capacity, and dynamic characteristics.

To realize humanoid gait, both legs are designed and fabricated with 4-bar closed-chain knee joints and flexible prosthetic foot. Joints in AL are actuated by Maxon DC servo-motors, which can realize arbitrary gait. The knee joint of BL are driven by a hybrid actuation system combing magneto-rheological (MR) damper augmented with a DC motor. Hybrid actuation manner greatly improves the mobility and environmental adaptability at the cost of a small rise in energy consumption.

3. Dual-Leg Coordination

3.1. Overview of Dual-Leg Coordinated Control. Dual-arm and multileg coordination are hot topics in the field of multirobot system research [5–7]. Dual-leg coordination belongs to the stable walking control of biped robot. In the past studies, however, the two legs were usually thought as a bifurcate mechanism of a single biped robot, the gait for the two legs are planned concurrently. Status information and command for both legs can be exchanged directly without the need of perception and the two legs are controlled as a whole by one locomotion controller. The concepts and methods of dual-leg coordination were not explicitly proposed for single biped robot control before.

Though quite similar to the dual-arm coordination at the aspect of system task, control principle and implementation method, dual-leg coordination is more difficult due to its complicated constrains. Table 1 shows the comparison for the two kinds of coordination.

According to control strategy, coordination methods of dual-arm and multileg can be roughly divided into two categories, named master/slave method and object-oriented method separately [8]. The former studies the motion tracking of slave arm (leg) to master arm (leg) by satisfying the constrain conditions of kinematics and dynamics; while the latter studies the tracking of desired motion trajectory and/or force trajectory for task object without considering the details of two arms (legs). For the simplicity of the master/slave method, the coordination capacity and adaptability is limited by the master/slave relations. Object-oriented method accords with the essence of human dual-arm (leg) coordination, however, control algorithm is comparatively complicated. The prosthetic leg has limitations at the aspect of intelligence and maneuverability. Therefore, master/slave strategy is suitable to AP coupling system.

According to task, dual-arm coordination can be divided into convey coordination and assemblage coordination. For biped robot, the system task can be described as: the two legs support the HAT, simultaneously, moving it smoothly through coordination to achieve stable walking of the whole biped locomotive system. Therefore, dual-leg coordination can be thought as a special form of convey coordination. Although the differences in terms of mechanism and actuation manner for the two legs, master/slave coordination method can be used in heterogeneous biped robot for the same basic configurations (e.g., length of leg, motion range of joints, etc.) and similar gait pattern for both legs. In dual-leg coordination system, AL is defined as master leg and BL as slave one. A coordination module is usually constructed and used to control the pose and position of interfaces between the HAT and the two legs to satisfy the constrain conditions of kinematics and dynamics. Therefore, motion planning, control, and compensation of the coordination

module become key problems in coordinated control system research for AP coupling system.

3.2. Coordinated Control Architecture. For the complicated control task, the distributed hierarchical control architecture is used to realize the master/slave coordination for AP coupling system. Hierarchical architecture [9] is an advanced control structure in which the complicated task is decomposed into several levels according to the intelligent grade. For different levels, a certain control strategy and implementation method is used and the complicated task can be accomplished by all levels through interrelation and coordination. By using hierarchical control architecture, fault debugging and system implementation can be easily achieved.

According to master/slave relation, two independent hierarchical control systems were built for both legs as shown in Figure 2. Either consists of three levels, named task layer, plan layer and drive layer separately.

In the hierarchical control architecture of AL, the task layer has the highest intelligent grade and decides what to do according to environment information or human command. As a master leg, AL decides the task of the whole system, which usually includes level-walking at different speed, walking on stairs and slopes, running, emergency acts, and so forth. Plan layer has the middle intelligent grade and is used to plan the gait to realize the desired task. In plan layer, the kinematics and dynamics equations are solved for gait control purpose, also the planned gait is verified whether satisfying constrain conditions of the coordination model. Drive layer has the lowest intelligent grade, in which DC motors actuate joints to produce planned gait based on Fuzzy-PID feed-back control algorithm.

As a slave leg, the task of BL is to track the gait of AL to realize gait synergy. The task layer of BL is used to realize the functions of gait measurement, gait recognition, and gait estimation. In plan layer, a major work is to calculate the desired damper force and motor torque to realize gait tracking. In drive layer, a MR damper augmented by DC motor actuates the knee joint of BL according to the optimized damper force and motor torque obtained in plan layer, and a PD feed-back control algorithm is applied on the MR damper and Fuzzy-PID feed-back control algorithm is applied on the DC motor.

4. Gait Planning for Artificial Leg

4.1. Modeling. Figure 3 shows the mass and geometry models of test-bed in homogeneous configuration.

It is modeled as a system of 11 material points. Each link has its weight at the position of its CoG. The trunk is counted as the base link. In this research, however, it is assumed that the robot is held in 2D space. The movement of the robot is considered only in sagittal plane. Therefore, it has 6 DOF for gait planning.

4.1.1. Kinematics. First, direct kinematics of AL in swing phase is established in the coordinate system $\sum XYZ$ and the matrix of transformation from torso to ankle is expressed in (1).

4-bar mechanism in knee joint of AL brought the closed-chain geometrical constrain with kinematics that is expressed in (2), which makes two of the angle variables in $\theta_2^a, \theta_3^a, \theta_4^a$, and θ_5^a dependent.

$$A^a = \begin{bmatrix} x_a^a \\ 0 \\ z_a^a \end{bmatrix} = \begin{bmatrix} -l_3^a \sin\theta_3^a - l_2^a \sin\theta_2^a - l_5^a \sin\theta_5^a - l_{11}^a \sin(\alpha - \theta_2^a) - l_{12}^a \sin(\beta - \theta_5^a) + x_h^a \\ 0 \\ -l_3^a \cos\theta_3^a - l_2^a \cos\theta_2^a - l_5^a \cos\theta_5^a + l_{11}^a \cos(\alpha - \theta_2^a) + l_{12}^a \cos(\beta - \theta_5^a) + z_h^a \end{bmatrix}, \tag{1}$$

$$\begin{aligned} f_1 : {} & (l_{11}^a - l_9^a)\cos(\theta_2^a - \alpha) + l_{10}^a \cos(\theta_5^a - \beta) \\ & - l_3^a \cos\theta_3^a + l_4^a \cos\theta_4^a = 0, \\ f_2 : {} & (l_{11}^a - l_9^a)\sin(\theta_2^a - \alpha) + l_{10}^a \sin(\theta_5^a - \beta) \\ & - l_3^a \sin\theta_3^a + l_4^a \sin\theta_4^a = 0. \end{aligned} \tag{2}$$

4.1.2. Dynamics. In swing phase, hip joint makes translational movement along a certain trajectory; in the meanwhile thigh and shank make rotational movement. If the generalized coordinate $\theta = [x_{1c} \;\; z_{1c} \;\; \theta_2^a \;\; \theta_3^a \;\; \theta_4^a \;\; \theta_5^a \;\; \theta_6^a]^T$ that θ_4^a and θ_5^a are the dependent variables, then the Lagrange function can be formed as

$$L_{sw}^a = \sum_{i=2}^{6} \left[\frac{1}{2} m_i^a (\dot{x}_{ic}^2 + \dot{z}_{ic}^2) + \frac{1}{2} J_i \dot{\theta}_i^{a^2} - m_i^a g z_{ic} \right]. \tag{3}$$

The drive forces of robot dynamic system are F_1^x, $F_1^z, T_2^a, T_3^a, T_4^a, T_5^a$, and T_6^a. According to principle of virtual work, the virtual work of drive force is calculated by

$$\begin{aligned} \delta A = {} & F_1^x \delta x_{1c} + F_1^z \delta z_{1c} + (T_2^a - T_3^a - T_4^a)\theta_2^a + (T_3^a - T_5^a)\theta_3^a \\ & + (T_4^a - T_5^a)\theta_4^a + (T_5^a - T_6^a)\theta_5^a + T_6^a \theta_6^a. \end{aligned} \tag{4}$$

Then the generalized force is

$$\begin{aligned} Q = [& F_1^x \quad F_1^z \quad T_2^a - T_3^a - T_4^a \quad T_3^a \\ & -T_5^a \quad T_4^a - T_5^a \quad T_5^a - T_6^a \quad T_6^a], \end{aligned} \tag{5}$$

where F_1^x and F_1^z are the forces supplied by the HAT; T_4^a and T_5^a are the redundancy drive force and are set to 0. Then the Lagrange equation of the first kind is deduced as

$$M_{sw}^a(\theta^a)\ddot{\theta}^a + C_{sw}^a(\theta^a)\dot{\theta}^{a^2} + G_{sw}^a(\theta^a) = B_{sw}^a T^a + \lambda \dot{f}, \tag{6}$$

Table 1: Comparison between dual-arm and dual-leg coordination.

Items	Dual-arm coordination	Dual-leg coordination
System task	Grasping and holding object to track the target trajectory	Supporting and moving the HAT of biped robot to realize stable walking
Methods	Planning, control, and compensation of coordinated motion	
Controlled object	Homogeneous arms	Heterogeneous legs
Target object	Grasped and held object	The HAT (Head, Arms, and Torso)
Base	Fixed	Mobile
Control complexity	Relative simple	Complicated
Interface	Multiple fingers	Hip joints
Target trajectory	Task trajectory of grasped and held object	Gait trajectory for the center of HAT or the center of crotch
Mechanism	Closed chain all the time	Closed chain in stance phase Open chain in swing phase
Constrains	Kinematics and dynamics	Dynamic constrains for open chain and both for closed chain
Motion pattern	Different for two arms	Similar for two legs (phase-difference)

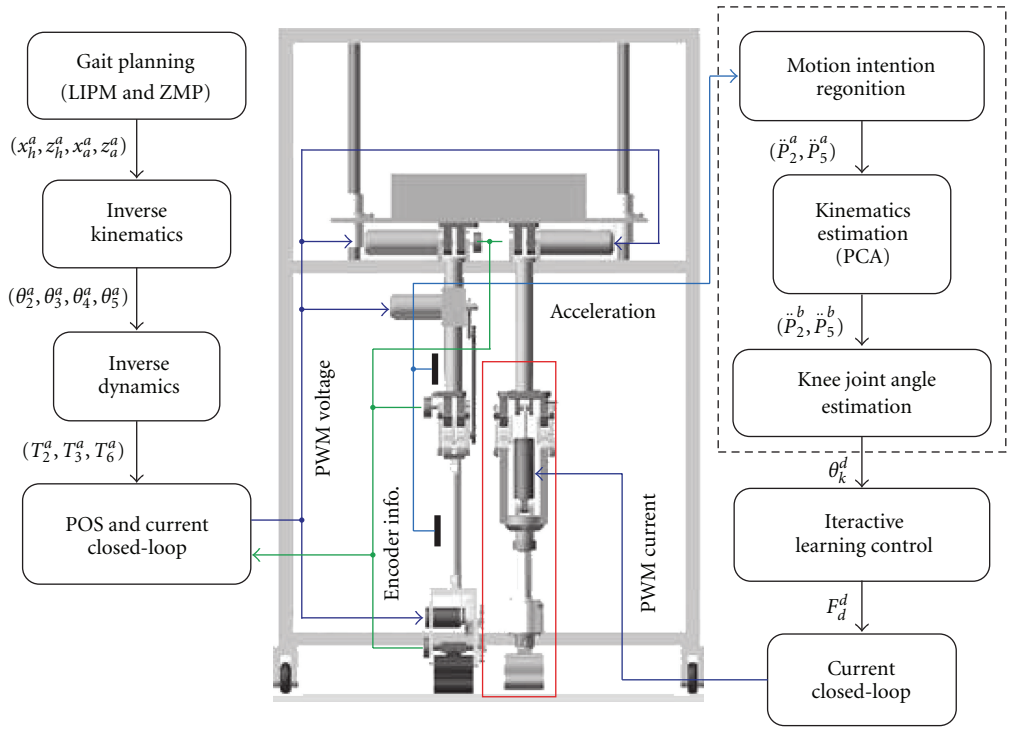


Figure 2: Coordinated control architecture.

where M_{sw}^a is a symmetric matrix called inertial matrix, C_{sw}^a is an antisymmetric matrix named centrifugal force or Coriolis force matrix, G_{sw}^a is matrix of gravitational forces, B is coefficient matrix of applied torques, and $\lambda \dot{f}$ represents constrain torque.

4.2. Gait Planning

4.2.1. ZMP Theory. One of the criterions for estimating the stability of the walking is a ZMP (Zero Moment Point) criterion proposed by Vukobratovic [10]. ZMP is a point on

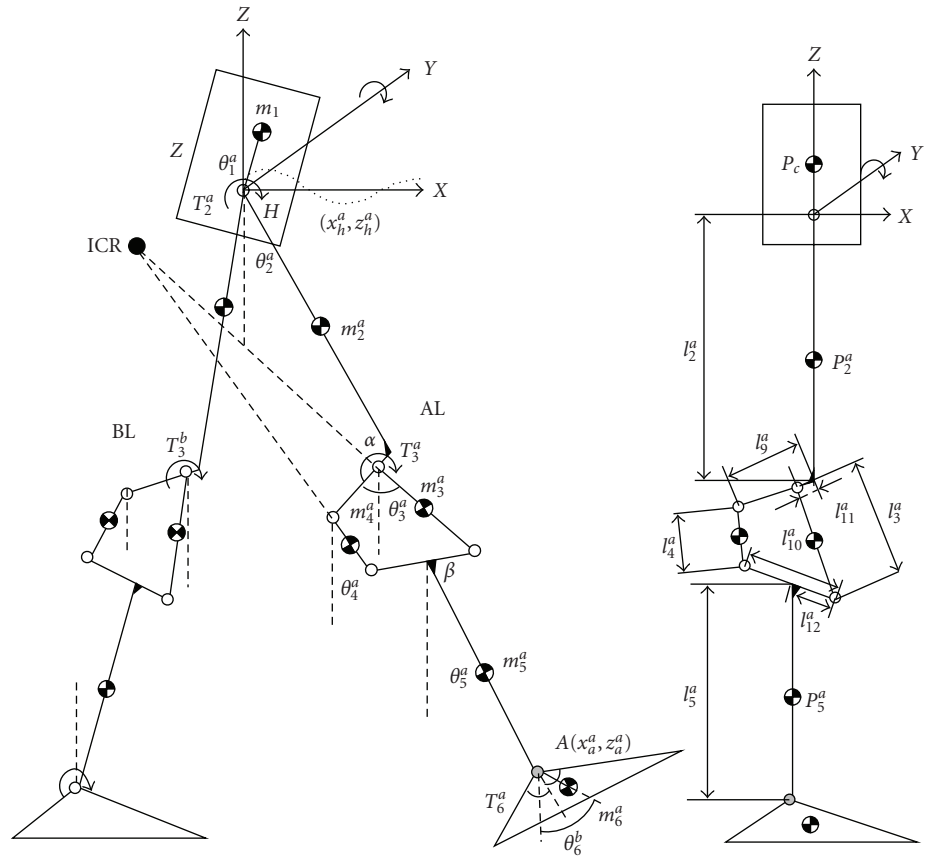


Figure 3: Mass and geometry models.

the surface about which the sum of all moments of active forces, momentums are equal to zero. The calculation of ZMP is described as follows

$$x_{\text{zmp}} = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)x_i - \sum_{i=1}^{n} m_i\ddot{x}_i z_i + \sum_{i=1}^{n} M_{iy}}{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)},$$
$$y_{\text{zmp}} = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)y_i - \sum_{i=1}^{n} m_i\ddot{y}_i z_i + \sum_{i=1}^{n} M_{ix}}{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)}, \tag{7}$$

where m_i is the mass, (x_i, y_i, z_i) is the Cartesian coordinate of CoG of i^{th} link, $(x_{\text{zmp}}, y_{\text{zmp}})$ is the Cartesian coordinate of ZMP.

While a biped robot is standing still or walking very slowly, a projective point of the CoG on the surface is in the polygon made from the contact points on the sole. In the case, the robot is in the statically stable state and keeps on walking without a tumble. However, while the body and/or the legs are moving fast, various accelerations due to motions are produced, and the projective point of the center of gravity is getting out of the polygon. ZMP is like a projective point of the CoG in the standing still state or the slow-walking state of the robot while the robot is in the dynamic-walking state. According to the ZMP criterion, the robot may keep walking without tumbling. In this study, based on the ZMP criterion, the gait planning for AL in the sagittal plane is designed.

4.2.2. Linear Inverted Pendulum Model. Bipedal walking system is complex, nonlinear, and naturally unstable. Linear inverted pendulum model (LIMP) theory provides an alternative approach to generate the gait trajectory of bipedal walking. Figure 4 shows the LIMP model and CoG transform model in a sagittal plane.

Where f is the stretching force imposed on the stick. τ is the moment generated by ground reaction force, θ is the angle of stick inclination (the clockwise direction is positive), M is the concentrated mass representing the HAT, and r is the length of the stick. The mass of the stick is neglected.

By using Lagrangian function approach, the dynamics model of LIPM can be deduced by

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = Q,$$
$$L = T - V = \frac{1}{2}mr^2\dot{\theta}^2 + \frac{1}{2}m\dot{r}^2 - mgr\cos\theta, \tag{8}$$
$$q = [\theta r]^T, \quad Q = [f\tau]^T,$$

where L is Lagrangian function, q is generalized coordinate and Q is the generalized force. Then the dynamics function of LIMP is expressed as

$$mr^2\ddot{\theta}^2 + 2mr\dot{r}\dot{\theta} + mgr\sin\theta = \tau,$$
$$m\ddot{r} - mr\dot{\theta}^2 + mg\cos = f. \tag{9}$$

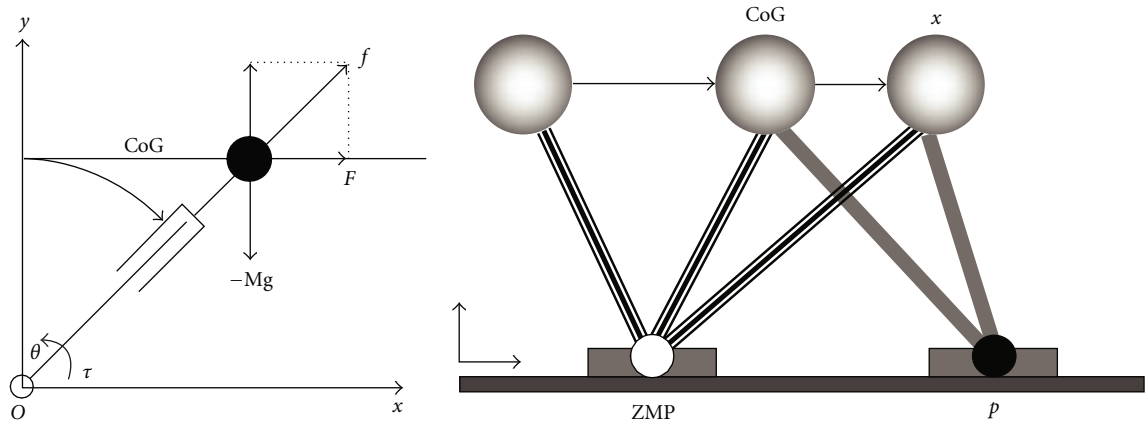


Figure 4: LIMP model and CoG transform model in sagittal plane.

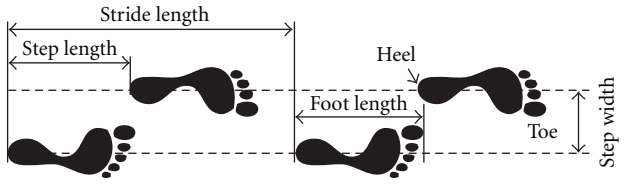


Figure 5: Gait spatial features.

In order to keep the movement of the CoG in the horizontal direction and get an acceleration motion, the stretching force f and mass m must satisfy the relation as

$$f\cos\theta = mg,$$
$$f\sin\theta = m\ddot{x}. \tag{10}$$

Solving this equation can get

$$x - p = \frac{g}{z}\ddot{x}, \tag{11}$$

where x is the length of the projection of the CoG in x-axis, and z is the height of CoG in z-axis, p is the projection of ZMP in x-axis.

4.2.3. Gait Planning for AL. The gait space features of human walking are shown in Figure 5. By set time-spatial features of AL as shown in Table 2, the corresponding ZMP trajectory can be obtained as shown in Figure 6.

In LIMP, the relation between ZMP and CoG is expressed

$$\ddot{x} = \frac{g}{z}\left(x - x_{\text{zmp}}\right),$$
$$\ddot{y} = \frac{g}{z}\left(y - y_{\text{zmp}}\right). \tag{12}$$

The trajectories of ZMP was planned as mentioned above, then the trajectories of CoG can be calculated by (12), the ideal trajectories of ZMP and CoG in X and Y axial directions are shown in Figure 7.

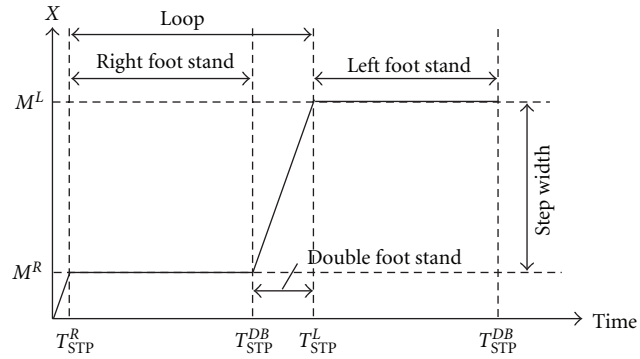


Figure 6: Ideal ZMP trajectories.

Table 2: Gait parameters of AL.

Item	Value
Thigh length	0.46 m
Shank length	0.48 m
Foot height	0.10 m
Foot width	0.07 m
Foot length	0.25 m
Horizontal distance between ankle joint and heel	0.055 m
Gait velocity	95~125 step/min
Step length	0.20~0.50 m
Step width	0.10 m
Step height	0.05~0.10 m
Stride length	0.75~0.83 m
Gait cycle	1.20 s

Gait in Cartesian space is intuitive, easy to describe, and good to reflect the relation between robot system and ground. However, the movement of artificial leg is achieved by the rotation of joints. Therefore, the solving of inverse kinematics is needed to map the variables from

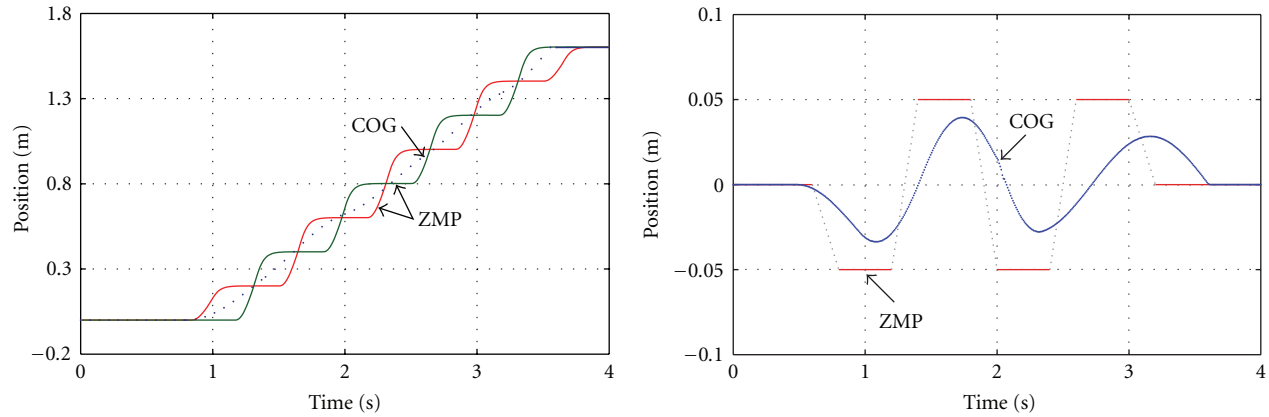


Figure 7: ZMP and CoG trajectories in X and Y axial directions.

Cartesian space to joint space. In this research, the Newton-Raphson algorithm is used to calculate the joint angles of AL. Figure 8 shows the desired angles of hip joints of both leg, angles of knee and ankle joints of AL separately, which are calculated through inverse kinematics. In general, above-knee amputee always has an intact hip, so that there is only an angular phase difference between hips of sound leg and stump. The ankle of BL is fixed without the need to gait plan. The angle of knee joint in BL should be estimated based on interjoint coordination described in the following section.

5. Gait Recognition and Estimation for Bionic Leg

Although amputee and his/her artificial limb (prosthesis) can exchange information directly through electromyography (EMG) or biological neural signal, EMG and neural signal-based prosthesis techniques are still premature. For common prosthesis, movement intention of amputee can only be perceived by its own perception system. In this research, embedded 3-axial attitude sensor can measure kinematic information of the thigh that reflects the amputee's posture and walking conditions. Based on these measured data, gait pattern of BL can be estimated by tuned gait classifier so that MR damper can drive the knee joint to coordinate with the sound leg of amputee to realize symmetrical stable walking.

In recent years, inertial motion capture has emerged as one of the most versatile methods of full-body ambulation measurement. By using sensor fusion of three-axis gyroscopes and three-axis accelerometers, inertial sensors accurately measure the orientation and position of body segments in a global coordinate system in this research. The nature of the sensor technology used allows inertial motion capture systems to overcome some of the most pressing limitations found in alternative methods such as optical, mechanical, acoustic, and magnetic motion capture.

5.1. Overview of Principal Component Analysis. Principal component analysis (PCA) is a general approach to compression and dimensionality reduction for mass data based on multivariable statistical analysis. The basic idea of PCA is to attempt to efficiently represent the data by decomposing a data space into a linear combination of a small collection of bases consisting of orthogonals that maximally preserves the variance in the data. If n measurements of m-tuple $X^T = (x_1, x_2, \ldots, x_m)$ represent original data with linear correlations, then the jth principal component (PC) can be expressed as

$$P_j = a_{j1}x_1 + a_{j1}x_2 + \cdots + a_{jm}x_m \sum_i a_{ji}^2 = 1. \quad (13)$$

The coefficients $a_{ji}(i = 1, 2, \ldots, m)$ are called the factor loadings. The magnitude indicates the amount of variation in variable x_i that is captured by the principal component P_j and the sign indicates the nature of correlation between P_j and x_i. For a given data set, PCA produces a unique solution. The common procedure of PCA can be summarized as the following 4 steps.

(1) *Normalization of Original Data.* In order to perform PCA, the data need to have zero mean as well as stand deviation of 1 and the normalization can be achieved by

$$x_{ij}^* = \frac{x_{ij} - \mu_i}{\sigma_i},$$
$$\mu_i = \frac{\sum_{j=1}^n x_{ij}}{n}, \quad (14)$$
$$\sigma_i = \sqrt{\frac{\sum_{j=1}^n \left(x_{ij} - \mu_j\right)}{(n-1)}},$$

where $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$.

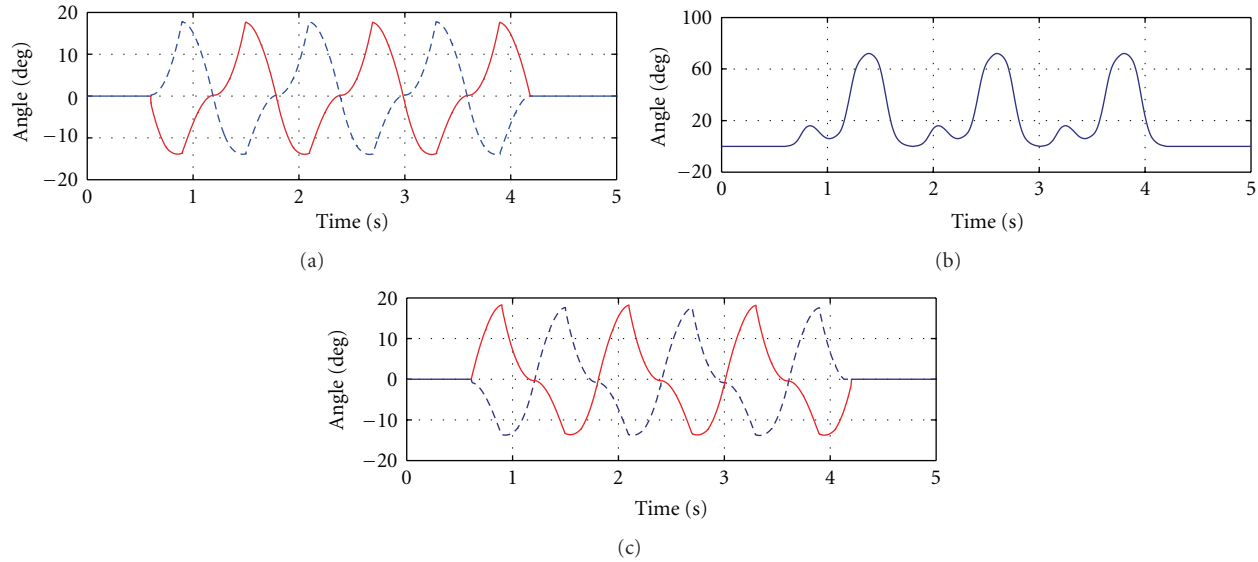


Figure 8: Desired angles of hip joint, knee joint, and ankle joint.

(2) *Calculation of Covariance Matrix R.* We have

$$R = \frac{1}{N-1} X^{*T} X^{*}, \tag{15}$$

where X^{*} is the normalized matrix of X.

(3) *Determination of the Number of PC.* The eigenvectors are obtained algebraically through decomposition of the covariance matrix R of the original data. If the eigenvalue λ_i $(i = 1, 2, ..., m)$ and eigenvector γ_i $(i = 1, 2, \ldots, m)$ are given, the number of PC p can be determined by

$$\eta_i = \frac{\lambda_i}{\sum_{i=1}^{m} \lambda_i}, \qquad \epsilon(p) = \sum_{i=1}^{p} \eta_i, \tag{16}$$

where p satisfies $\epsilon(p) \geq 85\% \sim 90\%$.

(4) *Determination of Transformation matrix* Γ. The transformation matrix Γ formed by the p eigenvectors of matrix R sorted in descending order of the corresponding eigenvalue

$$\Gamma = \left(\gamma_1, \gamma_2, \ldots \gamma_p\right) \tag{17}$$

which maps X on the new coordinates P with

$$P = \Gamma^{T} X. \tag{18}$$

5.2. Reconstruction of BL's Kinematics Using AL's Kinematics. In human gait, it has been observed that joint angle trajectories show strong linear correlations [11–13], which indicates that a compressed gait data set can be obtained by using PCA. If the original gait data X is acquired by sensors and preprocessed, the orthogonal unit eigenvector γ_i $(i = 1, 2, \ldots, m)$ and transformation matrix $R = (\gamma_1, \gamma_2, \ldots, \gamma_m)$ can be obtained and the new gait variables can be expressed as

$$Y = R^{T} X. \tag{19}$$

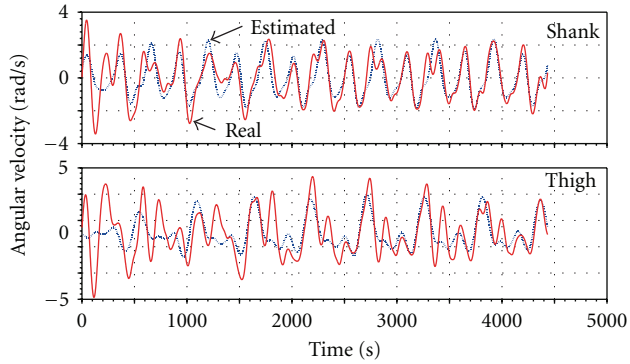


Figure 9: Motion estimation of IBL.

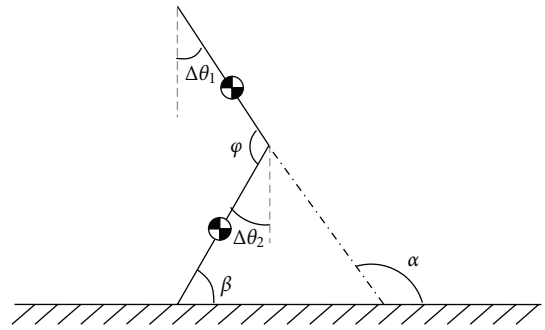


Figure 10: Lower limb joints geometry.

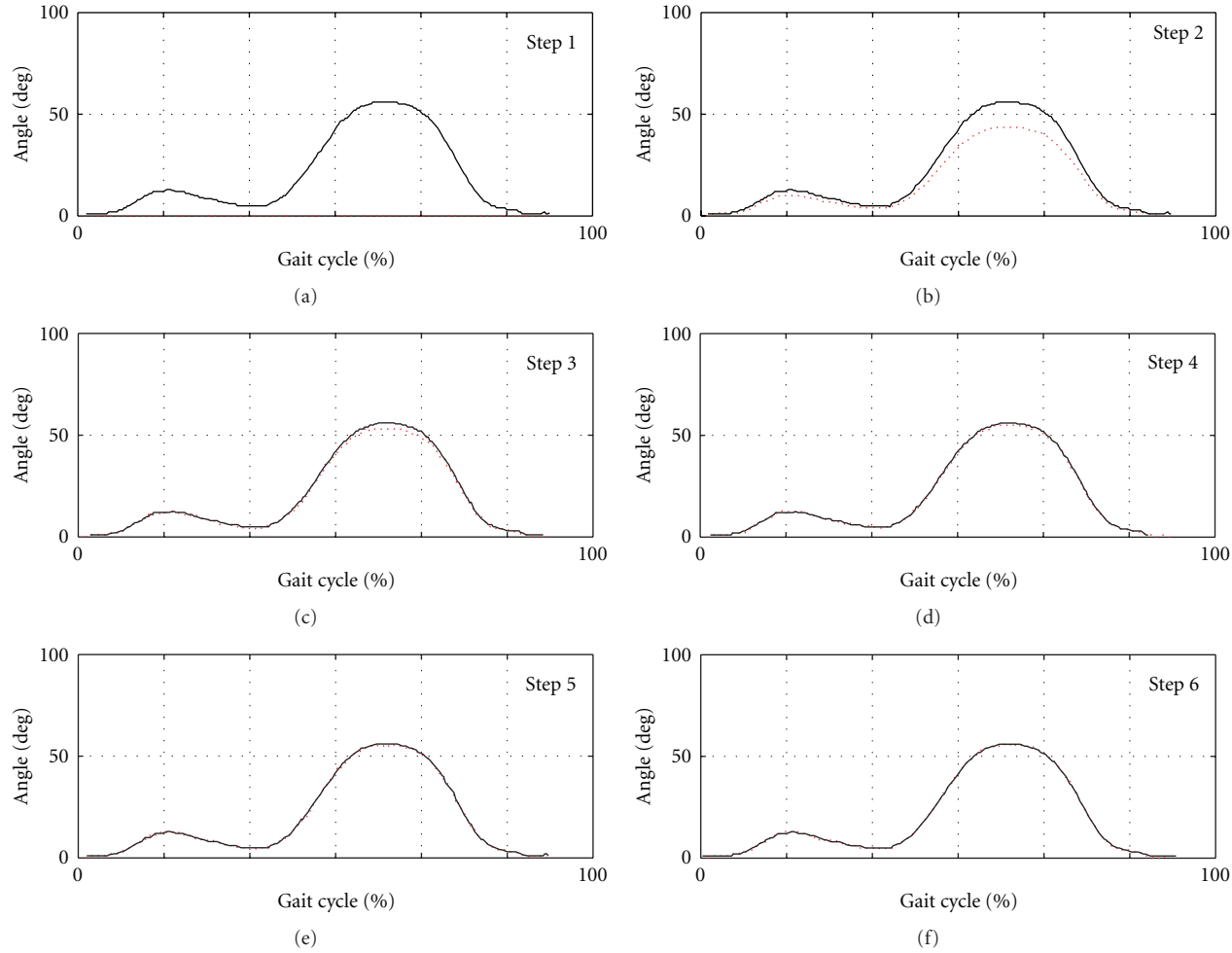


Figure 11: Tracking curve of IBL knee joint relative angle with P-type open/closed-loop ILC.

Since R is orthogonal matrix, X can be reconstructed from Y by

$$X = RY. \tag{20}$$

If p is determined, then the equation can be rewritten as

$$X = \Gamma Y^*, \tag{21}$$

where $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_p)$, Y^* is the top p rows of Y. Neglecting the last components of Y, the data is reduced in a way that the least information is lost. A reconstruction of X from the lower dimensional coordinate Y^*, leads to a least-square optimal fit of the original data. PCA cannot only be used for compression, but also for reconstruction of incomplete measurements.

Assuming that the first components of $X_1 \in \mathbb{R}^{(m-l)}$ of X are known, and the remaining part $X_2 \in \mathbb{R}^{(l)}$ is unknown, (21) is separated into

$$X_1 = \Gamma_1 Y^*, \quad X_2 = \Gamma_2 Y^*, \tag{22}$$

where $\Gamma_1 \in \mathbb{R}^{(m-l)\times p}$ and $\Gamma_2 \in \mathbb{R}^{l\times p}$ are the sub matrix of Γ. Thus X_2 is reconstructed from X_1 by

$$X_2 = \Gamma_2 \left(\Gamma_1{}^T \Gamma_1\right)^{-1} \Gamma_1{}^T X_1. \tag{23}$$

For AP coupling system, the observable data is the motion of sound leg, and the missing data is the motion of BL. To obtain the covariance matrix R, the test-bed was firstly configured as homogeneous form, CoGs of thigh and shank of two identical ALs were mounted with rate gyroscopes, and walking experiments were conducted with various cadences and stride length. The acquired data from gyroscopes were treated by following the procedures described above and the covariance matrix was obtained in the final. Then the test-bed is reconfigured as heterogeneous form and a pair of rate gyroscope mounted on the CoGs of thigh and shank was used to measure the motion of AL. By employing PCA, the motion of BL is estimated as shown in Figure 9. Solid line represents real angular velocities obtained by gyroscopes mounted on AL, while dashed line represents estimated angular velocities of BL calculated through (23).

By integrating angular velocities of thigh and shank calculated above, the angles $\Delta\theta_1$ and $\Delta\theta_2$ can be obtained.

Figure 12: Step sequence in swing phase.

Then the angle between the thigh and ground α and the angle between the shank and ground β can be calculated according to the geometric relation as shown in Figure 10

$$\alpha = 90 + \Delta\theta_1,$$
$$\beta = 90 - \Delta\theta_2. \tag{24}$$

Then the angle of knee joint of BL can be estimated by

$$\varphi = 180 - \alpha + \beta. \tag{25}$$

6. Gait Tracking for Bionic Leg

Gait tracking control is the key to realize stable walking of AP coupling system, which belongs to time-varying trajectory tracking problem. In the heterogeneous configuration of test-bed, MR damper and electro-motors are used to actuate the joints tracking the ideal trajectories predefined in gait planning strategy. Due to the complicated dynamical characters of nonlinear, time-varying, strong coupling, and the need of high precise repetitive motion, gait tracking control is a hard difficulty in AP coupling research especially for BL that is hybrid-controlled by MR damper augmented with electro-motor. Although the methods, such as nonlinear decoupling control, decomposition feedback control, adaptive control, computed torque control, and so forth, were widely studied and proved able to solve time-varying trajectory tracking problem effectively, complex algorithm, time-consuming computation, and the need of precise and accurate system model limit the use of these methods in gait tracking control of AP coupling system. Iterative learning control (ILC) [14] can meet the special control requirement of gait tracking.

ILC can be stated as follows.It is due to repetitive motion that initial state is the same with the end $\theta_0(0)$ using preceding control input and actual output error, to find a best control input which will track the reference trajectories in limited time, that is, $\theta_k(t) \to \theta^d(t)$; $u_k(t) \to u^d(t)$. $\theta^d(t)$ and $u^d(t)$ are the desired output and control input, and k is the number of iterative learning steps. In this paper, closed-loop ILC is combined with open loop ILC and the first order P-type open/closed-loop ILC is proposed. The control law is given by:

$$u_{k+1}(i) = u_k(i) + \Gamma e_{k+1}(i) + \Gamma' e_k(i), \quad k = 1, 2, \ldots, \tag{26}$$

where Γ and Γ' are the learning gains, $e_k(i) = \theta^d(t) - \theta(t)$ is the tracking error. Initial state, that is, $\theta_0(0) = \theta^k(0)$, must be satisfied in learning phase.

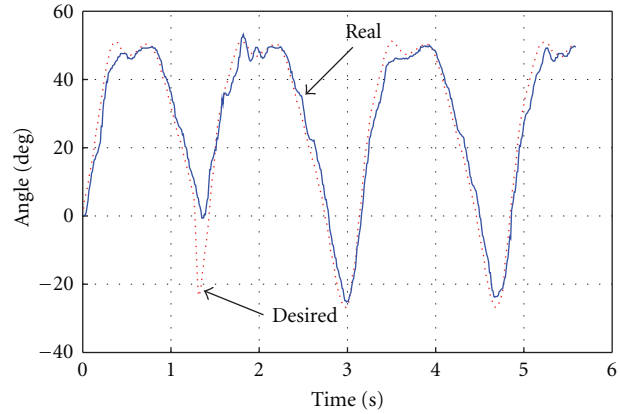


Figure 13: Real and desired hip joint angle of artificial leg during swing.

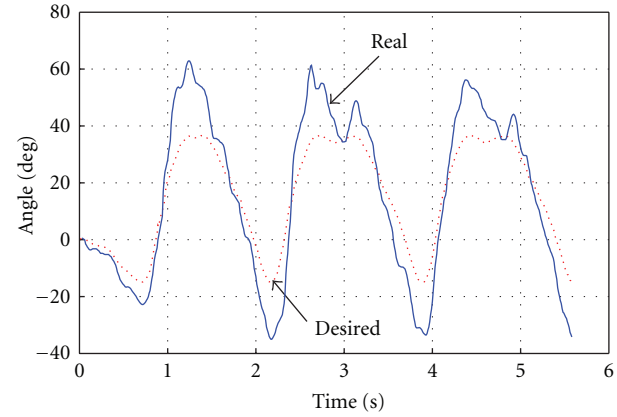


Figure 14: Real and desired hip joint angle of bionic leg during swing.

7. Experimental Results

To validate the effectiveness of the proposed coordinated control strategy for AP interaction, the virtual prototyping-based collaborative simulation and physical prototype experiments for stable walking control for biped robot were conduct.

Virtual prototype of test-bed is established using software ADAMS. Control module is built in MATLAB/Simulink. The sampling interval is 0.02 s and simulation time is 1.6 s. Relative knee joint angle tracking of BL to AL is illustrated

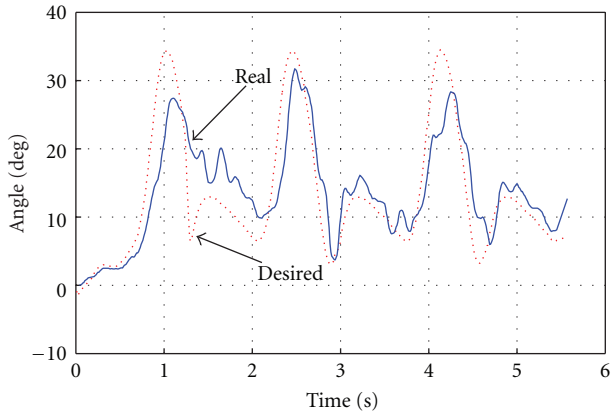


Figure 15: Real and desired knee joint angle of artificial leg during swing.

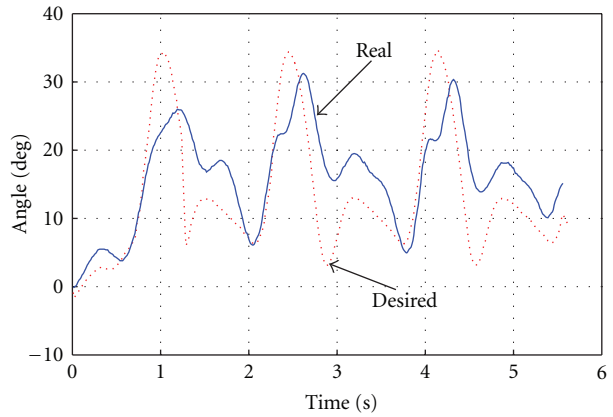


Figure 16: Real and desired knee joint angle of bionic leg during swing.

in Figure 11. The solid line represents reference knee joint angle estimated by method introduced in Section 5, while the dashed line represents the real output of ILC controller. From the figure, it can be seen that the learning was terminated after 6-step iterations when the error criterion was satisfied.

The simulation result indicates that BL controlled by hybrid actuation system can track the gait of AL well.

Figure 12 shows the gait tracking control in swing. Figures 13, 14, 15, and 16 shows the desired and real joint angles during experiment. The results shows that P-type open/close-loop ILC control scheme is effective and can insure the BL tracking gait of AL to realize stable walking of the whole system.

8. Conclusion

This paper describes the coordinated control for AP coupling system to realize symmetrical stable walking. The proposed scheme consists of walking gait synthesis, motion intention recognition, and gait tracking. Simulation and experimental results demonstrate that the proposed control scheme is correct and reasonable. The master/slave dual-leg coordinated control strategy is suitable for complex AP coupling system.

Acknowledgments

This work was supported by the Nature Science Foundation of China (Grant no. 60705031), by Specialized Research Fund for the Doctoral Program of Higher Education (Grant no. 20070145105), by State Key Laboratory of Robotics and System (HIT) (Grant no. SKLRS200711), and by Fundamental Research Funds for the Central Universities (Grant no. N090404007).

References

[1] H. Herr and A. Wilkenfeld, "User-adaptive control of a magneto-rheological prosthetic knee," *Industrial Robot*, vol. 30, no. 1, pp. 42–55, 2003.

[2] F. Wang, C. D. Wu, Y. Z. Zhang, and X. H. Xu, "Virtual prototyping based collaborative simulation of biped robot with heterogeneous legs," in *Proceedings of the 13th International Conference on Advanced Robotics*, pp. 874–879, 2007.

[3] H. Lim, A. Ishii, and A. Takanishi, "Basic emotional walking using a biped humanoid robot," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, pp. 954–959, 1999.

[4] S. Collins, A. Ruina, R. Tedrake, and M. Wisse, "Efficient bipedal robots based on passive-dynamic walkers," *Science*, vol. 307, no. 5712, pp. 1082–1085, 2005.

[5] N. Xi and T.-J. Tarn, "Sensor-referenced multiple robot cooperation for material handling," *Industrial Robot*, vol. 25, no. 2, pp. 129–133, 1998.

[6] H. G. Cai, H. Liu, and J. W. Li, "A survey of multi-fingered robot hands: issues of coordination and autonomous grasp," *Robot*, vol. 22, no. 4, pp. 319–328, 2000.

[7] V. Lippiello, L. Villani, and B. Siciliano, "An open architecture for sensory feedback control of a dual-arm industrial robotic cell," *Industrial Robot*, vol. 34, no. 1, pp. 46–53, 2007.

[8] C. Guodong, C. Wenshan, Z. Peng, and C. Jing, "Hybrid position/force control for dual-arm symmetric coordination-model algorithm and realization," *Acta Automatica Sinica*, vol. 22, no. 4, pp. 418–427, 1996.

[9] L. Jun, Z. Xinglong, and Y. Jingping, "Hierarchical control of dual-arm robot parts mate," *Manufacturing Automation*, vol. 24, no. 6, pp. 22–25, 2002.

[10] M. Vukobratovic, B. Borovac, and V. Potkonjak, "ZMP: a review of some basic misunderstandings," *International Journal of Humanoid Robotics*, vol. 3, no. 2, pp. 153–175, 2006.

[11] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa, "ZMP analysis for ARM/Leg coordination," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, vol. 1, pp. 75–81, 2003.

[12] A. Alexandrov, A. Frolov, and J. Massion, "Axial synergies during human upper trunk bending," *Experimental Brain Research*, vol. 118, no. 2, pp. 210–220, 1998.

[13] N. St-Onge and A. G. Feldman, "Interjoint coordination in lower limbs during different movements in humans," *Experimental Brain Research*, vol. 148, no. 2, pp. 139–149, 2003.

[14] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robotics by learning," *Journal of Robotic Systems*, vol. 1, no. 2, pp. 123–140, 1984.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 841286, 10 pages
doi:10.1155/2010/841286

Research Article

Evolving Neural Network Controllers for a Team of Self-Organizing Robots

István Fehérvári and Wilfried Elmenreich

Mobile Systems Group/Lakeside Labs, Institute for Networked and Embedded Systems, University of Klagenfurt, 9020 Klagenfurt, Austria

Correspondence should be addressed to István Fehérvári, istvan.fehervari@uni-klu.ac.at

Received 2 December 2009; Accepted 25 March 2010

Academic Editor: Noriyasu Homma

Self-organizing systems obtain a global system behavior via typically simple local interactions among a number of components or agents, respectively. The emergent service often displays properties like adaptability, robustness, and scalability, which makes the self-organizing paradigm interesting for technical applications like cooperative autonomous robots. The behavior for the local interactions is usually simple, but it is often difficult to define the right set of interaction rules in order to achieve a desired global behavior. In this paper, we describe a novel design approach using an evolutionary algorithm and artificial neural networks to automatize the part of the design process that requires most of the effort. A simulated robot soccer game was implemented to test and evaluate the proposed method. A new approach in evolving competitive behavior is also introduced using Swiss System instead of the full tournament to cut down the number of necessary simulations.

1. Introduction

The concept of systems consisting of multiple autonomous mobile robots is attractive for several reasons [1]. Multiple cooperative robots might be able to achieve a task with better performance or with lower cost. Moreover, loosely coupled distributed systems tend to be more robust, yet more flexible than a single powerful robot performing the same task. A benefit of the collaborative interaction of mobile robots can be an emergent service, that is, a progressive result that is more than the sum of the individual efforts [2]. A swarm of robots can thus build a self-organizing system [3].

The continuous technical development in robotics during the last decades has provided us with the hardware for swarms of small, cheap autonomous devices [4–6]. However, designing the behavior and interactions among the robots remains a very complex task. Using a standard top-down design approach with fixed task decompositions and allocation typically leads to systems working only for a small set of parameters. On the other hand, effects like changing environments or breakdowns and faults of hardware require a robust and flexible solution that provides a useful service for many possible system states.

An alternative to the classical design approach is to organize the robots as a self-organizing system performing the intended task. Thus, the robots achieve a global system behavior via simple local interactions without centralized control [7]. As shown by many examples in nature, simple rules for interactions can emerge to quite complex behavior while being scalable and robust against disturbances and failures. This would allow for simple control systems like for example having a small ANN on the particular robots.

Unfortunately, there is yet no straightforward way for the design of these rules so that the overall system shows the desired properties. Typically, the emergent service is really hard, or even impossible to predict. Thus, finding a set of rules that causes the overall system to exhibit the desired properties presents a great challenge to the system designers. The main problem is that a small change of a parameter might lead to unexpected and even counterintuitive results [8, 9].

To design a self-organizing system with the desired emergent behavior, it is crucial to find local rules for the

behavior of the system's components (agents) that generate the intended behavior at system scale. In many cases, this is done by a sumptuous trial and error process which in case of systems with high complexity is not efficient or even unfeasible. Parameter-intensive systems also suffer from the unpredictability of the results due to unexpected dependencies between parameters.

In this paper, we discuss the application of evolutionary methods for designing an ANN-based control system for a team of self-organizing robots. In particular, we tackle the interface design between neural controller and robot and elaborate the particular influence of fitness function parameters on the results. As a case study for the approach, we describe the evolution of the neural control program for simulated soccer robots.

The paper is structured as follows: In the next section, we give an overview about self-organization and systems presenting the background of this paper will be given. Then the design steps as a general approach with the evolutionary algorithm will be described. Section 4 focuses on the practical evaluation of the approach, presenting the setup of the robot soccer simulation while Section 5 shows and explains the acquired results. Related work is discussed in Section 6. This paper is concluded in Section 7.

2. Self-Organizing Systems

The concept of self-organizing systems (SOS) was first introduced by Ashby [10] in 1947 referring to pattern-formation processes taking place within the system by the cooperative behavior of its individuals. These could best be described by the way they achieve their order and structure without having any external directing influences. There are several definitions for SOS [11], the following was formulated on the Lakeside Research Days'08 [7]:

"A self-organizing system (SOS) is a set of entities that obtains global system behavior via local interactions without centralized control."

An example could be a team of workers acting on their own following a mutual understanding. If there was any external influence like a common blueprint or a boss giving orders it would result in no self-organization. Many examples of SOS can also be observed in nature, for example, a school of fish where each individual has knowledge only about its neighbors and having no leader amongst them. The key part is the communication between the individuals; the way they satisfy their own goal such as getting close, but not too close to other fish in the school while trying to find food in the water.

Furthermore, it is important to note that the emergent property cannot be understood by simply examining the system's components in isolation, but requires the consideration of the interactions among the components. This interaction is not necessarily direct, it can be indirect as well when one component modifies the environment which then influences other components [12]. This presents a continuous mixture of positive and negative feedback in the behavior.

By describing self-organizing systems the following advantages can be observed. These systems are often very robust against external disturbances; it is clear that a failure of one component rarely results in a full collapse. Also adaptability and scalability can be noticed meaning a dynamical behavior and a flexibility in the number of components. It is also important to note that usually, once the local rules of the SOS are found, the implementation takes less effort compared to a centralized solution. These properties make SOS an interesting, though difficult to design, option for the decentralized control of complex systems. Although there are some ideas for designing such systems, there is no general methodology yet explaining how this should be done.

3. Evolutionary Approach to Design SOS

This section describes the proposed method giving SOS design engineers a tool. The process starts by defining the main goals: what are the expectations from our system. The next step is to build an evolvable representation of local behavior. Our approach uses an evolutionary algorithm to explore the solution space. Therefore, the evolvability of the representation is required, which means that operators like mutation and/or recombination must be defined on the representation. Instead of using a standard representation like a bit string as in genetic algorithms, the applied algorithm employs mutation and recombination functions which are representation specific. We have implemented a Java program called FREVO (http://www.frevotool.tk) which is an open-source framework for evolutionary design. It identifies and separates the key components, such as the problem definition, candidate representation, and the optimization method. As depicted in Figure 1, our framework supports different representations, where each must embed specific functions for mutation and combinations of candidates. The advantage of this approach is that the search space can be reduced since operations which likely produce unfeasible solutions are filtered. On the other hand, there is an increased effort for the implementation of a new representation by adding the specific mutation/recombination functions. Usually, control software is written in programming languages (JAVA, C, etc.) which are inappropriate for phenotypical mutation and recombination operators. One notable exception could be the LISP programming language which is used for the representation of an evolvable algorithm in [13]. Unlike standard programs, artificial neural networks (ANNs) or representations based on fuzzy rules are qualified for this task. Structure and setup of this representation is also a question; it can be trained by the evolutionary algorithm or defined a priori.

In case of ANNs reinforced learning is needed, since we have to deal with belated rewards we get after a simulation of many steps of the revised ANN. Thus, the standard back propagation algorithm cannot be applied to program the ANN's weights. At this point, we need our goals to be formulated as rewards for reinforcing the candidates. To make the learning process smoother, we propose a step by step approach decomposing the overall goal into smaller achievements weighted according to their significance. A typical example would be an object manipulation task for

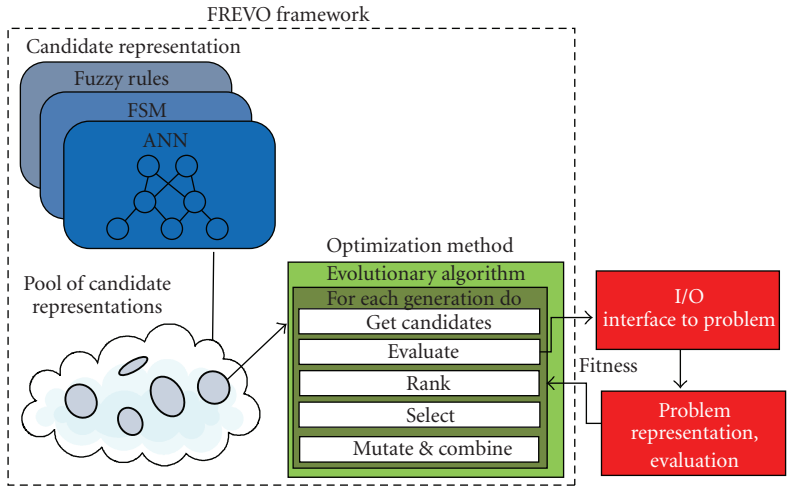


Figure 1: Components of the FREVO framework: The *agent representation* is optimized by the *evolutionary algorithm* to maximize the fitness for the given *problem*.

a robot where the three subtasks would be: finding, grabbing and then manipulating the object. In the next section, an example of this approach will be given.

With a simulation environment acting as a playground, the evolutionary algorithm can start evolving the possible candidates. Typically, a fitness value can be deduced from the simulation results. This fitness value is then used in the evolutionary algorithm to decide on the fittest individuals. An example for such a fitness value could be the throughput of a given setup of a wireless network. In many cases, an absolute fitness value cannot be assigned especially when a competitive emergent behavior is expected. In order to get a ranking of the individuals, it is necessary to play a tournament among the candidates of a population. In a native approach, the number of pairings equals $n(n-1)/2$; n being the number of individuals in a population. In case of long simulations, the time can be cut effectively by using the Swiss System, a pairing system used to organize (chess) tournaments which yield a ranking with a minimum number of pairings [14] instead of full tournament. Detailed description can be found in the next section.

4. Case Study

As a case study, we have defined the problem of teaching soccer to a team of autonomous robots in a 2D environment similar to the official RoboCup Simulation League. This problem provides a rich testbed domain for the study of control, coordination, and cooperation issues described in [15, 16]. We also presented some early results for this problem in [17]. In the following, we give a brief description of the actual problem of simulated robot soccer. Then, we present the representations of the particular elements according to the components depicted in Figure 1.

4.1. Problem Description. The problem is evaluated via a robot soccer simulator with simple physics, similar to the official simulator used for the RoboCup simulation league. In contrast to the official simulator, ours does not include the roles of a referee or goalkeeper and there is a simulated boundary around the field, which avoids situations where the ball goes out of bounds. The main change with respect to our approach is that our simulation does not run in real time (except for a built-in demonstration mode), but as a discrete event simulation that runs with maximum computation speed. This greatly reduces the actual time for performing the evolution. The chosen problem consists to the class of competitive evaluation, that is, we can only compare the relative fitness of two candidate solutions by simulating a match between them.

A simulation run consists of initially placing a configurable number of soccer players (typically 2×11) on a field and to simulate a game where each player can accelerate with a given strength towards a given direction and, if being close enough to the ball, can kick the ball with a given strength towards a given direction. One game lasts for 300 steps which yields 60 real-time seconds.

Table 1: Parameters of the evolutionary algorithms.

Population size	60
Number of generations	500
Percentage of elite selection	15
Percentage of mutations	40
Percentage of crossover	30
Percentage of randomly created offsprings	5
Percentage of randomly selecting an offspring from previous generation	10

4.2. Optimization Method. We used an evolutionary algorithm to evolve the controllers of the soccer players. The implementation of the optimization method is based on the one presented by Elmenreich and Klingler in [18]. The size of

the population was set to 60 and the length of the simulation was fixed at 500 generations. The parameters of the genetic operators can be seen in Table 1.

4.3. Candidate Representations. The candidates for the evolutionary algorithm were realized as ANN. Training has been applied to optimize the weights and biases of the neural network. We tested two different candidate representations in our case study, namely, a fully connected ANN and a three-layered ANN.

The fully connected network is a time-discrete, recurrent artificial neural network. Each neuron is connected to every other neuron and itself via several input connectors. Each connection is assigned a weight that is a floating value and each neuron is assigned a bias. The problem requires 16 inputs and 4 outputs. Additional "hidden" neurons are added in order to increase the expressiveness and the number of representable states.

The three-layered network only provides forward connections from the input nodes (the input layer) to the nodes in the hidden layer and forward connections from the hidden layer to the output layer.

In most cases, three-layered networks are employed for ANN applications, since they can be programmed via back propagation. However, our setup provides only belated rewards, that is feedback after a simulation involving many different actions of the ANN controller. The evolutionary algorithm works with both representations, so we can easily include the fully connected network.

The implementation of both types is almost identical, the only difference is that the three-layered network only features a subset of connections per neuron. At each step, each neuron i builds the sum over its bias b_i and its incoming connection weights $w_j i$ multiplied by the current outputs of the neurons $j = 1, 2, \ldots, n$ feeding the connections. Weights can be any real number, thus have either an excitatory or inhibitory effect. The output of the neuron for step $k+1$ is calculated by applying an activation function F

$$o_i(k+1) = F\left(\sum_{j=0}^{n} w_{ji} o_j(k) + b_i\right), \tag{1}$$

where F is a simple linear threshold function

$$F(x) = \begin{cases} -1.0, & \text{if } x \leq -1.0, \\ x, & \text{if } -1.0 < x < 1.0, \\ 1.0, & \text{if } x \geq 1.0. \end{cases} \tag{2}$$

As described below in detail, the output of the output neurons is further scaled to match the input range of the actuators.

4.4. I/O Interface between Simulation and Controllers. In the case study, the information passed to the simulation is predefined; it consist of the strength and direction for the players' move and, in case the player can kick the ball, the strength and direction of the kick. The information provided by the simulation is also determined; it consists of the position of the ball (if visible), a list of visible teammates, a list of visible opponent players, and information about the distance to the field's (upper, lower, left, right) border. The position of the goal is given indirectly by combining the distance to the borders with the knowledge that the goal resides in the middle of the side borders.

However, there are different ways to pass this information into the ANN. In general, the ANN will have a set of so-called input neurons, which activate their output according to a given input from external sources. Respectively, a number of output neurons is used to export information from the network. Finally, some unspecified or hidden neurons are added. All neurons are interconnected by directed weights, which are evolved in the framework (see Figure 2).

Since the number of neurons defines the search space, the number of neurons should not become too large. On the other hand, input neurons should be defined in a way that their result is easily interpretable by the ANN. For example in [18], we modeled a distance sensor that is periodically changing its orientation via several input neurons, each representing the input for a particular orientation.

In the robot soccer example, the inputs are arranged in groups of four neurons. Each group is responsible for communicating the detection of a particular object class (ball, teammate, opponent, border) and consists of a *north*, *south*, *east* and *west* neuron.

If the nearest object of that class is in a particular quadrant, lets say north-west, then the north neuron and the west neuron are activated inversely proportionally to the components of the vector to the object. So, if in our example the ball is towards the north-north-west, the north neuron gets a high activation and the west neuron a moderate one (see Figure 3).

For the output neurons, we tested two setups: in the first setup, the outputs are scaled to $[-100\%, 100\%]$ and $[-180, +180]$, respectively and interpreted as polar coordinates for the move and kick vectors. The second setup interprets the neurons as being the x and y components of a vector in cartesian coordinates. In general, both approaches are expected to work, since they transport exactly the same information and the ANN will be evolved to adjust to the given representation.

4.5. Fitness Evaluation. Typically, when the task is more complex, the definition of the fitness function is not trivial. In the case of a soccer game, the primary aim is to train teams scoring the most goals during the given time interval. However, this problem is far too complicated for teams, initially composed of random ANN controllers, to expect improvement over generations just by rewarding them by the final number of goals they score in each game. The idea here is to decompose the overall goal into smaller achievements (so-called guidelines) and let the teams fulfill them one after the other. This method tries to ensure a smooth learning process assuming some preliminary knowledge or ideas about the solution. The guidelines are assigned a weight to setup a hierarchical order. It means a task with smaller weight is less important, but will most likely be accomplished before

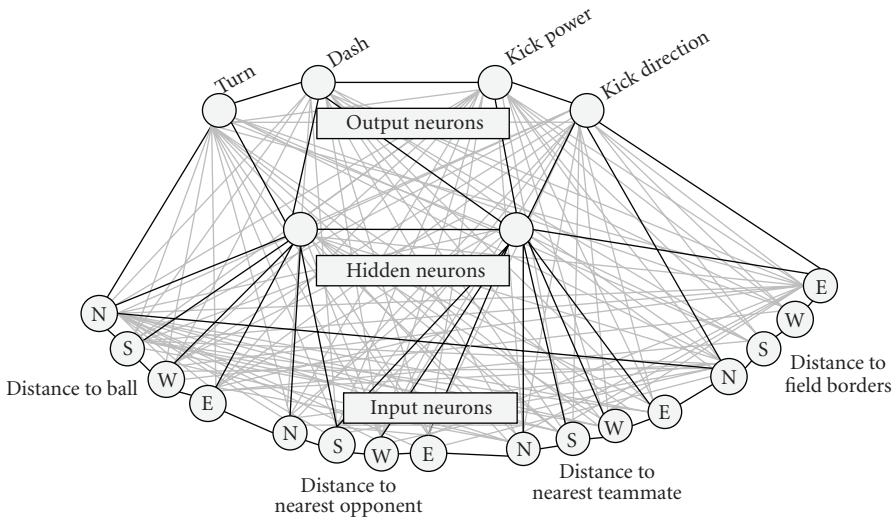


Figure 2: A possible wiring of the neural network showing the groups of inputs, outputs and hidden neurons. Connections with stronger weight are indicated with bold lines while ones with lower weight are colored with grey.

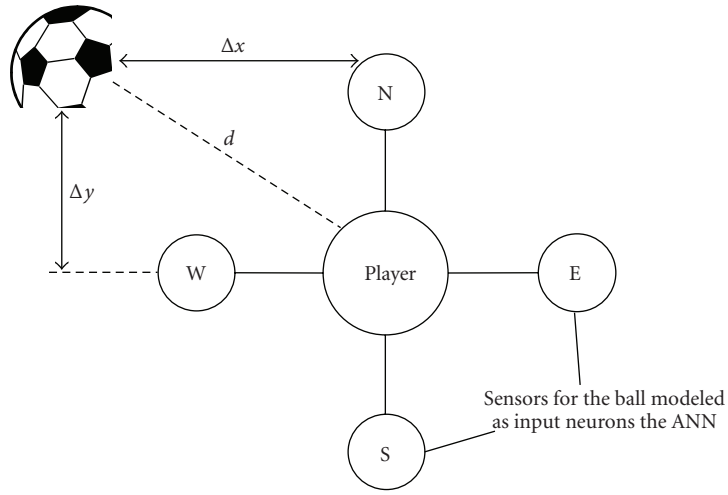


Figure 3: A group of input neurons detecting the ball.

another tasks with higher value. This is because the second one is too complex to be achieved without learning the first one. Figure 4 shows the applied tasks in their respective order in our simulation.

At the beginning of the training, we wanted the teams to learn that a good distribution on the field might lead to good overall play. Therefore, we introduced the first guideline (field distribution). It was implemented by defining 64 evenly distributed checkpoints on the field and counting the number of controlled points every 5 seconds for both teams. A point is controlled by a team if it has the nearest player to this point. The accumulated points are added to the final fitness value. The second guideline was an advice for the

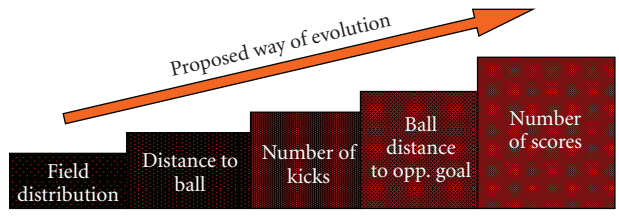


Figure 4: Weighted fitness.

teams to move their players closer to the ball. The distance of the nearest player for both teams to the ball is measured

Table 2: Parameters of the fitness function.

i	P_i	W_i
p	field distribution	10^0
bd	distance to the ball	10^3
k	number of kicks	$2 \cdot 10^4$
fk	number of false kicks (ball is kicked out of bounds)	10^4
bg	ball distance to the opponent's goal	10^5
s	number of scores	$4 \cdot 10^6$

and compared every 4 seconds. The team having a player closer to the ball earns one point. At the end of the game, this point is weighted and also added to the final fitness. The number of kicks is also counted with a weight however only the first 10 kicks are taken into account to create an upper bound and to prevent dead team strategies where they only pass the ball back and forth. Concerning the kicking direction, the ball distance to the opponents goal is also measured and calculated every 2 game seconds in the same manner as guideline two. The highest weight is assigned to the number of scores, being more significant than the other fitness components. Therefore, we define the fitness function as the following equation:

$$F = W_p P_p + W_{bd} P_{bd} + \left(W_k P_k - W_{fk} P_{fk}\right) + W_{bg} P_{bg} + W_s P_s, \quad (3)$$

where W and P stand for the weights and the points, respectively. Table 2 explains the corresponding indexes and values.

4.6. Tournament Ranking with Swiss System. Evolving competitive team behavior is a good example where one cannot assign a simple absolute fitness value. To rank the teams one solution is to play a tournament among the candidates in each generation (assuming one population with n candidates). A full tournament would mean $n(n-1)/2$ number of pairings when n is the number of entities in the population. In case a simulation run takes too much time or a high number of generations is needed, this approach can be very ineffective. For example, a population of 50 individuals would require 1225 runs for each generation. The proposed solution tries to minimize the number of necessary pairing using Swiss System style tournament [19]. It reduces the required number to $\lceil \log_2 n \rceil (n/2)$ which is in the mentioned case only 150 games (see Figure 5). Inspired by the official FIDE (http://www.fide.com) rules for chess tournaments, we established the following system.

In each game, the winner gets two points, loser gets zero, in case of a draw both get one point. After the first round, players are placed in groups according to their score (winners in group "2", those who drew in group "1", and losers in group "0"). The aim is to ensure that players with the same score play against each other. Since the number of perfect scores is cut in half each round, it does not take long until there is only one player left with a perfect score. The actual number of rounds needed is $\log_2 n$ to be able to handle n teams. In chess tournaments, there are usually many draws, so more players can be handled (a 5 round event can usually determine a clear winner for a section of at least 40 players, possibly more), although in our simulation a draw is very unlikely. To avoid early games between elite selected entities, the first round is not randomized but cut into two halves where the first half, consisting of teams which have performed well so far, is playing against the second half.

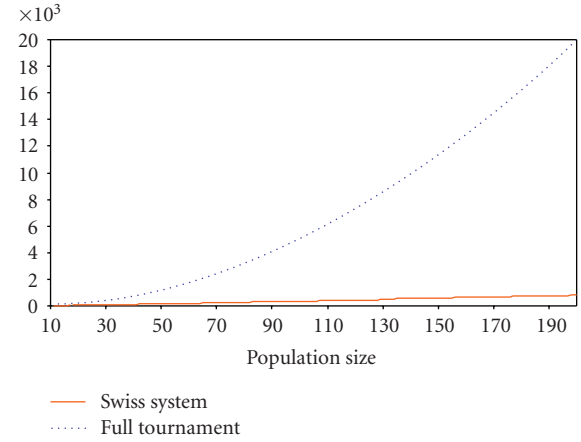


Figure 5: Total number of games in full tournament and Swiss System.

The drawback of the Swiss system is that it is only designed to determine a clear winner in just a few rounds. Regarding other players, we have little information about their correct ranking. For example, there could be many players with 3-2 scores and it is hard to say which player is better than the other, or whether a player with 3.5 points is better than a player with 3 points. To help determine the order of finish, a tiebreak method has been implemented. In order to decide on the ranking for players having the same score, we used a method developed by Bruno Buchholz [20]. There the score of the players' opponents is summed up thus favoring those who have confronted better opponents. In case it is still undecided, the sum is extended by the points of those opponents who have lost against the player. This uncertainty in the ranking could cause problems in the evolutionary algorithm when selecting entities for survival to the next generation. In our case, elite selection was 15% while the Swiss System ensures only the first and last position to be ranked correctly, thus the position of all other players carries also some obscurity. After observing this effect in our simulation, we came to the conclusion that having a somewhat imprecise selection among the top players slows down the process just a little or not at all. To select entities for survival, we used a roulette wheel selection where the probability being selected is directly proportional to the fitness, in our case the ranking of the Swiss System. Since this approach already carries some randomization some more uncertainty did not make a crucial impact.

5. Results

We ran several simulations evolving soccer teams. In particular, we varied

(i) the type of representation (fully connected or layered ANN),

(ii) the number of hidden nodes (2, 4, or 6),

(iii) the type of the interface between simulation and controllers.

We evolved each setting up to 500 generations. Unfortunately, there is no absolute fitness value for depicting the quality of an evolved result. Only relative comparisons of teams by matching them in a simulation are possible. For our evaluation, we picked the best result of every 20th generation. These "champions" have than been matched in a round-robin tournament against each other in order to determine if there is a constant evolution towards better gameplay and if one setting is performing better than the other.

We found out that the design of the interface between simulation and controllers is of major importance to the success of the evolutionary algorithm. The results showed that the selection of the interface between simulation and controllers has a significant influence on the speed of convergence and quality of the evolved solution. When the output neurons were interpreted as polar coordinates, the ANN controller needed to learn the coherent semantics of polar coordinates, and, probably, learn to emulate trigonometric functions. Figure 6 visualizes this observation for both, layered and fully connected ANN. The figure depicts the results of a tournament of the above mentioned champions for various settings. Most curves are increasing over generations which depicts that the gameplay of the teams has been improved by the evolutionary algorithm.

As can be seen in the graphs, the systems using cartesian coordinates, even after several hundreds generations, are ranked lower than almost every other systems using cartesian coordinates. So, in this case, yielding output in polar coordinates posed a higher "cognitive complexity" for the system than yielding output in cartesian coordinates. Figure 7 shows a boxplot covering 10 different iterations of the evolutionary algorithms and also confirms that the cartesian coordinate I/O model is significantly superior to the polar coordinate model. This, however, may be specific for the chosen problem and may be different for other problems.

There was no significant effect of the number of hidden neurons on the overall performance or speed of convergence. This could be explained by the fact that a less complex ANN is already sufficient to learn the local interactions producing a competitive behavior.

Figure 8 compares the different versions using polar coordinates with each other and depicts that the fully connected ANN have evolved faster and to a better gameplay than the layered networks. The box plot diagram in Figure 9 gives a statistic over 10 different runs of the evolutionary algorithm. While it confirms that all fully connected ANNs are typically better than the layered ANNs with 2 hidden neurons, it also shows that a layered ANN with enough neurons (that is 6 in that case) is in many iterations able to compensate for the lower number of connections.

Thus, a fully connected network with 6 hidden neurons and an I/O interface based on cartesian coordinates evolved 400 generations or more performed best according to the ability to win over others. Unfortunately, the quality and elegance of the result cannot be measured in this terms. By watching several games, we observed the following behavior (a video of the evolution of the gameplay can be found at http://mobile.uni-klu.ac.at/demesos):

(i) the player nearest to the ball runs to the ball,

(ii) other players (of the same team) in the vicinity of the ball also follow the ball, but they do not usually converge to the same spot; instead they keep spread out,

(iii) the player at the ball kicks it to a direction bringing it nearer to the opponent's goal,

(iv) players far from the ball spread out and build a defense mesh in front of their own goal

(v) players sometimes tend to stick to opponent players (man marking).

Considering the relatively small size and complexity of the neural network controllers, the versatility of the emerging strategy is impressive. When both teams are well evolved, the ball is passed over several stations until the ball possession changes. Goals are scored roughly every few hundred simulation steps.

6. Related Work

In literature, only a few proposals for designing self-organizing systems can be found. First, a method proposed by Gershenson [21] introducing a notion of "friction" between two components as a utility to design the overall system using trial and error. Methods building on trials, even if they are improved by certain notions, often suffer from counterintuitive interrelationships between local rules and emergent behavior.

Observing and learning from nature is also proven to be useful in several scenarios [22]. If an appropriate model exists and is available for study, top-down approaches can be very effective by applying the same phenomena. Conversely, bottom-up approaches adopt principles from nature and use on a fundamental basis [23].

There is also an imitation-based approach proposed by Auer et al. [24] where the behavior of a hypothetical omniscient "perfect" agent is analyzed and mimicked in order to derive the local rules. A good example would be a perfect poker player who can see the hand of all other players and his decisions can be analyzed to create a relatively good normal player. The problem here and in all methods based on imitation is the limitation to cases where an appropriate example model is available.

Evolutionary algorithms have been applied in several ways to evolve ANNs. Yao and Liu [25] describe a general method for simultaneously evolving the weights of an ANN.

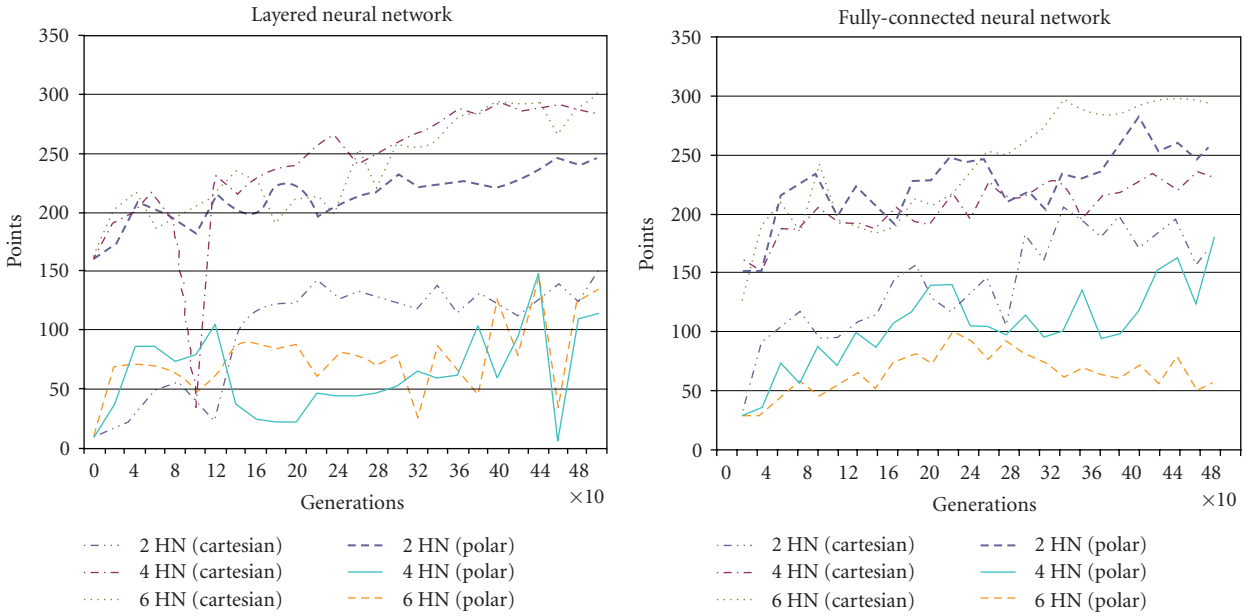


Figure 6: Tournament results of ANN with different I/O interfaces.

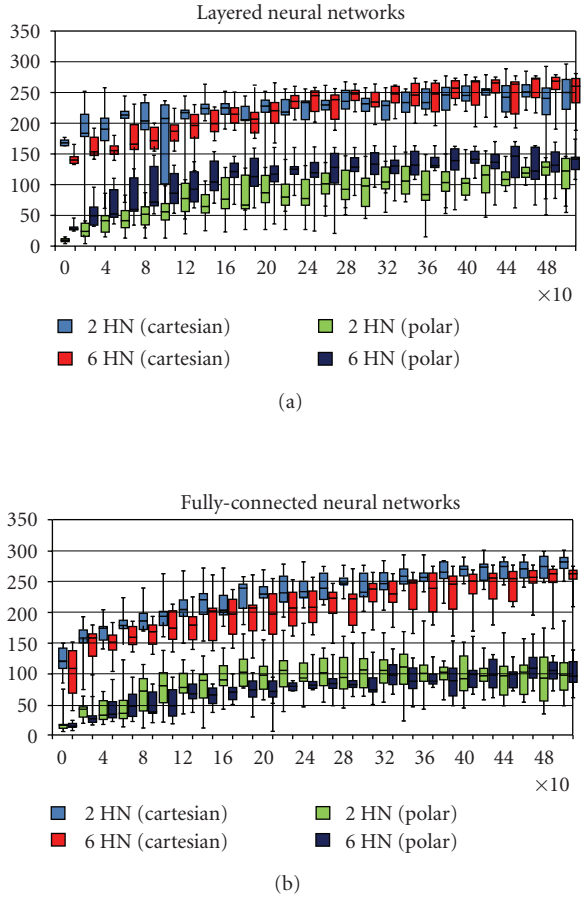


Figure 7: Box-and-whisker diagram of the repeated evaluation of different I/O models and different number of hidden neurons for layered and fully connected ANNs.

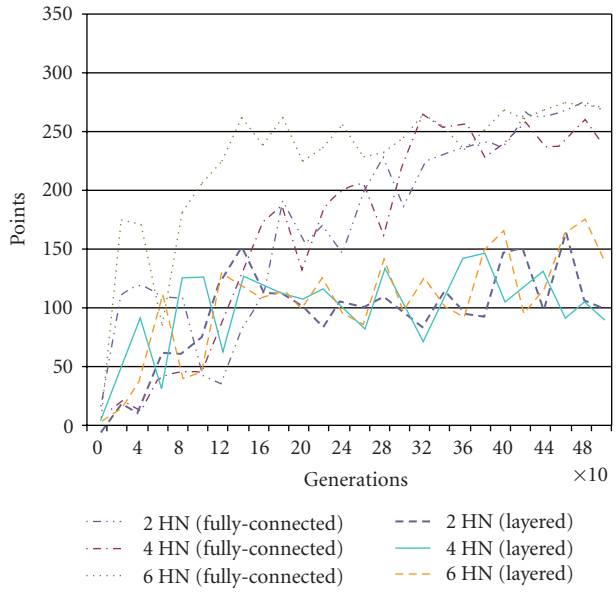


Figure 8: Tournament results of fully connected versus layered ANN with cartesian interface.

Meeden [26] proposes a solution solely based on mutation and selection without crossovers. Floreano and Mondada [27] describe the evolving of a navigation system based on discrete-time recurrent networks. They successfully evolve a network for controlling a single Khepera robot. The work of Baldassarre et al. [28] shows evolving physically connected robots using ANN controllers. Sipper et al. [29] shows the versatility of the approach by applying it to different game playing problems.

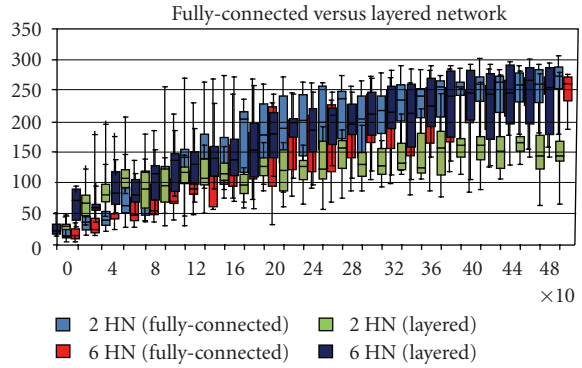


Figure 9: Box-and-whisker diagram of the repeated evaluation of fully connected versus layered ANN with cartesian interface.

Nelson et al. [30] describes the evolution of multiple robot controllers towards a team that plays "Capture the flag" against an opponent team of robots. This work applies the relative fitness concept as it is proposed in our approach. In contrast, Coelho et al. [31] evolve soccer teams evaluating against a control team. In particular, they evolve separate ANN for the different player behaviors (defense, middle and attack). In our approach, we evolve teams, where the same replicated ANN controls all the players. Based on inputs about situation and the behavior of the other teammates, a self-organizing process leads to a differentiation into the particular behavior types during run time.

7. Conclusion and Future Work

We have described a method for evolving neural network controllers for a team of cooperative robots. Given an overall goal function, we evolve the particular weights and biases of the neural network controllers using an evolutionary algorithm. Thus, the neural network learns to interpret the sensory inputs, to control the robots actuators, and to behave according to a strategy that is beneficial for the given task. The approach is very flexible and can be applied to a wide variety of problems but it depends on a sufficiently accurate simulation and a fitness function that provides the necessary gradients for the evolutionary algorithm.

In a case study, we have evolved a control behavior for simulated soccer robots to cooperatively win soccer games. After a few hundred generations, the players of a team adopt a useful behavior. In contrast to related work, the players were not evolved to a priori defined roles, like defender, midfielder or striker, but all have an instance of the same neural network controller. Still, during a game, different behavior of the players emerge based on their situations. Thus, similar to biological systems, the entities specify to different roles in a self-organizing way. Since the entities are identical, the system has a high robustness against failure of some of the entities.

We have examined the influence of various factors to the results. The most important factor was the design of the interface between neural network and sensors/actuators. Although an ANN could theoretically adopt to different representations of sensor/actuator interfaces, it was necessary to find an interface with low cognitive complexity for the ANN, which was in our case a simple cartesian representation of the sensors and intended robot movements. Furthermore, we analyzed the influence of using different sizes and types of ANN. While the number of neurons had the smallest effect on the performance, the type of representation favored the fully connected network type.

In the future, we plan to assess the robustness and fault tolerance of the generated solutions. Furthermore, our system is open to changes in the representation (e.g., using different controller types), the optimization method, and the problem definition (e.g., apply the approach to different problem domains).

Acknowledgments

This paper was supported by the European Regional Development Fund and the Carinthian Economic Promotion Fund (contract KWF 20214/18128/26673) within the Lakeside Labs project DEMESOS. The authers would like to thank Herwig Guggi and Kornelia Lienbacher for the constructive comments on an earlier version of this paper.

References

[1] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng, "Cooperative mobile robotics: antecedents and directions," *Autonomous Robots*, vol. 4, no. 1, pp. 1–23, 1997.

[2] S. Johnson, *Emergence: The Connected Lives of Ants, Brains, Cities, and Software*, Scribner, 2002.

[3] D. Floreano and S. Nolfi, *The Role of Self-Organization for the Synthesis and the Understanding of Behavioral Systems*, chapter 1, MIT Press, Cambridge, UK, 2000.

[4] G. Novak, "Roboter soccer: an example for autonomous mobile cooperating robots," in *Proceedings of the 1st Workshop on Intelligent Solutions for Embedded Systems (WISES '03)*, pp. 107–118, Vienna, Austria, 2003.

[5] S. Bergbreiter and K. S.J. Pister, "Design of an autonomous jumping microrobot," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 447–453, 2007.

[6] J. Roberts, T. Stirling, J. Zufferey, and D. Floreano, "Quadrotor using minimal sensing for autonomous indoor flight," in *Proceedings of European Micro Air Vehicle Conference and Flight Competition (EMAV '07)*, 2007.

[7] W. Elmenreich and H. de Meer, "Self-organizing networked systems for technical applications: a discussion on open issues," in *Proceedings of the 3rd International Workshop on Self-Organizing Systems (IWSOS '08)*, J. P. G. Sterbenz and K. A. Hummel, Eds., vol. 5343 of *Lecture Notes in Computer Science*, pp. 1–9, Springer, Vienna, Austria, December 2008.

[8] M. Resnick, *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds (Complex Adaptive Systems)*, The MIT Press, 1997.

[9] I. Harvey, E. Di Paolo, R. Wood, M. Quinn, and E. Tuci, "Evolutionary robotics: a new scientific tool for studying cognition," *Artificial Life*, vol. 11, no. 1-2, pp. 79–98, 2005.

[10] W. R. Ashby, "Principles of the self-organizing dynamic system," *Journal of General Psychology*, vol. 37, no. 3, pp. 125–128, 1947.

[11] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*, Princeton University Press, 2001.

[12] E. Bonabeau, "Editor's introduction: stigmergy," *Special Issue of Artificial Life on Stigmergy*, vol. 5, no. 2, pp. 95–96, 1999.

[13] J. R. Koza, Ed., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, 1st edition, 1992.

[14] T. Just and D. B. Burg, Eds., *U.S. Chess Federation's Official Rules of Chess*, Random House Puzzles & Games, New York, NY, USA, 5th edition, 2003.

[15] J. Kummeneje, *RoboCup as a measure to research, education, and dissemination*, Ph.D. thesis, Stockholm University and the Royal Institute of Technology, Kista, Sweden, 2003.

[16] S. Buck and M. A. Riedmiller, "Learning situation dependent success rates of actions in a robocup scenario," in *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, p. 809, Melbourne, Australia, August-September 2000.

[17] I. Fehérvári and W. Elmenreich, "Evolutionary methods in self-organizing system design," in *Proceedings of the International Conference on Genetic and Evolutionary Methods*, 2009.

[18] W. Elmenreich and G. Klingler, "Genetic evolution of a neural network for the autonomous control of a four-wheeled robot," in *Proceedings of the 6th Mexican International Conference on Artificial Intelligence, Special Session (MICAI '07)*, pp. 396–406, Aguascalientes, Mexico, November 2007.

[19] FIDE swiss rules. Approved by the General Assembly of 1987. Amended by the 1988 & 1989 General Assemblies.

[20] Wikipedia, "Buchholz system—wikipedia, the free encyclopedia," March 2009.

[21] C. Gershenson, *Design and control of self-organizing systems*, Ph.D. thesis, Vrije Universiteit Brussel, Brussel, Belgium, 2007.

[22] A. Tyrrell, G. Auer, and C. Bettstetter, "Biologically inspired synchronization for wireless networks," in *Advances in Biologically Inspired Information Systems: Models, Methods, and Tools*, vol. 69, pp. 47–62, 2007.

[23] M. Dorigo and T. Stützle, *Ant Colony Optimization*, Bradford Books, 2004.

[24] C. Auer, P. Wüchner, and H. de Meer, "A method to derive local interaction strategies for improving cooperation in self-organizing systems," in *Proceedings of the 3rd International Workshop on Self-Organizing Systems (IWSOS '08)*, vol. 5343 of *Lecture Notes in Computer Science*, pp. 170–181, Springer, Vienna, Austria, December 2008.

[25] X. Yao and Y. Liu, "Evolving artificial neural networks through evolutionary programming," in *Proceedings of the 5th Annual Conference on Evolutionary Programming*, pp. 257–266, MIT Press, 1996.

[26] L. A. Meeden, "An incremental approach to developing intelligent neural network controllers for robots," *IEEE Transactions on Systems, Man, and Cybernetics Part B*, vol. 26, no. 3, pp. 474–485, 1996.

[27] D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics Part B*, vol. 26, no. 3, pp. 396–407, 1996.

[28] G. Baldassarre, D. Parisi, and S. Nolfi, "Distributed coordination of simulated robots based on self-organization," *Artificial Life*, vol. 12, no. 3, pp. 289–311, 2006.

[29] M. Sipper, Y. Azaria, A. Hauptman, and Y. Shichel, "Designing an evolutionary strategizing machine for game playing and beyond," *IEEE Transactions on Systems, Man and Cybernetics Part C*, vol. 37, no. 4, pp. 583–593, 2007.

[30] A. L. Nelson, E. Grant, and T. C. Henderson, "Evolution of neural controllers for competitive game playing with teams of mobile robots," *Robotics and Autonomous Systems*, vol. 46, no. 3, pp. 135–150, 2004.

[31] A. L. V. Coelho, D. Weingaertner, and F. Gomide, "Evolving coordination strategies in simulated robot soccer," in *Proceedings of the 5th International Conference on Autonomous Agents*, pp. 147–148, Montreal, Canada, May-June 2001.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 908132, 11 pages
doi:10.1155/2010/908132

Research Article

Computer Simulation Tests of Feedback Error Learning Controller with IDM and ISM for Functional Electrical Stimulation in Wrist Joint Control

Takashi Watanabe[1] and Yoshihiro Sugi[2]

[1] *Department of Biomedical Engineering, Graduate School of Biomedical Engineering, Tohoku University, Sendai 980-8579, Japan*
[2] *Department of Electrical and Communication Engineering, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan*

Correspondence should be addressed to Takashi Watanabe, nabet@bme.tohoku.ac.jp

Received 29 October 2009; Accepted 18 April 2010

Academic Editor: Noriyasu Homma

Feedforward controller would be useful for hybrid Functional Electrical Stimulation (FES) system using powered orthotic devices. In this paper, Feedback Error Learning (FEL) controller for FES (FEL-FES controller) was examined using an inverse statics model (ISM) with an inverse dynamics model (IDM) to realize a feedforward FES controller. For FES application, the ISM was tested in learning off line using training data obtained by PID control of very slow movements. Computer simulation tests in controlling wrist joint movements showed that the ISM performed properly in positioning task and that IDM learning was improved by using the ISM showing increase of output power ratio of the feedforward controller. The simple ISM learning method and the FEL-FES controller using the ISM would be useful in controlling the musculoskeletal system that has nonlinear characteristics to electrical stimulation and therefore is expected to be useful in applying to hybrid FES system using powered orthotic device.

1. Introduction

Functional electrical stimulation (FES), which applies electric current or voltage pulses to peripheral nerves and muscles, is a method of restoring or assisting motor functions lost by the spinal cord injury or the cerebrovascular disease. FES has been found to be effective clinically, especially in controlling paralyzed upper limbs [1–3]. For restoring lower limb functions, the hybrid FES system, which uses an orthosis with FES, has been accepted as one of practical methods [4, 5].

In the recent years, powered orthotic devices or robotic exoskeletons have been focused on an assist or rehabilitation of lower limb functions [6, 7]. Therefore, the hybrid FES system is also expected to be realized with powered orthotic devices. In such system, cooperative control between FES and powered orthosis will be necessary. Feedforward control scheme would be useful for controlling fast movements of lower limbs in tracking to movements developed by the powered orthosis because control performance of a feedback controller is limited by large time delay and time constant in responses of electrically stimulated muscles. However, complex, time-consuming adjustment of many parameters of the feedforward controller such as creating stimulation data for a lot of muscles and time-varying properties of the musculoskeletal system make it difficult to use practically the feedforward FES controller in clinical application.

The Feedback Error Learning (FEL) proposed by Kawato et al. [8, 9] can realize a feedforward controller by learning inverse dynamics of controlled object. The FEL will be useful in FES control because it can learn nonlinear characteristics of the musculoskeletal system to electrical stimulation and can remove the problem of manual adjustment of controller parameters by medical staffs in applying to various subjects that have different characteristics of the musculoskeletal system.

In order to apply the FEL, a feedback controller is required. The multichannel feedback FES controller has

to solve the ill-posed problem in regulating stimulation intensities because the number of stimulated muscles is larger than that of controlled joint angles. The feedback FES controller based on the Proportional-Integral-Derivative (PID) control algorithm that we developed could provide a way of solving the ill-posed problem [10, 11]. In our previous work, the FEL controller for FES (FEL-FES controller) using the PID controller was found to be feasible in controlling 1-Degree-Of-Freedom (1-DOF) of wrist joint movement (dorsi- and palmar flexions) stimulating 2 muscles [12].

The FEL-FES controller makes it possible to use both the feedforward and feedback controllers, which is an advantage for the cooperative control between FES and powered orthosis in the hybrid FES system. Therefore, we performed preliminary test to expand the previous FEL-FES controller into controlling 2-DOF movements stimulating 4 muscles through computer simulation. However, the previous FEL-FES controller had a problem in learning the inverse dynamics. That is, learning the inverse dynamics model (IDM) in the previous FEL-FES controller sometimes failed.

Since a major problem in applying the FEL to FES is inappropriate learning of the IDM in FES control, a modification of the FEL-FES controller was discussed through computer simulation before testing with human subjects and applying the controller to hybrid FES system in this paper. In the previous FEL-FES controller, the IDM was only used for the feedforward controller since learning an inverse statics model (ISM) was not easy in clinical applications of FES because of difficulty in acquiring training data, while the FEL controller by Kawato was composed of the ISM and the IDM.

In this paper, in order to include the ISM into the feedforward controller, a simple measurement method of training data for the ISM was introduced considering FES applications. The ISM learning and the modified FEL-FES controller including the ISM were examined in wrist joint movement control by computer simulations in order to be compared to our previous work.

2. Feedback Error Learning Controller for FES

2.1. Outline. A block diagram of the feedback error learning controller for FES examined in this study is shown in Figure 1. The sum of output stimulation intensities from feedforward controllers (ISM and IDM) and a feedback controller is applied to each muscle after adding offset (threshold value of electrical stimulation intensity) and clipping out with the limiter to prevent excessive stimulation.

The PID controller outputs positive and negative values of stimulation intensity for each muscle to cancel out the difference between the desired joint angle (θ_d) and the actual angle (θ) during movement control. The outputs were also used in IDM learning on line.

Two three-layered artificial neural networks (ANNS) were used for ISM and IDM. The IDM and the ISM output positive values of stimulation intensity to each muscle calculated from the desired joint angle (θ_d), while the IDM uses the first and second derivatives of the desired angle. The ISM is trained off line before IDM learning, and then the IDM is done on line using outputs of the feedback controller.

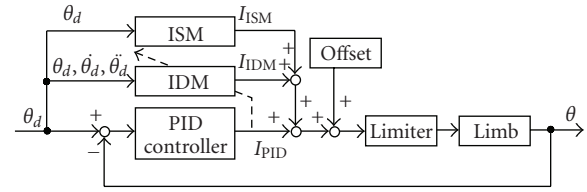


Figure 1: Feedback error learning controller tested in this study. The inverse statics model (ISM) and inverse dynamics model (IDM) were used as the feedforward controller.

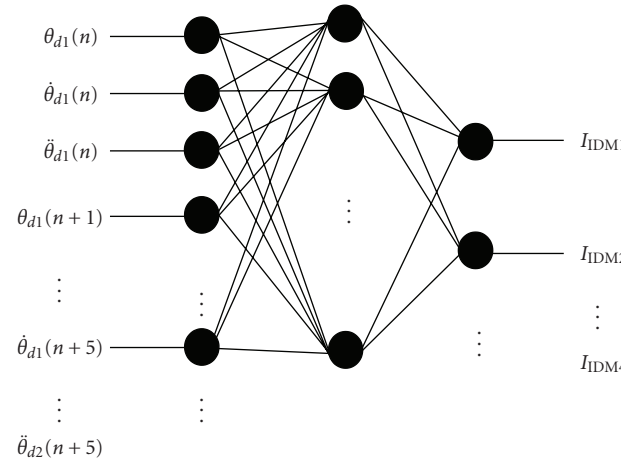


Figure 2: Structure of ANN for IDM used in the FEL-FES controller.

2.2. Feedforward Controller. The structure of ANN for the IDM is shown in Figure 2. The input data of the desired joint angle and its first and second derivatives at continuous 6 times, from t to $t + 5$, (50 ms interval) in the directions of dorsi/palmar flexion ($\theta_{d1}, \dot{\theta}_{d1}$, and $\ddot{\theta}_{d1}$) and radial/ulnar flexion ($\theta_{d2}, \dot{\theta}_{d2}$, and $\ddot{\theta}_{d2}$) were given simultaneously. Outputs were stimulation intensities to 4 muscles. Therefore, the numbers of neurons in the IDM were 36 for the input layer and 4 for the output layer. That for the hidden layer was 18, which was determined based on our previous results [12].

The output of each neuron in the hidden and the output layers was defined as

$$y = f\left(\sum_i w_i x_i + c\right) \tag{1}$$

where x_i represents outputs of the neurons in the previous layer, w_i is the connection weight from neurons in the previous layer, c is the bias term, and i is the index of the neuron in the previous layer. The output function $f(x)$ of the neuron is the sigmoid function

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{2}$$

The IDM was trained on line by the error backpropagation algorithm [13, 14] using outputs of the PID controller. ANN connection weights are changed to reduce total error, E, as follows:

$$E = \frac{1}{2}(I_{\text{desired}} - I_{\text{IDM}})^T \cdot (I_{\text{desired}} - I_{\text{IDM}}) \quad (3)$$

$$\frac{dw}{dt} = \varepsilon\left(\frac{\partial I_{\text{IDM}}}{\partial w}\right)^T (I_{\text{desired}} - I_{\text{IDM}}) \quad (4)$$

where I_{desired} and I_{IDM} are desired stimulation intensity and stimulation intensity of the IDM, respectively. ε is the learning speed coefficient that has effect on convergence speed of learning. $I_{\text{desired}} - I_{\text{IDM}}$ is approximated by stimulation intensity of the PID controller, I_{PID}.

The ISM was trained off line before the IDM learning by using the error backpropagation algorithm. The three-layered ANN that had 2 neurons for the input layer, 18 and 4 for the hidden and the output layers, was used for the ISM. The ISM and the PID controller output stimulation during control for IDM learning, although outputs of the ISM were not used for IDM learning.

2.3. Feedback Controller. The following PID control algorithm was used in the FEL-FES controller as the feedback controller:

$$\mathbf{I}_{\text{PID}}(n) = \mathbf{K}_{\text{P}}\mathbf{e}(n) + \mathbf{K}_{\text{I}}\sum_{i=0}^{n}\mathbf{e}(i) + \mathbf{K}_{\text{D}}\{\mathbf{e}(n) - \mathbf{e}(n-1)\} \quad (5)$$

where the error vector $\mathbf{e}(n)$ is defined as difference between desired and measured joint angle vectors at time n. The PID parameter matrices $\mathbf{K}_{\text{P}}$, $\mathbf{K}_{\text{I}}$, and $\mathbf{K}_{\text{D}}$ were determined by modifying the Chien, Hrones, and Reswick (CHR) method, and their elements were expressed as follows [10]:

$$K_{\text{P}ij} = \frac{0.6T_i}{L_i}m_{ij}^-, \qquad K_{\text{I}ij} = \frac{0.6\Delta t}{L_i}m_{ij}^-, \qquad K_{\text{D}ij} = \frac{0.3T_i}{\Delta t}m_{ij}^- \quad (6)$$

where L_i and T_i are the latency and the time constant of the step response of muscle i, when the response is approximated to the first order delay with latency. Δt is the sampling period. In case that a muscle has two or more functions (j shows index of the function), the delay time and the time constant obtained for every components in a movement were averaged, respectively. The coefficient m_{ij}^- corresponds to a reciprocal of the steady state gain of the system, which is calculated as an element of a generalized inverse matrix of a transformation matrix $\mathbf{M}$. The matrix $\mathbf{M}$ transforms change of stimulation intensity vector into change of joint angle vector. Calculation method of the coefficient m_{ij}^- is shown in Appendix A.

3. Computer Simulation Tests

The FEL-FES controller including the ISM was tested in controlling 2-DOF movements of the wrist joint. The muscles to be stimulated were the extensor carpi radialis longus/brevis (ECRL/ECRB), the extensor carpi ulnaris (ECU), the flexor crpi radialis (FCR) and the flexor crpi ulnaris (FCU). The ECRL and the ECRB were assumed to be one muscle group (ECR) because of difficulty in selective stimulation to them in experiments using surface electrodes that we performed [10].

For computer simulation tests of learning the ISM and the IDM and of control performance, a musculoskeletal model of the upper limb was developed. In brief, muscle force F_{CE} produced by electrical stimulation was described by the Hill type muscle model with nonlinear length-force relationship $k(l)$ and nonlinear velocity-force relationship $h(v)$, which included muscle activation level $a_m(s)$ determined by nonlinear recruitment characteristics with dynamics to applied electrical stimulation (refer to Appendix B for details). That is,

$$F_{\text{CE}} = a_m(s)k(l)h(v)F_{\max} \quad (7)$$

where s, l, and v were normalized stimulation intensity, muscle length and contraction velocity, respectively. $F_{\max}$ showed a constant of maximum muscle force. Active torque τ_{CE} produced by electrical stimulation was calculated by muscle force F_{CE} and moment arm $r_f(\theta)$. That is,

$$\tau_{\text{CE}} = F_{\text{CE}}\, r_f(\theta). \quad (8)$$

Moment arm $r_f(\theta)$ was represented by an approximated polynomial equation as a nonlinear function of joint angle θ for each movement developed by each muscle [15]. Six different subject models were prepared, in which the difference between 6 subjects was represented by adjusting mainly parameters of recruitment characteristics based on step responses and input-output (stimulus intensity-joint angle) relationships of the muscles measured on 6 neurologically intact subjects.

In this study, ISM learning was carried out off line using training data that consisted of stimulation intensities to 4 muscles and 2 joint angles. A set of training data was obtained by the tracking control of very slow movements using the PID controller. Figure 3 shows target trajectories of the tracking controls to obtain the training data set. The cycle period was 30 s for all trajectories. In Figure 3(a), the training data set was obtained from 2 target trajectories which were ellipses on the joint angle plane with the major radius of 20 deg in dorsi/palmar flexion and the minor radius of 15 deg in radial/ulnar flexion and those of 10 deg and 7.5 deg. Four target trajectories as shown in Figure 3(b) were also used for measurement of another training data set for ISM learning, in which 2 trajectories with the radius of 15 deg and 11.25 deg and 5 deg and 3.5 deg were added to those in Figure 3(a). The ISM was trained off line applying training data in random order. Initial values of the ANN connection weights were random values.

The IDM was trained on line for different 5 target trajectories shown in Figure 4, which were also ellipses on the joint angle plane with the radius of 20 deg in dorsi/palmar flexion and that of 15 deg in radial/ulnar flexion. The centers of those trajectories were 0 deg, 5 deg moved to the radial, ulnar, dorsi, and palmer directions. Three cycle periods, 2, 3, and

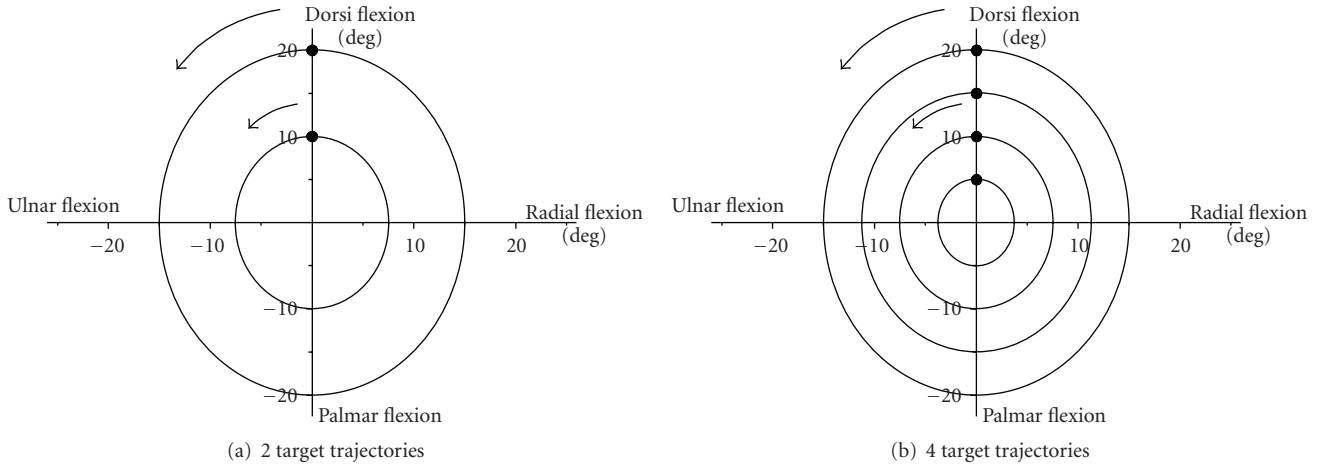


Figure 3: Target joint angle trajectories to obtain training data for ISM learning. Cycle period was 30 s for all trajectories.

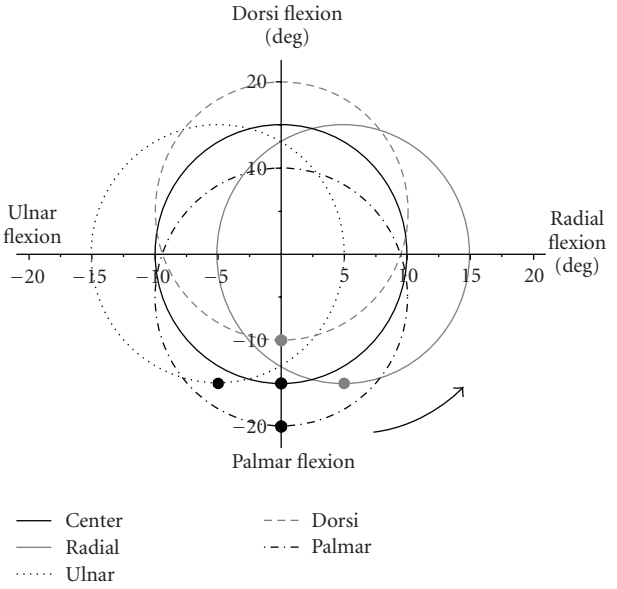


Figure 4: Target joint angle trajectories for IDM learning. Each movement had different center.

6 s, were used for all trajectories. Six cycles were included in one control trial for IDM learning. Three sets of initial values of ANN connection weights were prepared, which were random small values that did not have effect on movements at the 1st control trial (before IDM learning). Therefore, a total of 45 learning tasks were tested on 6 subject models with all controllers (without ISM, using ISM trained with 2 trajectories, and using ISM trained with 4 trajectories). Iteration number of IDM learning was fixed at 50.

4. Results

The ISM was evaluated by feedforward control of positioning. Target position for the control was set by a pair of dorsi/palmar flexion and radial/ulnar flexion angles at every 2 deg in the range of 20 deg in dorsi- and palmar flexions and in the range of 16 deg in radial and ulnar flexions. An example of the evaluation result of the ISM is shown in Figure 5. In the case of using 2 target trajectories for obtaining training data (ISM-2), the error did not reduce around the center of the target trajectory and at positions between training data. As for the 4 trajectories for training data (ISM-4), the errors were small inside the largest target trajectory. Larger target joint angles outside the largest trajectory could not be controlled appropriately with both ISM-2 and ISM-4.

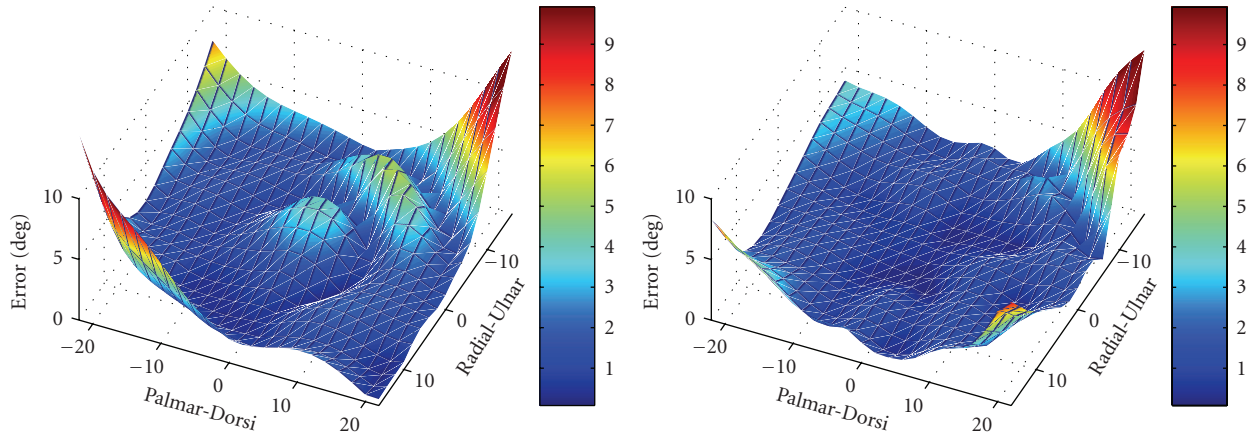
Figure 6 shows average errors in open loop control of the positioning for ISM-2 and ISM-4. There was no large difference in the error between 6 subject models. Positioning errors shown in Figure 6(a) are for evaluation including targets outside the largest trajectory, and those in Figure 6(b) show those excluding targets outside the largest trajectory. Average positioning errors inside the largest trajectory (Figure 6(b)) were smaller than those in Figure 6(a). Figure 6(b) suggests that positioning in the radial/ulnar flexion was not trained sufficiently with the ISM-2.

Figure 7 indicates an example of control result of the FEL-FES controller using the ISM with the IDM. The IDM was trained during the tracking control. The first cycle period of 5 s, which was set for moving to the start position of tracking control, was not used in the IDM learning. Before IDM learning (the 1st control trial), the ISM and the PID controller performed tracking control without the IDM. After IDM learning (the 50th control trial), the FEL-FES controller could perform good tracking with very small outputs of the PID controller.

In order to evaluate performance of the FEL-FES controller, mean error (ME) and power ratio (PR) shown in the following equations were calculated in each learning task:

$$\mathrm{ME} = \frac{\sum_n |e(n)|}{N} = \frac{\sum_n |\theta_d(n) - \theta(n)|}{N} \quad [\mathrm{deg}], \tag{9}$$

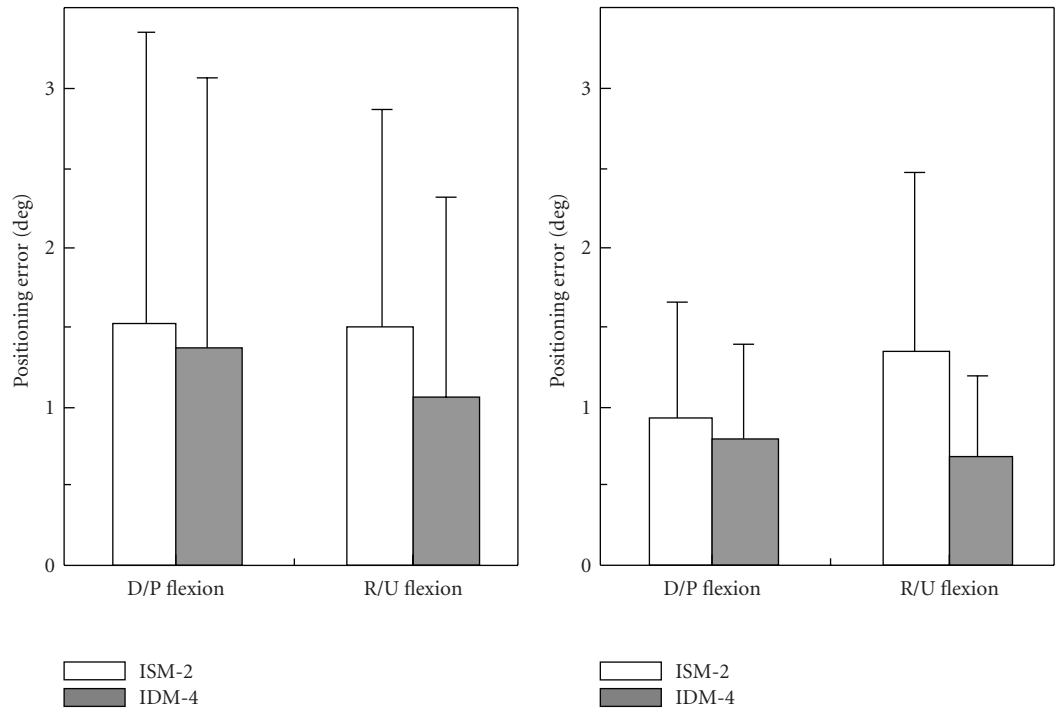
$$\mathrm{PR} = \frac{\sum_n P_{\mathrm{FF}}(n)}{\sum_n P_{\mathrm{FB}}(n) + \sum_n P_{\mathrm{FF}}(n)} \times 100 \quad [\%], \tag{10}$$



(a) ISM trained with 2 target trajectories (ISM-2)

(b) ISM trained with 4 target trajectories (ISM-4)

Figure 5: An example of evaluation results of ISM in positioning control (model A).



(a) including larger targets

(b) excluding larger targets

Figure 6: Evaluation results of ISM learning in positioning control.

where, $e(n)$ represents the error between target joint angle and the resulted one at time n. N is the number of sampled data. $P_{FF}(n)$ and $P_{FB}(n)$ represent the output power of the feedforward and the feedback controllers, respectively. The ME was calculated for each movement direction, and the PR was done for each muscle.

Average values of ME are shown in Figure 8. The controllers using the ISM decreased the error at the 1st control trial (before IDM learning). Especially, the ME was very small for slow movement control. All 3 controllers performed good tracking control after the IDM learning (the 50th control trial). There was no difference in ME after the IDM learning between ISM-2 and ISM-4 and also between with and without the ISM.

The power ratio, PR, gives us information of IDM learning. Figure 9 shows average value, the minimum and the maximum values of the PR. The FEL-FES controller using the IDM and the ISM achieved larger average value and larger minimum value of the PR than those of the previous controller before and after IDM

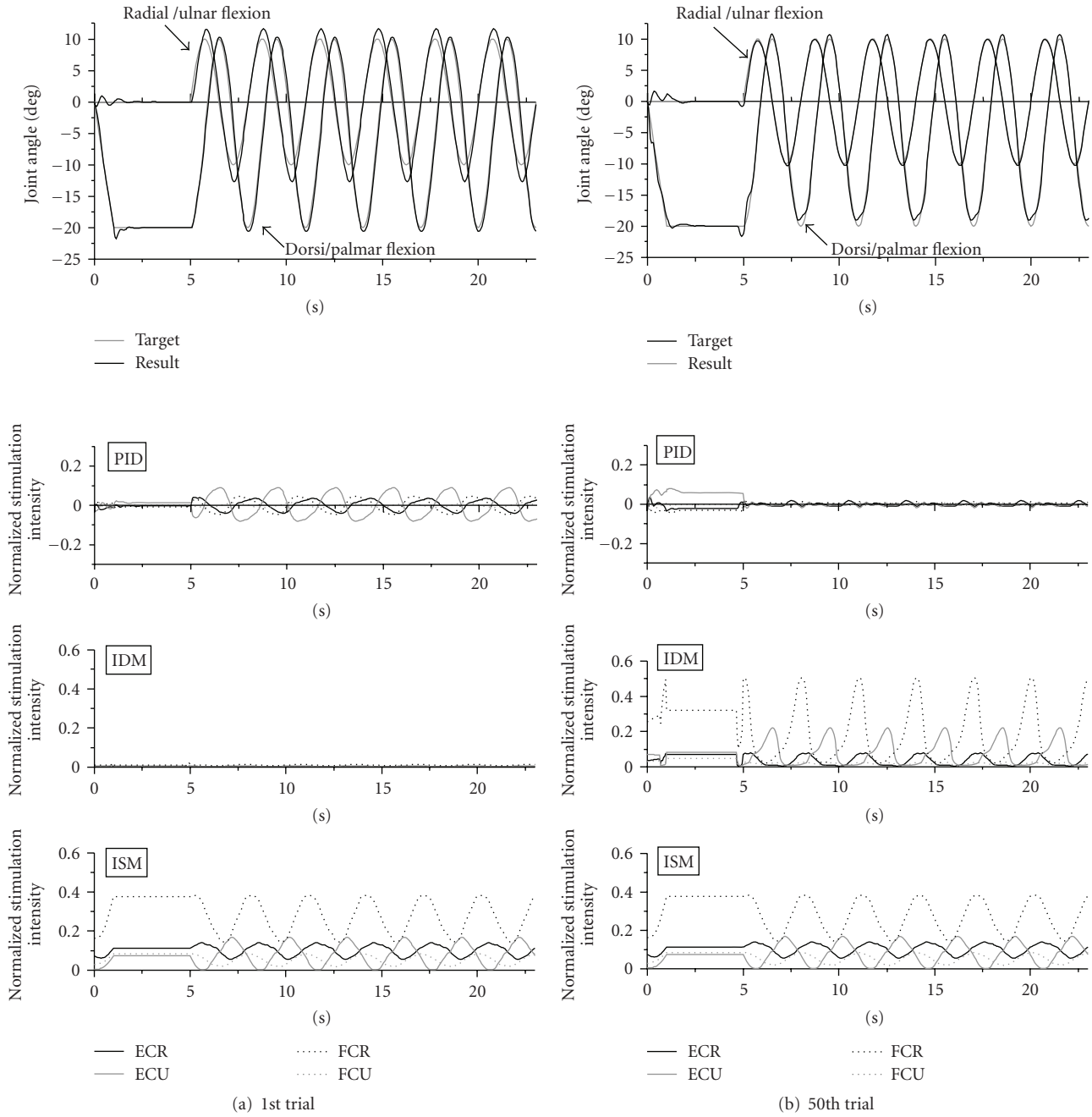


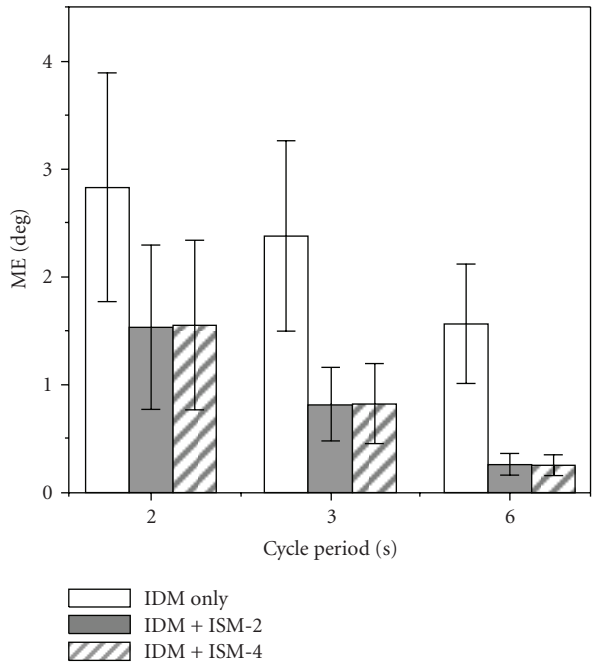
Figure 7: An example of control result by the FEL-FES controller using the ISM and the IDM. (model C, center at palmar position, cycle period of 3 s).

learning. After IDM learning, the minimum value of PR was greatly improved by using the ISM. There was no difference in those improvements between ISM-2 and ISM-4.

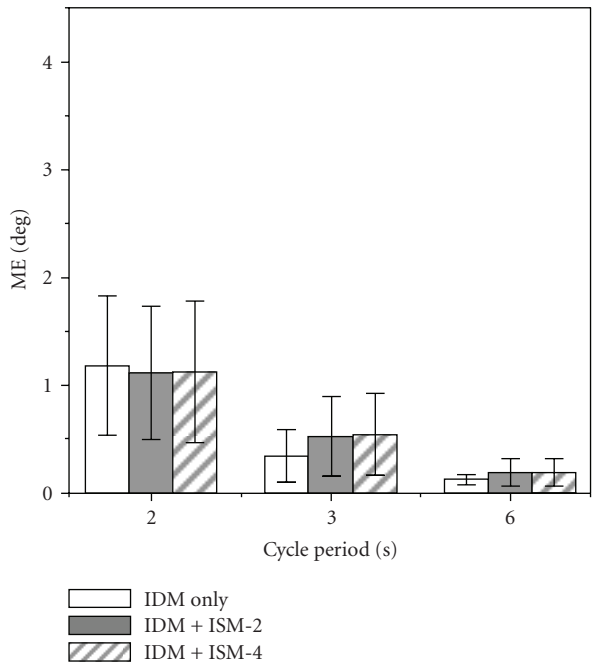
5. Discussion

The off line ISM learning was effectively achieved with the small number of measurements of training data. For practical clinical application, small number of measurements and short period of control time for acquiring the training data are required to avoid muscle fatigue and burden to patients. Therefore, training data acquired from feedback FES control of very slow continuous movements can be useful in ISM learning for FES.

Increasing the number of target trajectories to obtain training data may be required for learning the ISM of the musculoskeletal system that has nonlinear characteristics. However, if the ISM is mainly used to improve learning performance of the IDM, it is possible to decrease the

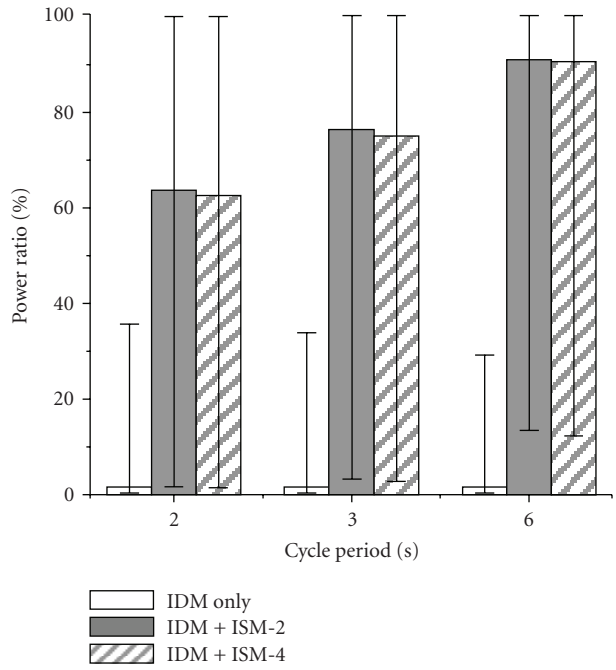


(a) 1st control trial

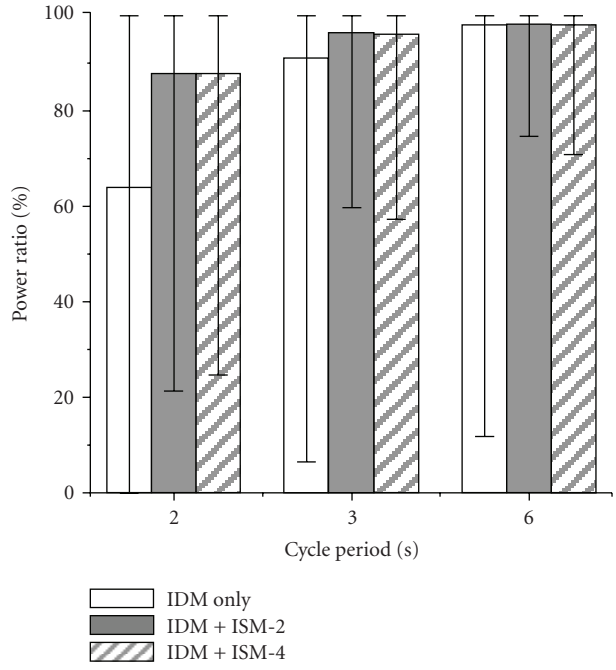


(b) 50th control trial

Figure 8: Average values of the mean error (ME) in tracking control by FEL-FES controllers. Error bar shows the standard deviation.



(a) 1st control trial



(b) 50th control trial

Figure 9: Average values of the power ratio (PR) in tracking control by FEL-FES controllers. Error bar shows the minimum and the maximum values of the PR.

number of measurements of training data because there was no large difference between ISM-2 and ISM-4. On the other hand, target positions that had larger joint angles outside the largest trajectory could not be controlled appropriately as seen in Figure 5. This was a natural result because those targets were outside the training data. Since the control performance of the ISM was improved by adding target trajectories to obtain training data, the ISM is expected to perform properly in the range of motion if the training data that cover the range of motion are added.

The output power of the feedforward controller, PR, was increased by using the ISM as shown in Figure 9. More

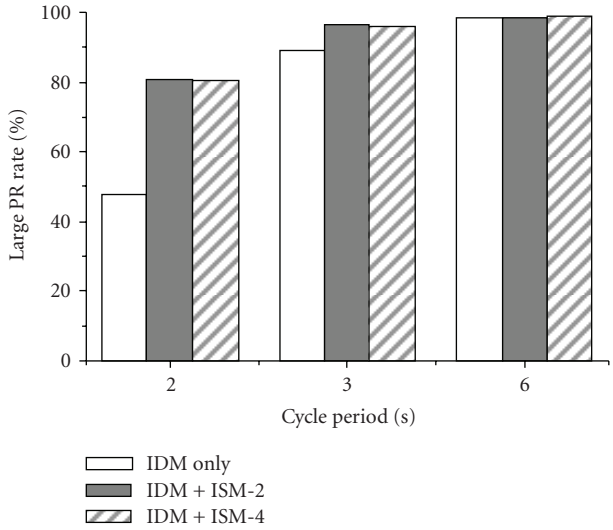


Figure 10: Large PR rate after IDM learning for each FEL-FES controller.

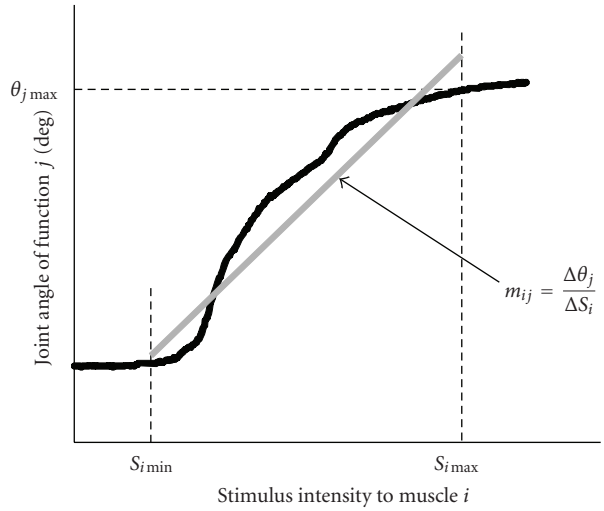


Figure 11: Outline of determination of gain of the musculoskeletal system. The gray solid line shows the approximated linear line of the input-output relationship of the muscle.

than 84% of the number of muscle outputs showed the increase of PR for movements with the cycle period of 2 s. For movements with the cycle period of 3 s and 6 s, it was more than 65% and more than 40%, respectively. These results show that IDM learning was improved in most of learning tasks. For evaluating the improvement of IDM learning, the large PR rate that was defined as the percentage of the number of muscle outputs that had PR larger than 80% was calculated (Figure 10). The large PR rate was also improved by using ISM, especially for fast movement control. These results suggest that the FEL-FES controller using the ISM can be effective to realize a feedforward controller by learning nonlinear characteristics of the musculoskeletal system to electrical stimulation. For practical applications of the FEL to FES, an effective method of IDM learning will be needed, because the musculoskeletal system has nonlinear characteristics and also has hysteresis characteristics.

The FEL-FES controller using the ISM made better control with small values of ME at the first control trial for IDM learning as expected (Figure 8(a)). Since the difference in ME between with and without the ISM was not so large, the feedback controller was considered to perform well. However, control performance of the feedback FES controller sometimes deteriorated in tracking control because of nonlinear characteristics of the musculoskeletal system to electrical stimulation [16] although the feedback controller has been shown to perform properly [10, 11]. Therefore, the ISM is expected to become useful in controlling before IDM learning.

After IDM learning, all controllers showed small values of ME with no significant difference between the controllers (Figure 8(b)). However, the controllers using ISM resulted in larger average and minimum values of PR than those of the controller without the ISM (Figure 9(b)). This suggests that the PID controller had effect on decreasing errors for the controller without the ISM even after IDM learning while the feedforward controller worked mainly in the controllers using ISM. Therefore, there is a possibility that the controller without ISM has a problem in movement control of the musculoskeletal system that has nonlinear characteristics.

6. Conclusions

Feedback error learning (FEL) controller using the ISM with the IDM was applied to FES control. The FEL-FES controller was examined in controlling 2-DOF movements of the wrist joint through computer simulation. In order to train the ISM in FES application, training data were acquired by controlling very slow movements with the PID controller. The ISM trained off line using the training data obtained by the simple measurement method was found to perform properly in the positioning task. The output power ratio of the feedforward controller in the FEL-FES controller was increased by using the ISM showing improvement of IDM learning. The FEL-FES controller using ISM would be useful in realizing feedforward controller for controlling musculoskeletal system that has nonlinear characteristics to electrical stimulation and therefore expected to be useful in applying to hybrid FES system.

Appendix

A. Calculation of Gain of Feedback Controller

The transformation matrix $\mathbf{M}$ was obtained as follows (see Figure 11). First, the input (stimulus intensity)-output (joint angle) characteristics of each muscle were measured by applying electrical stimulation, in which stimulation intensity was increased very slowly. Then, the minimum ($S_{i\min}$) and the maximum ($S_{i\max}$) stimulus intensities for FES control were determined, and the characteristics were approximated to a linear line between these intensities by the least square method. The slope of the approximated line was used as an element of the matrix $\mathbf{M}$, m_{ij}. Here,

the input-output relationship of the musculoskeletal system was represented approximately by using experimentally determined constant matrix **M**:

$$\Delta\Theta = \mathbf{M}\Delta S. \tag{A.1}$$

In case of controlling 2-DOF movements stimulating 4 muscles, the following equation is obtained:

$$\begin{pmatrix}\Delta\theta_1\\ \Delta\theta_2\end{pmatrix} = \begin{pmatrix} m_{11} & m_{21} & m_{31} & m_{41}\\ m_{12} & m_{22} & m_{32} & m_{42}\end{pmatrix}\begin{pmatrix}\Delta S_1\\ \Delta S_2\\ \Delta S_3\\ \Delta S_4\end{pmatrix}, \tag{A.2}$$

where $\Delta\theta_1$ and $\Delta\theta_2$ show change of joint angles of dorsi/palmar flexion and radial/ulnar flexion, respectively. ΔS_i means change of stimulation intensity to muscle i.

The matrix **M** is not the square matrix in general because the number of muscles stimulated is larger than that of degree-of-freedom of movement controlled. Therefore, the generalized inverse matrix of the matrix **M**, $\mathbf{M}^-$, was calculated. That is,

$$\Delta S = \mathbf{M}^-\Delta\Theta, \tag{A.3}$$

$$\begin{pmatrix}\Delta S_1\\ \Delta S_2\\ \Delta S_3\\ \Delta S_4\end{pmatrix} = \begin{pmatrix} m_{11}^- & m_{12}^-\\ m_{21}^- & m_{22}^-\\ m_{31}^- & m_{32}^-\\ m_{41}^- & m_{42}^-\end{pmatrix}\begin{pmatrix}\Delta\theta_1\\ \Delta\theta_2\end{pmatrix}. \tag{A.4}$$

Since there are many generalized inverse matrices for **M**, the generalized inverse matrix $\mathbf{M}^-$ has to be determined uniquely.

Here, after changing negative sign of m_{ij} into positive one, the calculation of the generalized inverse matrix can be solved as the quadratic programming problem using (A.5) as the objective function under the constraints shown by (A.6) and (A.7) [17]

$$L = \sum_i \sum_j \left(m_{ij}^-\right)^2, \tag{A.5}$$

$$\begin{pmatrix} m_{11} & m_{21} & m_{31} & m_{41}\\ m_{12} & m_{22} & m_{32} & m_{42}\end{pmatrix}\begin{pmatrix} m_{11}^- & m_{12}^-\\ m_{21}^- & m_{22}^-\\ m_{31}^- & m_{32}^-\\ m_{41}^- & m_{42}^-\end{pmatrix} = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}, \tag{A.6}$$

$$m_{ij}^- > 0. \tag{A.7}$$

This type of the quadratic programming problem can be converted to the linear programming problem by the Wolfe's algorithm [18]. The unique solution of such linear programming problem can be obtained after the finite number of iterative calculations by the simplex method [18]. That is, a set of positive values of m_{ij}^- minimizing the value L can be calculated under the condition of $\mathbf{M}\mathbf{M}^- = \mathbf{I}$ after changing negative sign of m_{ij} into positive one. Finally, the sign of m_{ij}^- was changed to negative sign based on the sign of m_{ij}.

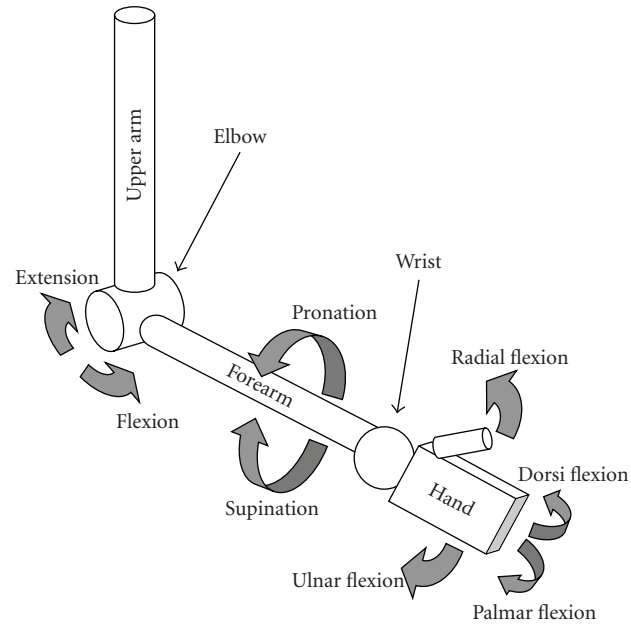


Figure 12: Skeletal model structure of the upper limb.

B. Musculoskeletal Model for FES Control

In this study, the 2-DOF wrist joint movements (dorsi/palmar flexions and radial/ulnar flexions) were controlled stimulating the flexor carpi radialis (FCR), the flexor carpi ulnaris (FCU), the extensor carpi radialis longus/brevis (ECRL/B), and the extensor carpi ulnaris (ECU). Since the four stimulated muscles also relate to forearm or elbow movements, the skeletal model structure of the upper extremity was constructed in order to represent elbow flexion/extension, forearm pronation/supination, and wrist dorsi/palmar flexions and radial/ulnar flexions as shown in Figure 12. The shoulder joint was designed to be fixed at arbitrary angles of flextion/extention and rotation. The 15 muscles relating these movements as the agonist were included as listed in Table 1. Some muscles were also modeled as the synergistic muscles for other movements.

The musculoskeletal model to predict responses of electrically stimulated muscles is outlined in Figure 13. Muscle force F_{CE} produced by electrical stimulation was described by the Hill type muscle model including muscle activation level determined by electrical stimulation $a_m(s)$, length-force relationship $k(l)$, velocity-force relationship $h(v)$, and maximum muscle force F_{max}. That is,

$$F_{CE} = a_m(s)k(l)h(v)F_{max} \tag{B.1}$$

where s, l, and v were normalized stimulation intensity, muscle length, and contraction velocity, respectively. Active torque τ_{CE} produced by electrical stimulation was calculated by muscle force F_{CE} and moment arm $r_f(\theta)$. That is,

$$\tau_{CE} = F_{CE}\, r_f(\theta). \tag{B.2}$$

Moment arm $r_f(\theta)$ was represented by an approximated polynomial equation as a nonlinear function of joint angle θ

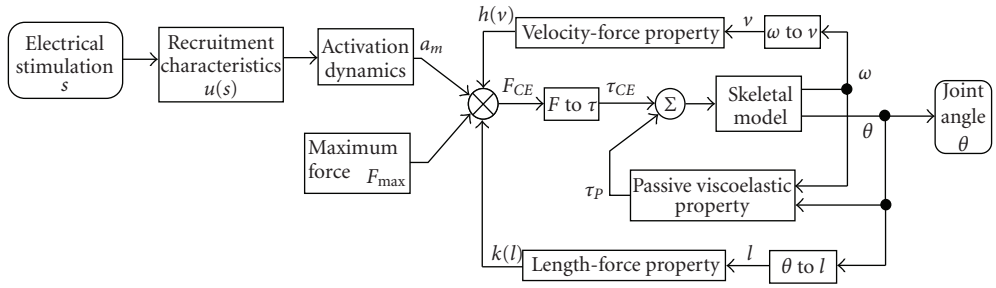


Figure 13: Outline of the musculoskeletal model for FES.

Table 1: Muscles included into the model.

Joint	Movement	Agonist muscle	Synergistic muscle
Elbow	flexion	biceps brachii long head	flexor carpi radialis
		biceps brachii short head	extensor carpi radialis longus
		Brachialis	extensor carpi radialis brevis
		brachioradialis	pronator teres
	extension	triceps brachii long head	extensor carpi ulnaris
		triceps brachii medial head	
		triceps brachii lateral head	
Forearm	pronation	pronator quadratus	
		pronator teres	
	supination	biceps brachii long head	Brachioradialis
		biceps brachii short head	
		Supinator	
Wrist	palmar flexion	flexor carpi radialis	
		flexor carpi ulnaris	
	dorsi flexion	extensor carpi radialis longus	
		extensor carpi radialis brevis	
		extensor carpi ulnaris	
	radial flexion	extensor carpi radialis longus	
		extensor carpi radialis brevis	
		flexor carpi radialis	
	ulnar flexion	extensor carpi ulnaris	
		flexor carpi ulnaris	

for each movement developed by each muscle [15]. For example, the moment arm for the wrist dosri/palmar flexion and elbow flexion/extension was described by the following equation:

$$r_f(\theta) = a_0 + a_1\theta + a_2\theta^2 + a_3\theta^3 + a_4\theta^4 + a_5\theta^5 \quad \text{(B.3)}$$

where $a_0 \sim a_5$ were parameters for each movement of each muscle. Each element of the F_{CE} is described in the following.

The nonlinear recruitment property of electrically stimulated muscle $u(s)$ was modeled by the following [19]:

$$u(s) = s_c \ \tanh\{s_h(s - x_c)\} + y_c \quad \text{(B.4)}$$

where s_c, s_h, x_c, and y_c were constants. Electrical stimulation was expressed in normalized stimulation intensity s. The muscle activation a_m was described by the following dynamics using the recruitment property with different two time constants, t_r and t_f [20]:

$$\frac{da_m}{dt} = \frac{1}{t_r}\{u(s) - a_m\}u(s) + \frac{1}{t_f}\{u(s) - a_m\}. \quad \text{(B.5)}$$

The length-force relationship $k(l)$ was described by the following equation. l_o means optimum muscle length [21]:

$$k(l) = 1 - \left(\frac{l - l_{\text{o}}}{0.5l_{\text{o}}}\right)^2. \quad \text{(B.6)}$$

The velocity-force relationship $h(v)$ during shortening and lengthening of muscle was modeled. $v_{\max}$ shows

maximum contraction velocity [21, 22]:

$$h(v) = \frac{v_{\max} - v}{v_{\max} + 2.5v} \quad (v \le 0 : \text{shortening}),$$
$$h(v) = 1.3 - 0.3\,\frac{v_{\max} + 2.5v}{v_{\max} - 2.5^2 v} \quad (v > 0 : \text{lengthening}). \tag{B.7}$$

The maximum muscle force produced by electrical stimulation $F_{\max}$ was determined by PCSA (physiological cross-sectional area) as follows [15]:

$$F_{\max} = 2.2\ PCSA. \tag{B.8}$$

The passive viscoelastic element developed passive torque τ_P calculated by the following equation for each joint movement [23]. The range of motion was also represented by this property:

$$\tau_p = k_0\theta + b_0\omega + k_1\{\exp(k_2\ \theta) - 1\} \tag{B.9}$$

where θ and ω were joint angle and angular velocity, respectively. Constants k_0, b_0, k_1, and k_2 were determined for each joint movement.

Acknowledgments

The authors thank Dr. Kenji Kurosawa for his helpful advice on FEL controller for FES. This work was supported in part by the Saito Gratitude Foundation.

References

[1] N. Hoshimiya, A. Naito, M. Yajima, and Y. Handa, "A multichannel FES system for the restoration of motor functions in high spinal cord injury patients: a respiration-controlled system for multijoint upper extremity," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 7, pp. 754–760, 1989.

[2] B. Smith, Z. Tang, and M. W. Johnson et al., "An externally powered, multichannel, implantable stimulator-telemeter for control of paralyzed muscle," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 4, pp. 463–475, 1998.

[3] Y. Handa, K. Ohkubo, and N. Hoshimiya, "A portable multichannel FES system for restoration of motor function of the paralyzed extremities," *Automedica*, vol. 11, pp. 221–231, 1989.

[4] K. A. Ferguson, G. Polando, R. Kobetic, R. J. Triolo, and E. B. Marsolais, "Walking with a hybrid orthosis system," *Spinal Cord*, vol. 37, no. 11, pp. 800–804, 1999.

[5] R. Kobetic, C. S. To, and J. R. Schnellenberger et al., "Development of hybrid orthosis for standing, walking, and stair climbing after spinal cord injury," *Journal of Rehabilitation Research and Development*, vol. 46, no. 3, pp. 447–462, 2009.

[6] C. R. Kinnaird and D. P. Ferris, "Medial gastrocnemius myoelectric control of a robotic ankle exoskeleton," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 17, no. 1, pp. 31–37, 2009.

[7] G. S. Sawicki and D. P. Ferris, "A pneumatically powered knee-ankle-foot orthosis (KAFO) with myoelectric activation and inhibition," *Journal of NeuroEngineering and Rehabilitation*, vol. 6, no. 1, article 23, 2009.

[8] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biological Cybernetics*, vol. 57, no. 3, pp. 169–185, 1987.

[9] H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki, "Feedback-error-learning neural network for trajectory control of a robotic manipulator," *Neural Networks*, vol. 1, no. 3, pp. 251–265, 1988.

[10] T. Watanabe, K. Iibuchi, K. Kurosawa, and N. Hoshimiya, "A method of multichannel PID control of two-degree-of-freedom wrist joint movements by functional electrical stimulation," *Systems and Computers in Japan*, vol. 34, no. 5, pp. 25–36, 2003.

[11] K. Kurosawa, T. Watanabe, R. Futami, N. Hoshimiya, and Y. Handa, "Development of a closed-loop FES system using 3-D magnetic position and orientation measurement system," *Journal of Automatic Control*, vol. 12, no. 1, pp. 23–30, 2002.

[12] K. Kurosawa, R. Futami, T. Watanabe, and N. Hoshimiya, "Joint angle control by FES using a feedback error learning controller," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 13, no. 3, pp. 359–371, 2005.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[14] M. A. Arbib, *The Handbook of Brain Theory and Neural Networks*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2002.

[15] M. A. Lemay and P. E. Crago, "A dynamic model for simulating movements of the elbow, forearm, and wrist," *Journal of Biomechanics*, vol. 29, no. 10, pp. 1319–1330, 1996.

[16] T. Watanabe, T. Matsudaira, N. Hoshimiya, and Y. Handa, "A test of multichannel closed-loop FES control on the wrist joint of a hemiplegic patient," in *Proceedings of the 10th Annual Conference of the International FES Society*, pp. 56–58, 2005.

[17] K. Kurosawa, H. Murakami, T. Watanabe, R. Futami, N. Hoshimiya, and Y. Handa, "A study on modification method of stimulation patterns for FES," *Japanese Journal of Medical Electronics and Biological Engineering*, vol. 34, no. 2, pp. 103–110, 1996.

[18] S. I. Gass, *Linear Programming: Methods and Applications*, McGraw-Hill, New York, NY, USA, 1969.

[19] M. Levy, J. Mizrahi, and Z. Susak, "Recruitment, force and fatigue characteristics of quadriceps muscles of paraplegics isometrically activated by surface functional electrical stimulation," *Journal of Biomedical Engineering*, vol. 12, no. 2, pp. 150–156, 1990.

[20] M. G. Pandy, B. A. Garner, and F. C. Anderson, "Optimal control of non-ballistic muscular movements: a constraint-based performance criterion for rising from a chair," *Journal of Biomechanical Engineering*, vol. 117, no. 1, pp. 15–26, 1995.

[21] B. M. Nigg and W. Herzong, *Biomechanics of the Musculo-Skeletal System*, John Wiley & Sons, New York, NY, USA, 1995.

[22] G. M. Eom, T. Watanabe, R. Futami, N. Hoshimiy, and Y. Handa, "Computer-aided generation of stimulation data and model identification for functional electrical stimulation (FES) control of lower extremities," *Frontiers of Medical and Biological Engineering*, vol. 10, no. 3, pp. 213–231, 2000.

[23] J. M. Winters and L. Stark, "Analysis of fundamental human movement patterns through the use of in-depth antagonistic muscle models," *IEEE Transactions on Biomedical Engineering*, vol. 32, no. 10, pp. 826–839, 1985.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 301923, 14 pages
doi:10.1155/2010/301923

Research Article

An Extensible Dialogue Script for a Robot Based on Unification of State-Transition Models

Yosuke Matsusaka,[1] Hiroyuki Fujii,[2] and Isao Hara[1]

[1] *National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, Ibaraki, 305-8568, Japan*
[2] *Japan Science and Technology Agency (JST), 2-1-6 Sengen, Tsukuba, Ibaraki, 305-0047, Japan*

Correspondence should be addressed to Yosuke Matsusaka, yosuke.matsusaka@aist.go.jp

Received 1 November 2009; Revised 23 February 2010; Accepted 17 May 2010

Academic Editor: Noriyasu Homma

We propose extension-by-unification method to improve reusability of the dialogue components in the development of communication function of the robot. Compared to previous extension-by-connection method used in behavior-based communication robot developments, the extension-by-unification method has the ability to decompose the script into components. The decomposed components can be recomposed to build a new application easily. In this paper, first we, explain a reformulation we have applied to the conventional state-transition model. Second, we explain a set of algorithms to decompose, recompose, and detect the conflict of each component. Third, we explain a dialogue engine and a script management server we have developed. The script management server has a function to propose reusable components to the developer in real time by implementing the conflict detection algorithm. The dialogue engine SEAT (Speech Event-Action Translator) has flexible adapter mechanism to enable quick integration to robotic systems. We have confirmed that by the application of three robots, development efficiency has improved by 30%.

1. Introduction

In recent years, there has been an increasing demand for robots that work in a human life environment.

Replacement of human labor by robots in the manufacturing sectors (e.g., factory production lines) has already shown success. In the case of manufacturing robots, professional operators give commands to the robot. Professional operators have expert knowledge, and they are able to command the robot in a robot-friendly manner.

However, in the case of the robots used in a life environment, the operator who gives commands to the robot only has imperfect knowledge about the robot (called a "naïve user" hereafter). Naïve users often use natural language to command the robot. To create a robot that can be easily used by naïve users, the robot not only needs to have mechanical skills but also linguistic ability to understand a variety of commands.

The biggest problem in understanding language is diversity. Words used by a naïve user to command the robot will be diverse for various reasons (described in Section 3). This problem has been solved commonly by two methods: the machine learning methods and the behavior-based "scripting" methods. Each method has advantages and disadvantages.

An advantage of using the machine learning method is that the developer can implement the vast patterns of language understanding without any programming effort. For example, Iwahashi has used Markov model and stochastic context-free grammar to let the robot understand lexicons as well as associations between objects and words [1]. Roy has implemented on-line learning algorithm on a robotic platform, which automatically acquires the concept of the words and the objects [2]. However, the disadvantage of this method is that the models generated by the machine learning method cannot be edited or modified for reuse. Some methods enable retraining of the model by controlling a meta-level learning parameter (e.g., [3]), but we need to realize intended behaviors in complex situations, so it becomes generally difficult to find optimal learning parameters.

In contrast, in the case of scripting methods, the developer can program the specific behavior of the robot as intended. While the disadvantage is, however, the difficulty to cover the diversty of language understanding ability required in each application, because the effort of human developer is limited.

SHRDLU [4] is one of the most successful applications based on scripting method. The system was developed by Winograd in 1972. The system uses "inference-based" scripting approach. The script consists of planning part and vocabulary part and uses inference to complement the meaning of words.

The inference-based scripting approach is useful for the developer who has deep understanding about the inference system, but this requirement is sometimes difficult to fulfill in collaborative and incremental development (discussed later in Section 7.1).

Recently, "behavior-based" scripting method has been applied in many practical robotic systems. The application presented by Brooks [5] used hieratical structure model. The recent applications [6, 7] use state-transition model (finite state automata) to model the situation of the system. The developer incrementally develops the script by adding each behavior which fits to each small situation. Diverse situation understanding ability can be realized as a result of long-term incremental development.

The behavior-based scripting method can also be applied to communication robots by incorporating speech input with the situation model. Application of the behavior-based scripting method to the communication robot is first presented by Kanda et al. [8] in 2002. In their work, they not only proposed an incremental development framework, but also implemented an on-line development environment which can realize automated control of the robot. They have confirmed through a 25-day field study that with the help of the development environment, the conversation ability of the robot was incremented on line and succeeded to decrease the operation time of the human operator [9].

However, in the existing behavior-based scripting methods for communication robot, there is an inefficiency in terms of reusing the script to develop different types of robots (this problem is described in Section 3.1). In this paper, we present the extension-by-unification method in order to push forwards the behavior-based scripting approach to develop communication robots.

In our approach, we will not only focus on the ability of the model itself, but also on the descriptive format of the script and its operation. We show that the reuse can be enhanced by reformulating the conventional descriptive format and also show the effectiveness of the reformulation by implementing a computer-assisted development environment to enhance the development activity of the developer.

In Section 2, we give an overview of a basic state-transition model and its characteristics.

In Section 3, a formal discussion of incremental development methods for the state-transition model is presented. Here, we introduce the formalization of the proposed incremental development method and clarify its characteristics by comparing it to the previous method.

In Sections 4 and 5, the implementations of the script server and script engine are presented. The script engine and script server implemented support functions that will allow developers to reduce their development efforts.

In Section 6, examples of script development in actual applications are presented, and the effectiveness of the development environment is discussed.

2. State-Transition-Based Models

2.1. Formalization. A state-transition model is a modeling method in which the input and output of the system assume the following form:

$$A := \langle I, S, O, \gamma, \lambda, s_0 \rangle, \tag{1}$$

where I represents the input alphabet, O represents the output alphabet, S represents the internal states, γ represents the state-transition function, λ represents the output function, and s_0 is the initial state.

The state transition function γ is defined in association with the state to the input.

$$\gamma : S \times I \longrightarrow S. \tag{2}$$

The output function λ is defined in association with the state to the input.

$$\lambda : S \times I \longrightarrow O. \tag{3}$$

When the system is in state s_t and gets input alphabet i_t, state transition to s_{t+1} will occur as follows:

$$s_{t+1} = \gamma_{s_t, i_t}. \tag{4}$$

At the same time, we get output alphabet o_t as follows:

$$o_{t+1} = \lambda_{s_t, i_t}. \tag{5}$$

Even the input to the system is the same, the output of the system may be different, because the internal state s_t will be updated each time the system gets the input.

We have explained the state-transition model in an equation form, however, the state-transition model can be also presented in a 2-dimensional diagram called "state-transition diagram". In the diagram, each state is represented by a circle, and the transition between states is represented by arrows. In this paper, we annotate the transition conditions and the associative actions by including text over each arrow. We use a black circle (called a "token") to represent the current state.

For example, Figure 1 represents a conversation modeled by the state-transition model.

In the model presented in Figure 1, the initial state of the system is in "TV control" state. When the model gets the instruction "Turn on" as an input, it will output the command "turn-on-TV", and state transition "(a)" will occur. Then the token turns back to the same "TV control" state. When the model gets the instruction "Video" as an input, state transition "(b)" will occur, and the token will

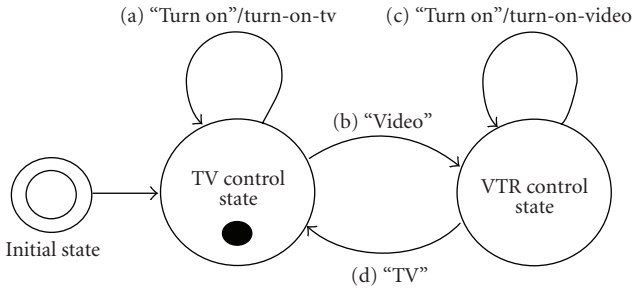


Figure 1: Example of state-transition model.

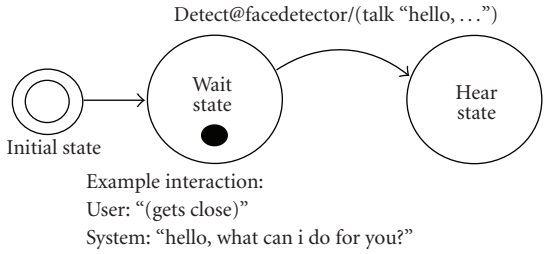


Figure 2: Example of state-transition model using multichannel input.

move to "VTR control" state. This time when the instruction "Turn on" is given, state transition "(c)" occurs and outputs the command "turn-on-video". In this way, we can model the context by defining an appropriate state and state transitions between the states.

The above example is expressed as follows in the equation form:

$$
\begin{aligned}
A &:= \langle I, S, O, \gamma, \lambda, s_0 \rangle, \\
I &= (\text{"Turnon"}, \text{"Turnoff"}, \text{"TV"}, \text{"Video"}), \\
S &= (\text{"tv-control"}, \text{"vtr-control"}), \\
O &= (\text{"turn-on-tv"}, \text{"turn-on-video"}, \\
&\qquad \text{"turn-off-tv"}, \text{"turn-off-video"}), \\
\gamma &= \begin{pmatrix} s_0 & s_0 & s_0 & s_1 \\ s_1 & s_1 & s_0 & s_1 \end{pmatrix}, \\
\lambda &= \begin{pmatrix} o_0 & o_2 & \text{none} & \text{none} \\ o_1 & o_3 & \text{none} & \text{none} \end{pmatrix}.
\end{aligned}
\tag{6}
$$

As we have seen here, the expression in equation form has an advantage in formalization, while the expression in diagram form has an advantage in quick understanding. In later discussion, we will use both the equation and the diagram forms to explain the concept quickly and formally.

State-transition model is a very simple get very powerful modeling method and has been applied to very wide applications. Because the structure of state-transition model is very simple, it is frequently misunderstood that the state-transition model can only model simple behavior. However, it can model diverse behavior by applying some extensions (e.g., [10, 11]).

2.2. Extensions

2.2.1. Multichannel Input. The original state-transition model uses a single input channel. In the case of a conversational system, the input channel is assigned to receive input from the speech recognition subsystem. However, it can accept multichannel input by formulating the transition function γ as $\gamma : S \times I \times C \rightarrow S$ and the output function λ as $\lambda : S \times I \times C \rightarrow \langle O, C \rangle$, where C is the type of input channel. By this extension, the model can integrate voice input as well as the other sensory inputs.

The example in Figure 2 shows the use of context in image and voice input.

2.2.2. Loop-Back Events. The state-transition model updates its internal state using external input. But by connecting output of the system to the input, it can realize autonomous behavior generation based on the internal event (in this paper, we call this a "loop-back event"). Loop-back events are important in realizing the autonomous behavior of the robot (examples are presented in Section 6).

2.2.3. Automatic Generation of Frame-Based Questions. A frame-based question is an interaction that requires answers to two or more questions in an arbitrary order. Example in Figure 3 shows realization of frame-based question using state-transition model. The structure of the model is apparently complex; however, we can generate this model using a simple algorithm.

2.3. Existing Implementations Used in Industry. There have been many script engines implemented (e.g., [12]). Most of them implement both multichannel and loop-back event extensions.

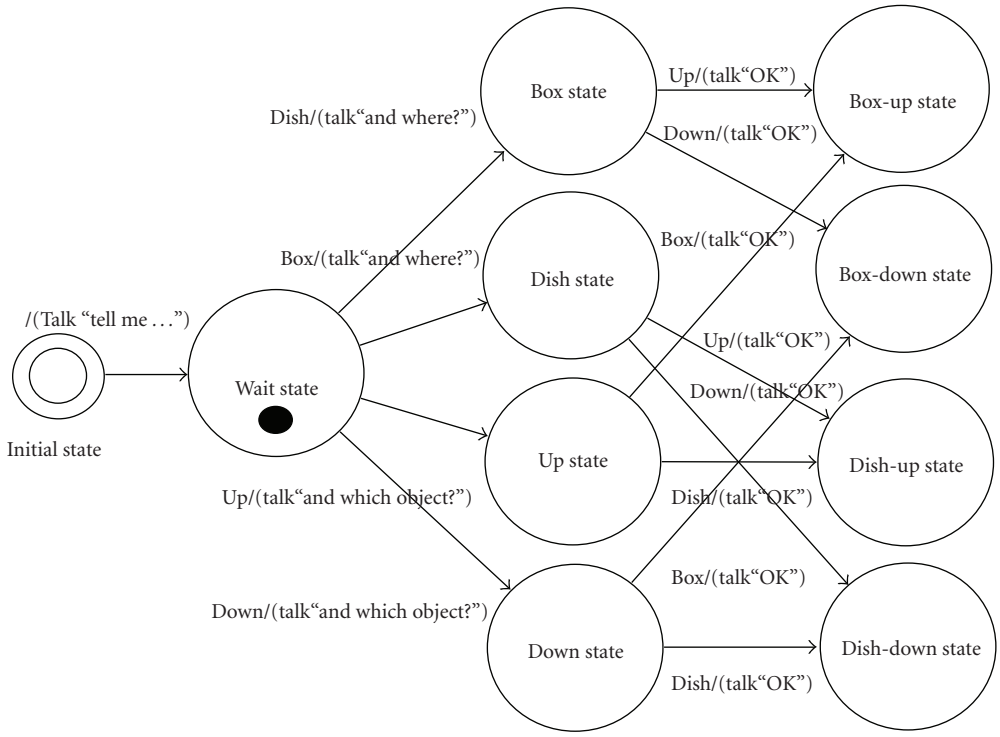
VoiceXML [13] is the de-facto standard of the script format used in various dialogue systems. It uses a more sophisticated format to describe the script than the state-transition model does. However, as we have shown the example of frame-based question in the previous section, we can easily convert the sophisticated description into the state-transition-based form. The implementation of VoiceXML script engines also uses this conversion, and the core part of these engines are based on a state-transition model.

Our script engine does not only implement the above extensions, but also has a function to support incremental development. In the next section, we discuss our incremental development method.

3. Incremental Development Method

Commands given by the human to the robot are diverse. The following are the factors that cause this diversity.

The Nature of Language. Human language is ambiguous, and different expressions can be used to give instructions that carry the same meaning.



Example interaction:
System: “tell me which object to move and which direction to move.”
User: “box.”
System: “and where?”
User: “up.”

```
(q1, q1n) = (("box", "dish"), "where")
(q2, q2n) = (("up", "down"), "which object")
func generate_2frame_question(q1, q1n, q2, q2n):
  for q1st, qn, q2nd in ((q1, q1n, q2), (q2, q2n, q1)):
    for quest1 in q1st:
      state[start].rule.push(quest1, "And "+qn+"?",
                                        state[quest1])
      for quest2 in q2nd:
        state[quest1].rule.push(quest2, "OK",
                                  state[quest1+quest2])
```

Figure 3: Example of state-transition model that can realize a 2-frame question. The structure of the model looks complex, but it can be generated easily using a simple algorithm.

Tasks. Robots working in a life environment have to accept a variety of tasks. In order to cope with this, it is necessary for them to understand a variety of commands.

Ability of the Robot Itself. The diversity is also caused by the ability of the robot itself. A command from a human becomes effective due to the functions of the robot. For example, humans do not say “walk N steps” to a robot on wheels.

The language comprehension system of the robot must be able to deal with these diversities.

In the script-based development approach, diversity has been dealt with by stacking a newly developed script onto the existing scripts. By accumulating a number of scripts, the developer can accumulate the number of commands that the system can deal with.

Incremental development of the state-transition model has previously been conducted using the “extension-by-connection” method (described in the next section). In

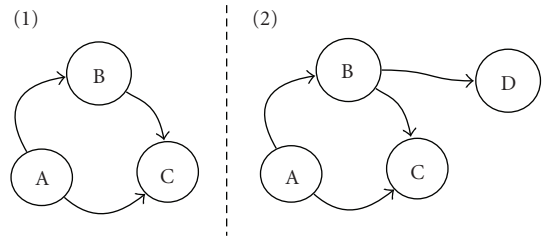


Figure 4: Extension of a state machine using the extension-by-connection model.

this section, we propose an "extension-by-unification" method that can cope with the diversities mentioned above (described in Section 3.3).

3.1. Extension-by-Connection Method. The simplest way to extend state-transition model is as follows.

(1) Add a new state to the existing state-transition model.

(2) Add a new transition from the existing state to the new state.

This process is illustrated in Figure 4.

Here, we formulate the above process. Let the existing state-transition model be A, and the accumulated state-transition model be A'.

As explained in Section 2.1, the existing state-transition model A can be represented by the following form.

$$A := \langle I, S, O, \gamma, \lambda, s_0 \rangle. \tag{7}$$

Here, S is the set of state $s \in S$. The transition function γ can be defined in any form. In this paper, we use the matrix of $S \times I$, in which the transition from state s_t to state s_{t+1} can occur if $\gamma_{S_t,i} = s_{t+1}$.

Similarly, we define the accumulated state-transition model A' as follows:

$$A' := \langle I, S', O, \gamma', \lambda', s_0' \rangle. \tag{8}$$

Then, the new state ΔS can be calculated as follows:

$$S' = S \cup \Delta S. \tag{9}$$

Here, $S' \cap \Delta S = \varnothing$.

The new state transition $\Delta\gamma$ can be calculated as follows:

$$\gamma'_{s_t,i} = \gamma_{s_t,i} \quad (s_t \in S, i \in I), \tag{10}$$

$$\gamma'_{s_t',i} = \Delta\gamma_{s_t',i} \quad (s_t' \in S', i \in I), \tag{11}$$

$$\lambda'_{s_t,i} = \lambda_{s_t,i} \quad (s_t \in S, i \in I), \tag{12}$$

$$\lambda'_{s_t',i} = \Delta\lambda_{s_t',i} \quad (s_t' \in S', i \in I). \tag{13}$$

The transition function of the accumulated part $\Delta\gamma$ needs to be defined based on the transition from the existing state S. Therefore, $\Delta\gamma$ will be a matrix of $S' \times I$. Note that the new state ΔS can be expressed only by the newly defined part, but the transition of the accumulated part $\Delta\gamma$ includes both old state S and new state ΔS in its definition.

The state-transition model is easy to understand in drawing a state-transition diagram. Extension-by-connection can also be carried out very easily by editing this diagram. There are several GUIs that can add state-transition rules through the operation of mouse clicks (e.g., [14]).

3.2. Problems with the Extension-by-Connection Method. Extension-by-connection is a useful method, but it has the following problems.

As we can see in (9) and (11), the definition of $\Delta\gamma'$ requires both S and ΔS. This causes problems in the function development of robots. For example, let us consider the following scenario.

(1) Robot "A" has function A, and we have already developed a state-transition model A^A to realize the function.

(2) For the robot "A" to accumulate function C, we have extended the state-transition model to A^{AC}.

(3) We have developed another robot, "B", which has function B. And we want to add function C to this robot.

Here, the state-transition model for function C is already developed for robot A. We want to reuse the model for robot B. Here, we discuss whether such a diversion would be possible.

First, the state S^{AC} is easily separable from state S^A and state S^C

$$S^C = S^{AC} - S^A. \tag{14}$$

However, the definition of state-transition function γ^{AC} is as follows:

$$\begin{aligned} \gamma^{AC}_{s_t^A,i} &= \gamma^A_{ij} \quad \left(s_t^A \in S^A, i \in I\right), \\ \gamma^{AC}_{s_t^{AC},i} &= \gamma^C_{ij} \quad \left(s_t^{AC} \in S^{AC}, i \in I\right). \end{aligned} \tag{15}$$

γ^C contains state S^A in its definition.

Because states S^A and S^B are defined for different types of robots, A and B are not equal. In addition, because the transition for the function C is defined dependently on state S^A, we cannot replace variables like $S^{AC} = S^{BC}$, which means that we cannot use γ^C to extend the state-transition model A^B. The state transition of function C developed for robot A cannot be diverted for the extension of robot B.

Ideally, once a feature is developed, it would be possible to share with other robots that need the same feature. In order to achieve this, we introduce the extension-by-unification method.

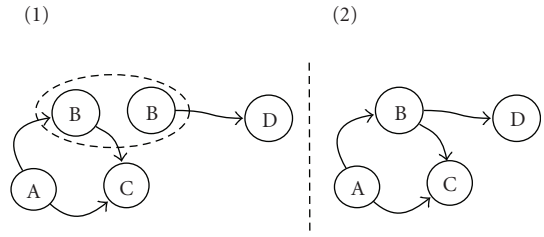


Figure 5: Extension of the state-transition model using the extension-by-unification method.

3.3. Extension-by-Unification Method. In the extension-by-unification method, we extend the state-transition model by the following procedure.

(1) Develop a state-transition model to realize a new function.

(2) Unify a state with the same ID between the existing and the new state-transition models.

This process is illustrated in Figure 5.

Here, we formulate the above process.

The existing state-transition model A can be represented by state S, state transition γ, and initial state s_0

$$A := \langle I, S, O, \gamma, \lambda, s_0 \rangle. \tag{16}$$

Similarly, the new state-transition model A' is represented as follows:

$$A' := \langle I, S', O, \gamma', \lambda', s_0' \rangle. \tag{17}$$

We accumulate the state-transition model A'' by unifying A and A'. First, we calculate state as follows:

$$S'' = S \cup S'. \tag{18}$$

Here, $S \cap S' \neq \emptyset$.

Next, the transition between the state S'' is calculated as follows:

$$\gamma''_{s_t,i} = \gamma_{s_t,i} \quad (s_t \in S, i \in I), \tag{19}$$

$$\gamma''_{s'_t,i} = \gamma'_{s'_t,i} \quad (s'_t \in S', i \in I), \tag{20}$$

$$\lambda''_{s_t,i} = \lambda_{s_t,i} \quad (s_t \in S, i \in I), \tag{21}$$

$$\lambda''_{s'_t,i} = \lambda'_{s'_t,i} \quad (s'_t \in S', i \in I). \tag{22}$$

By defining initial state s_0'' to be $s_0'' = s_0$, the extended state-transition model A'' will be as follows:

$$A'' = \langle I, S'', O, \gamma'', \lambda'', s_0'' \rangle. \tag{23}$$

As visible in (20), the transition function γ' is an $S' \times S'$ matrix that only includes state S' in its definition. The extension-by-unification method does not require the definition of the original state in the accumulated part of the state-transition model.

As noted in Section 3.2, in the conventional extension-by-connection method, the definition of the accumulated part of the state-transition model depends on information on the existing state. It is limited in terms of reusing scripts for this reason. The proposed extension-by-unification method does not have this problem. Using this method, we can significantly increase the reusability of the state-transition model (examples shown in Section 6).

3.4. Problems of the Extension-by-Unification Method. As discussed above, the extension-by-unification method can overcome a limitation in the extension-by-connection method by applying a simple reformulation. However, as a counterpart to this reformulation, we have dealt with the following problems that do not occur in conventional methods.

First, a conflict in transition conditions may occur. For example, when we try to unify two states with one another, the states may have different actions associated with the same transition conditions. In this case, the state-transition models cannot be unified.

Second, an isolated state may occur. For example, when we try to unify state-transition models that do not have the same state IDs in common, there will be no transitions between the old and the new states. In this case, the developer cannot activate the new function as intended.

In this study, we not only implement a script engine that has a state unification function (detailed in Section 5), but also implement a script-management server that includes conflict detection, isolated state detection, and executability detection functions (detailed in the next section).

4. The Development Environment

4.1. Script-Management Server. We developed the script-management system, which is based on wiki.

The developer can write the script in XML form on the wiki page, and the document of the script can also be written on the same wiki page. The developer can annotate each wiki page using tags. Tags are used as identifiers to indicate multiple pages working as a set.

Algorithm 1 is an example of the state-transition model written in the XML form.

Our run-time engine SEAT can read script using HTTP protocol. Thus, the developer can directly load and run the script (or the set of scripts defined by the tag) by specifying the URL.

The script-management server uses the core functions of dokuwiki (http://www.dokuwiki.org/). Functions described in the next sections are realized by extending the dokuwiki.

4.2. Detection of Isolated State. When we try to unify state-transition models that do not have state IDs in common, there will be no transition between the old and the new states. In this case, the developer cannot activate the new function as intended. Isolation of the state can be detected in Algorithm 2.

```
<state id="Robot" dict="julian-conf/hrp_operate">
 <rule>
   <key>[take] one step [forward]</key>
   <command host="talk">
      (talk "Take one step forward.")
   </command>
   <command host="control">
      (robot hwalk :set-target-pos 0.2 0 0)
   </command>
 </rule>
</state>
```

Algorithm 1

```
fstate = []

func checkisolatedstate_recur(stateid):
  for command in states(stateid).commands:
    if command.type == statetransition:
     if fstate(command.target) == 0:
       fstate(command.target) = 1
       checkisolatedstate_recur(command.target)

func checkisolatedstate(stateid):
  checkisolatedstate_recur(command.target)
  for state in states:
    if fstate(state) != 1
      detected = 1
```

Algorithm 2

4.3. Detection of State-Transition Conflict. When we try to unify two states with one another, the states may have different actions associated with the same transition conditions. In this case, the state-transition models cannot be unified. A conflict between the state-transition conditions can be detected in Algorithm 3.

4.4. Detection of an Unexecutable Action. An "unexecutable action" is an action that is defined in the state-transition model but cannot produce any output because the robot does not have the ability to generate the actual output. In this case, the developer cannot achieve the intended output. By using the instance ID of the adaptor mechanism (described in Section 5.2), an unexecutable action can be detected in Algorithm 4.

4.5. Visualization of Unifiable States. By using the above algorithms, the possibility of unification between scripts can be identified as "Unifiable", "Unifiable (occurrence of isolated state)", or "Conflict". Similarly, scripts can be classified as "Executable" or "Unexecutable". By comparing a script and an adaptor definition for the existing scripts, we can obtain a list of scripts annotated with 6 (3×2) classes.

Our script-management server displays the above list at the bottom of each wiki page. By displaying the list, the developer can easily find a script that can be included in his/her current application.

Figure 6 shows example of using the web-based interface.

5. Implementation of the Run-Time Engine

5.1. Architecture. SEAT consists of an adaptor mechanism, phrase matcher, automaton driver, and automaton unifier. In the next sections, we briefly overview each subsystem.

5.2. Adaptor Mechanism. The adaptor mechanism is used to connect the run-time engine to the other subsystems of the robot.

Adaptors are configured in XML format. For each adaptor configuration, an instance ID is defined. In the body of the state-transition model, the instance ID is used to describe the actions. By using this mechanism, even if the developer has changed the hardware configuration, the same state-transition model can be used by employing an adaptor definition that has the same instance IDs.

SEAT supports BSD socket communication, child process communication, UNIX standard input and output, and OpenRTM [15] as default interface types. Because the adapter mechanism is defined in an abstract form, the developer can easily add his/her own interface types.

```
func checkconflict(state1, state2):
  for c1 in state1.conditions:
    for c2 in state2.conditions:
      if (c1 == c2) && (c1.action == c2.action):
        detected = 1
```

Algorithm 3

```
func checkactions(adaptor, state):
  for c in state.conditions:
    if not exist c.action.instanceid in adaptor:
      detected = 1
```

Algorithm 4

5.3. Noise Robust Speech Recognition. A speech recognition function is also important in improving the accuracy of the robots' linguistic understanding. In the human life space, many noises occur around the robot. In such an environment, normal speech recognition algorithms are not accurate enough.

We have developed a speech recognition algorithm that works in a practical noise environment by using a signal processing technique combined with the speech recognition engine Julius [16]. Signal processing technique uses MUSIC spectrum method and fusion of video by Bayesian network, and it reduces environment noise by using ML beamforming (details are described in [17, 18]).

For HumanAID application (Figure 7), evaluation is done with two persons speaking simultaneously. Number of vocabulary was 492. Under this condition, the word error rate of speech recognition was over 19.9%, while the word error rate of normal speech recognition is 90.4%.

Speech recognition accuracy not only depends on environmental noises, but also depends on number of vocabularies. Because the recognizer needs to distinguish each word among given vocabularies, as the vocabulary increases, the recognition accuracy will go down. SEAT has a function to switch the speech recognition vocabularies depending on the situation. By using this function, the developer can increase the number of vocabularies of the total system while keeping the high speech recognition accuracy.

5.4. Phrase Matcher and Automaton Driver. The phrase matcher compares the input for each state-transition condition. To cope with the diversity of human language, we utilized a subset of regular expressions. If we write "[A]", phrase A is omissible. If we write "(A | B)", either phrase A or B can be matched.

When a match is found, the result is passed to the automaton driver. The automaton driver updates the current state and executes the commands based on the definition of the model. When a state transition occurs, switching of the speech recognition dictionary occurs at the same time.

6. Applications

6.1. Robots and Tasks. In this section, we present the applications we have developed using the development environment.

HRP-2. We have implemented the HumanAID task in the HRP-2 humanoid robot. The task is designed to assist people in everyday life. In the task, the robot greets the human, and the human gives commands to the robot, such as controlling the video or the TV, carrying drinks from the refrigerator to the table (Figure 7).

TAIZO. TAIZO is a health exercise demonstration robot [19]. It is a small robot character that greets people and demonstrates various exercises (Figure 8(a)).

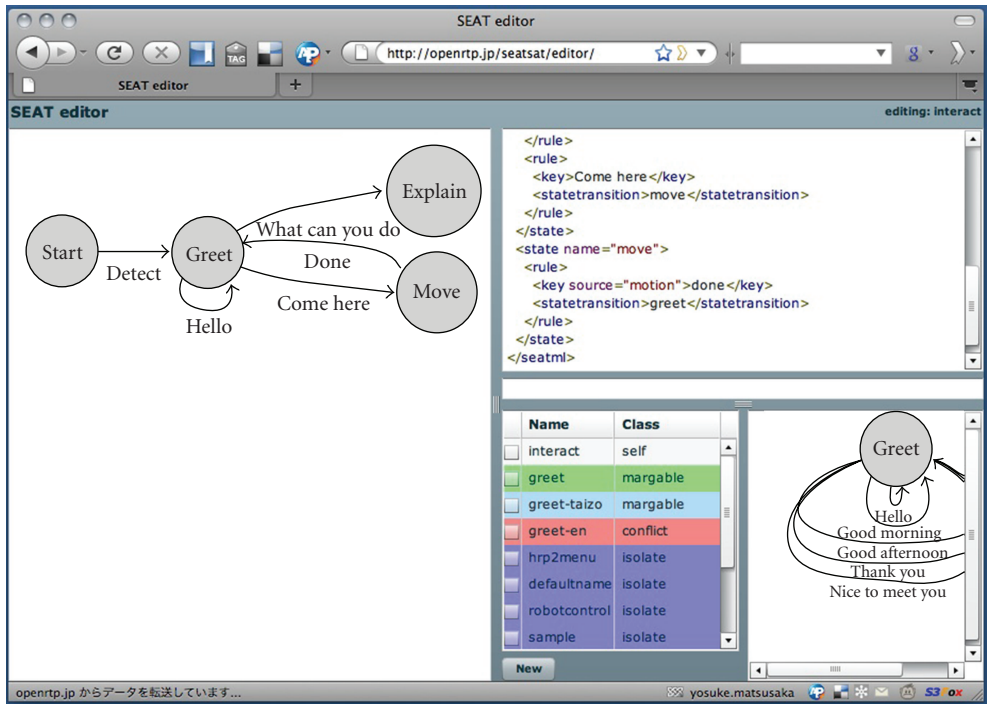
RH-1. RH-1 is a mobile robot that is designed to assist humans in the office environment (Figure 8(b)).

6.2. Development History. The development of the HumanAID task in HRP-2 has taken place from 2006 to 2007. Development of TAIZO and RH-1 has taken place from 2007 to the present. Table 1 shows the name of each script and its development period.

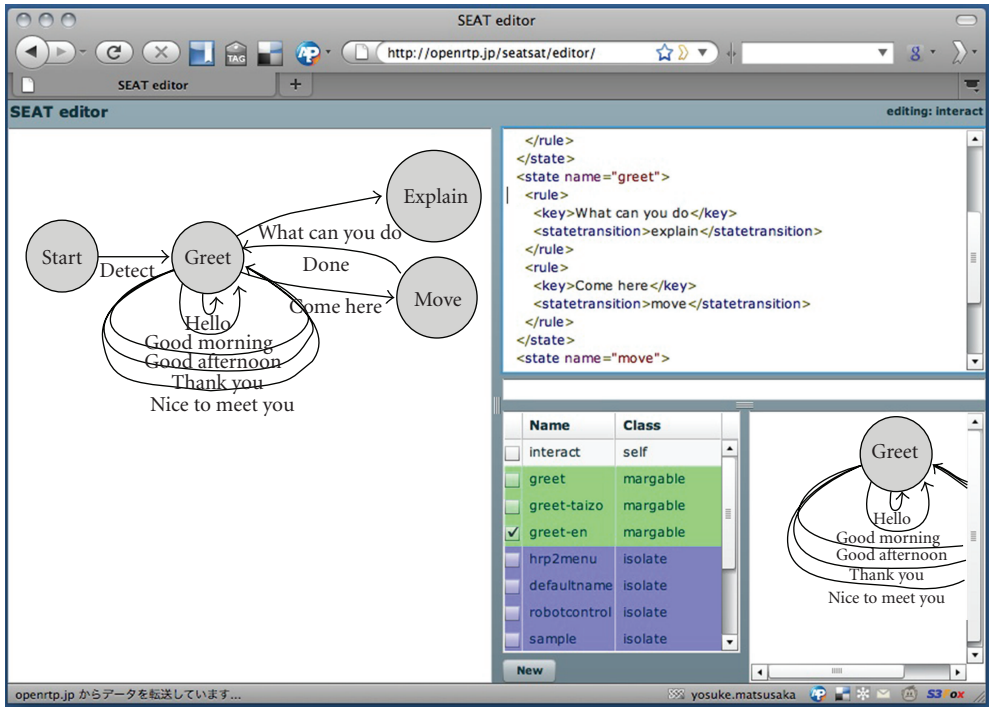
Here, we list the development history.

Unit-Based Development in HRP-2 (Period 1). HumanAID task functions have been developed separately. The state-transition model for demo conversation (e.g., saying "hello", "bye", introducing itself), the model for controlling the robot (e.g., walking, picking up objects), and the model for controlling the TV and VTR using an infrared controller were split into different scripts and developed simultaneously in this period.

Integration in HRP-2 (Period 2). After the development of each part of the function, a script was developed that



(a)



(b)

FIGURE 6: Example of using the web-based interface. (a) Overview of the development interface. Visualization of the state-transition model (left), XML-based editing panel (right top), real-time annotation of existing scripts (right bottom). Editing task script. When the developer types the keyword "Hello", the existing script from the script database is annotated as "conflict" and suggests reuse. At this step, the system only accepts 3 ("Hello", "What can you do?", "Come here") phrases. (b) When the developer checks the "greet" script, which already contains several vocabularies for greeting, it is unified to the task script. As a result, the developer only has to increment the application specific vocabulary to realize the whole script with many vocabularies. At this step, the system accepts 7 ("Hello", "Good morning", "Good afternoon", "Thank you", "Nice to meet you", "What can you do?", "Come here") phrases.

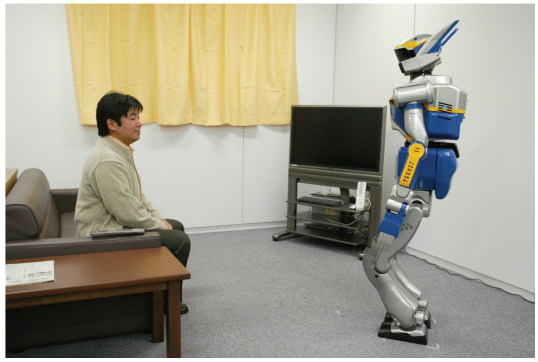
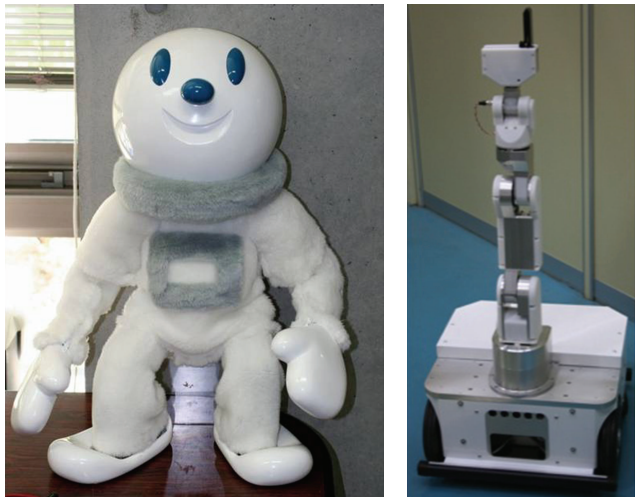


Figure 7: HRP-2 during a HumanAID task.



(a) (b)

Figure 8: TAIZO (a) and RH-1 (b) robots.

defined a menu. Within this script, a central state and states corresponding to each function were defined. For states corresponding to each function, nothing was contained in this script, but it was unified with other scripts to obtain the functions. Only transitions from the central state to each function state were defined in the script.

Each automaton has been developed and tested individually, but at the final stage of development, it was possible to unify the script by simply confirming the warning messages given by the script-management server.

Extension of the Scripts in TAIZO (Periods 3 and 4). Script development of TAIZO was conducted by extending the scripts developed for HRP-2. The scripts for greeting and basic demo tasks were selected for reuse, and the other scripts (TV-control, VTR-control) were not selected because TAIZO has no ability to control this equipment. To add more patterns to the greeting, the "greet-taizo" script was defined, which shares the same state as "greet" but adds more transitions. Functions specific to TAIZO were developed as the script "exercise". Finally, a menu script was developed to integrate all of the functions.

Table 1: Scripts used by each application and its development period.

Name of script	Used by			# of trans.	Period (developed for)
	HRP-2	TAIZO	RH-1		
greet	∗	∗	∗	3	1, 3 (HRP-2)
robot-ctl	∗	∗		13	1 (HRP-2)
tv-ctl	∗		∗	14	1 (HRP-2)
vtr-ctl	∗		∗	10	1 (HRP-2)
hrp-menu	∗			3	2 (HRP-2)
greet-taizo		∗		4	3 (TAIZO)
exercize		∗		17	3 (TAIZO)
taizo-menu		∗		17	4 (TAIZO)
wander			∗	15	5 (RH-1)
ask-who			∗	3	6 (RH-1)

Although the composition of the subsystems (e.g., speech recognizer, behavior generation) of TAIZO and HRP-2 was different, it was possible to share the scripts with no modification by simply switching the adaptor configurations.

Development of RH-1 (Periods 5 and 6). The script development of RH-1 was conducted simultaneously with the development of TAIZO. In RH-1, some control functions were imported from HRP-2. The script "wander" was defined as wandering around the office. This script not only uses speech input, but also uses visual information to find people. A multichannel input mechanism was used to integrate visual and speech inputs.

6.3. Results and Effects of Development. As a result of these developments, the number of acceptable command types has reached 43, 54, 45 for the respective applications.

Each application shares 93%, 30%, or 60% of its scripts with the other applications, respectively. As a total, 30% ($= 1 - (3+13+14+10+3+4+17+17+15+3)/(43+54+45)$) of the script development effort is reduced. Because the time used to develop the system was in proportion to the number of transitions defined in the script, the development time is estimated to have decreased by 30%.

7. Discussion

7.1. Comparison with Other Accumulative Development Methods. In the above discussion, we compared our method to the extension-by-connection method. Both the extension-by-connection and extension-by-unification methods belong to the same state-transition model group. For other modeling methods, and especially for artificial intelligence applications, a production system model is used in some applications.

A production system model is a modeling method that maintains the state of the system as a multidimensional feature vector, and controls the execution of actions by comparing the state to the pattern written in the script.

By using the same symbols as in Section 2, the production system model is formalized as follows:

$$A := \langle \mathbf{S}, R, s_0 \rangle, \tag{24}$$

where $\mathbf{S}$ is the state in vector (in the state-transition model, S was a set of states), R represents the conditions for each rule, and $\mathbf{s}_0$ is the initial state in vector.

Condition R is a comparison function that can be defined arbitrarily. In this paper, we define R as a matrix that consists of the conditions for each rule R_{ij}. The rows of R stand for each rule, and columns stand for the conditions corresponding to each dimension of features in the state vector. We define the conditions as "1": the value of the feature is positive, "−1": the value of the feature is negative, and "0": does not care.

Production system models are generally known to be extensible. Our model can easily accumulate rules, as follows:

$$\begin{aligned} R''_{i,f_m(j)} &= R_{i,j} \quad \left(R_{i,j} \neq 0, i, j \in S\right), \\ R''_{i+N,f'_m(j)} &= R'_{i,j} \quad \left(R'_{i,j} \neq 0, i, j \in S'\right), \\ R''_{i,j} &= 0 \quad (\text{otherwise}, i, j \in S''), \end{aligned} \tag{25}$$

where R'' represents the accumulated rules, R represents the existing rules, and R' is the rule for accumulation. f_m is a mapping function that converts each feature vector dimension into the other. S is a dimension of the feature vector, and N is the number of rules.

Although these are good points theoretically, in practical development there have only been a few examples of successful large-scale development. It is generally said that in a production system, the developers are required to be proficient in script development in order to ensure the extensibility of the system. We discuss this problem from the viewpoint of handling the states.

Let developer "A" has defined the state $\mathbf{S}_A$ and rule R_A, and developer "B" has accumulated rule R_B to extend the system. Rule R_B not only extends the rule, but also adds a new dimension to the state vector that is not used in rule R_A. Let the new state vector be $\mathbf{S}_B$ and the old state vector be $\mathbf{S}_A$. The final state vector in the extended system is $\mathbf{S}_{A+B} = \mathbf{S}_A \cup \mathbf{S}_B$.

In the extended system, the rules R_A and R_B are evaluated on an equal footing. However, when developer "A" developed rule R_A, only $\mathbf{S}_A$ was considered. On the other hand, when developer "B" developed, rule R_B, $\mathbf{S}_{A+B}$ was considered. $\mathbf{S}_{A+B}$ contains more information than $\mathbf{S}_A$, and this may cause an antinomy to rule R_A, which only considers $\mathbf{S}_A$. Figure 9 illustrates this antinomy.

To avoid this problem, the script developer needs to project the final system before beginning to develop the rule R_A, and maintain consistency during the development of rule R_B. However, this problem is as difficult as the frame problem [20] discussed in early artificial intelligence research.

In contrast, the state-transition model clearly defines the model in the form of state and transition conditions in the design phase.

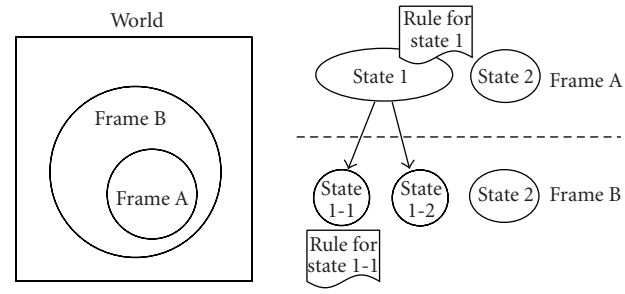


Figure 9: Antinomy between an existing rule and an accumulated rule. State 1 may be split into state 1-1 and state 1-2 when a new feature vector (frame B) is considered. There are no clear criteria to use to decide whether to select rule 1 or rule 1-1.

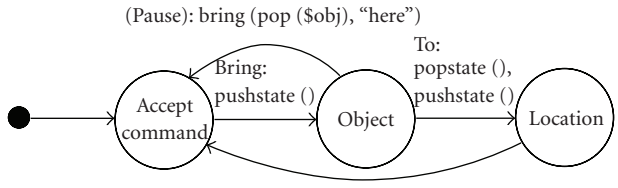
In state-transition model, we need to define a new state every time we add a new feature vector to the internal state so that the system can handle it. Here, we explain this using a concrete example. In rule R_A, developer "A" has considered feature A and defined $S_1 = \langle A = 0 \rangle, S_2 = \langle A = 1 \rangle$. Developer "$B$" wants to consider feature B in addition to feature A. Here, developer "B" has to define the new states $S_3 = \langle A = 0, B = 0 \rangle, \ldots, S_6 = \langle A = 1, B = 1 \rangle$. In the production system model, the state with smaller dimensions will be included in the state with larger dimensions. In the above case, all states will be included in $S = \langle A, B \rangle (A, B \in [0, 1])$, and states S_1 and S_2 will be overwritten.

In the state-transition model, the developer assesses the internal parameters of the system in the design step, but for the description, the developer needs to break down the combination of parameters into a set of states. When the developer wants to increase the internal parameters, he/she has to use a different state. Due to this restriction, the definition of a state is always clear, and is not overwritten by a script that is added later. Thus, the developer can proceed without being trapped by the issues discussed earlier in this section.

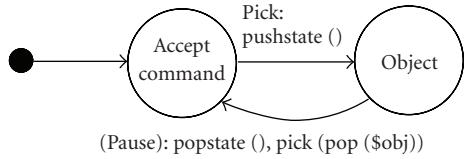
7.2. Reuse of Motion Content. In this paper, we have proposed a development environment to enhance the reuse of dialogue components. However, total development cost of the robot has to be calculated from both speech communication part and motion generation part.

Because the cost for reusing the motion depends on the algorithm, we first explain the algorithm we used. There are two methods for robot motion generation, one is the planning-based algorithm, and the other is the motion database. In our examples, for HumanAID and RH-1, we have used the planning-based algorithm. For TAIZO robot, we have used the motion database.

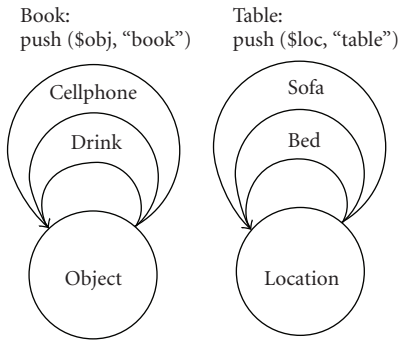
In the planning-based algorithm (HumanAID and RH-1) the motion generation algorithm will automatically generate the motion. We do not need to adjust the motion by hand, but only have to change the parameter such as structure of the arm, location of the target object. Because we share the motion generation algorithms between the robot, the cost of reusing the motion is very low in this case.



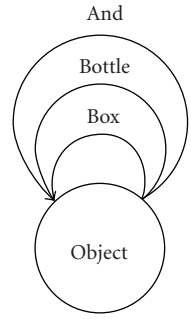
(a) Syntactics model for “bring” command



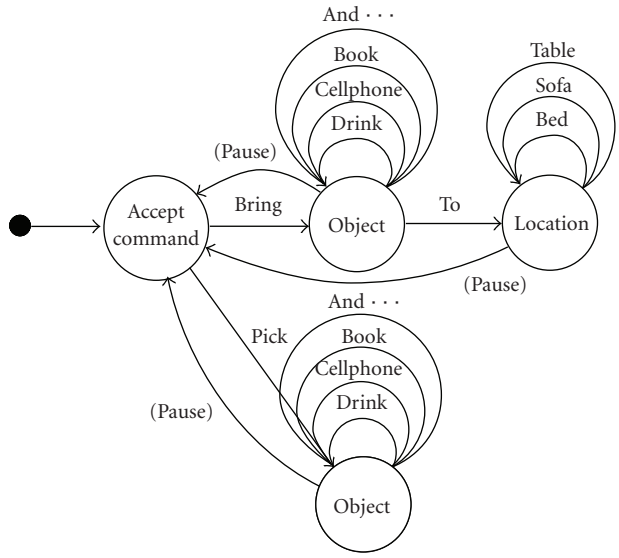
(b) Syntactics model for “pick” command



(c) Vocaburary for objects and locations (part of command is abbriviated)



(d) vocaburary for additional objects



(e) Unified automaton (commands are abbriviated in this figure)

Figure 10: An example of word-level input model.

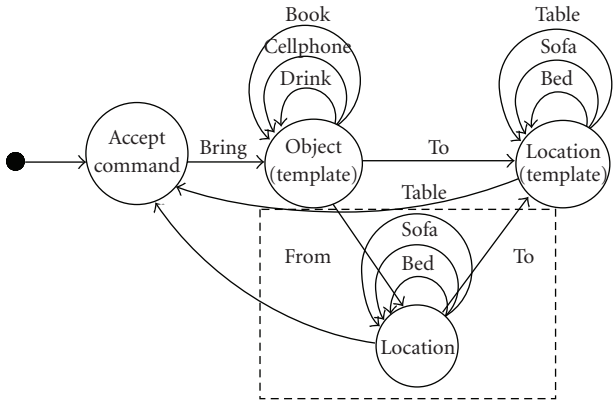


Figure 11: An example of three automaton unification. Dotted rectangle part is the extended part from the example shown in Figure 10.

When we use motion database (TAIZO), there is a problem in reusing the motion. Because the motion database is generated by hand, we have to create a new motion by hand for the new robot with different structure. For this problem, we are planning to implement motion retargeting technique. This algorithm was originally developed for creating a motion for computer animated creatures in movies [21]. The algorithm automatically generates a motion of animated creature from the motion of human actor by calculating the mapping between both structures. By using this motion retargeting algorithms, we believe the cost of reuse for motion database-based robot can be also reduced to very low level.

7.3. Word-Level Input and Pushdown Functions. State-transition model is possible to model the input in word level. An example is shown in Figure 10.

In the example, first, state-transition model in Figures 10(a), 10(b), and 10(c) is developed to interpret command. Second, state-transition model in Figure 10(d) is developed to extend the vocabulary. Finally, all the developed models are unified as Figure 10(e). This model uses pushdown automata [22] ("pushstate" and "popstate" command) to enable transition to the shared state "object" and return back. It also uses "push" and "pop" commands to hold the information about each object and location. Now the system can accept the input such as: "Bring Cellphone (pause)" converted as bring (cellphone, here), and "Bring Drink to Sofa (pause)" converted as bring (drink, sofa).

In our application, we have used the input in command (sentence) level. Modeling of input in word-level may be also useful, especially when the developer wants to share the syntactical structure of commands among the several scripts.

7.4. Comparison with the Template Methods. In VoiceXML, the industrial standard for scripting general voice operation system, there is an extension called RDC (Reusable Dialogue Component) [23]. The VoiceXML-RDC allows the developer to write the scripts of primitive interactions in an abstract form. The user can instantiate the primitive interactions by filling in the template parameters. By using the template, the user can generate dialogue scripts which fit to their application with less programming effort.

Our proposed method has equivalence to the template method. As we have seen in Figure 10, "object" slot of the syntax can be filled either with Figures 10(c) or 10(d).

The difference between our method and the template method is the possiblity to compose single model like in Figure 11 because our method uses "state" as a unit of unification, and allows unification of two or more components to one,. The template method is not able to realize this example, because it explicitly distinguish template part and instance parameters and only allows to unify those two.

7.5. Left Problems and Future Research. As shown in the examples of previous section, by using the proposed extension-by-unification method, the developer can develop functions simultaneously, and in the final step he/she can easily unify those functions to create an integrated system. In addition, scripts created in the past can easily be reused in the new application. In the conventional extension-by-connection method, the developer had to develop each function in turn, because it did not support the "merging" of scripts that had been developed simultaneously. Moreover, the developer needs to erase the unneeded functions manually when he/she wants to reuse the script in another robot. The proposed method does not require this process, because the script keeps information about the function even after the integration, and it can easily be separated for reuse. In our example, the script created for HRP-2 could be reused for TAIZO, but we had to remove the functions for TV and VTR control because TAIZO does not have this capability. In the conventional extension-by-connection method, we would have to do this manually. However, in the extension-by-unification method, this process is done simply by selecting the scripts to be unified.

As a result, we have confirmed that the extension-by-unification method significantly improves the efficiency of developing conversational function of the robots.

In terms of unification problems, it was possible to prevent the occurrence of isolated states by displaying a warning message, but there was a problem when the developer intentionally isolated the state. This happens when the developer has written a script in a redundant manner, or when he/she has tried to use pushdown automation. We are currently trying to solve this problem using 2 methods. One is a more intelligent isolation detection algorithm that can reduce misdetection (e.g., [24]), and the other will allow the developer to use an annotation tool that indicates that the state is intentionally isolated.

By applying a probabilistic weight to each transition of the state-transition model, the model became equivalent to the Markov model. There are some robots that have realized interactions with humans using such a model (e.g., [25]). In this paper, we have discussed the accumulation of the state-transition model based on deterministic input and output. However, probabilistic models are effective in modeling

real-world information, which includes noise. In further research, we are planning to incorporate these probabilistic models to the extensible development environment we have developed in this paper.

8. Summary

In this paper, we have proposed an extension-by-unification method to improve reusability and flexibility in the incremental development of state-transition models. The dialogue engine SEAT has been developed to realize the incremental development of state-transition models to give robots a dialogue ability that can cope with various kinds of speech inputs in various tasks. SEAT has a flexible adaptor mechanism that can connect to many types of robotic interfaces, and the developer can accumulate scripts by using the script server, which has a function to propose existing reusable scripts to the developer. We have confirmed that the application of this system to the development of three robots has significantly improved the efficiency of their development.

References

[1] N. Iwahashi, "Language acquisition through a human-robot interface," in *Proceedings of the International Conference on Spoken Language Processing (ICSLP '00)*, vol. 3, pp. 442–447, Beijing, China, 2000.

[2] D. Roy, "Grounded spoken language acquisition: experiments in word learning," *IEEE Transactions on Multimedia*, vol. 5, no. 2, pp. 197–209, 2003.

[3] A. L. Symeonidis, I. N. Athanasiadis, and P. A. Mitkas, "A retraining methodology for enhancing agent intelligence," *Knowledge-Based Systems*, vol. 20, no. 4, pp. 388–396, 2007.

[4] T. Winograd, *Understanding Natural Language*, Academic Press, New York, NY, USA, 1972.

[5] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, no. 1–3, pp. 139–159, 1991.

[6] L. Kaelbling, "A situated-automata approach to the design of embedded agents," *ACM SIGART Bulletin*, vol. 2, no. 4, pp. 85–88, 1991.

[7] B. Yartsev, G. Korneev, A. Shalyto, and V. Kotov, "Automata-based programming of the reactive multi-agent control systems," in *Proceedings of the IEEE International conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS '05)*, pp. 449–453, Waltham, Mass, USA, 2005.

[8] T. Kanda, H. Ishiguro, T. Ono, M. Imai, and R. Nakatsu, "Development and evaluation of an interactive humanoid robot Robovie," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1848–1855, Anchorage, Alaska, USA, May 2002.

[9] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita, "An affective guide robot in a shopping mall," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, pp. 173–180, La Jolla, Calif, USA, 2009.

[10] F. Huang, J. Yang, and A. Waibel, "Dialogue management for multimodal user registration," in *Proceedings of the International Conference on Spoken Language Processing*, vol. 3, pp. 37–40, Beijing, China, 2000.

[11] M. Denecke, "Informational characterization of dialogue states," in *Proceedings of the International Conference on Spoken Language Processing*, vol. 2, pp. 114–117, Beijing, China, 2000.

[12] SMC, "The State Machine Compiler," http://smc.sourceforge.net/.

[13] "Voice Extensible Markup Language (VoiceXML) Version 2.0," http://www.w3.org/TR/voicexml20/.

[14] "NEC RoboStudio," http://www.necst.co.jp/product/robot/.

[15] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W. Yoon, "RT-middleware: distributed component middleware for RT (robot technology)," in *Proceedings of theIEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3555–3560, Ibaraki, Japan, 2005.

[16] T. Kawahara, A. Lee, T. Kobayashi, et al., "Free software toolkit for Japanese large vocabulary continuous speech recognition," in *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP '00)*, vol. 4, pp. 476–479, Beijing, China, 2000.

[17] I. Hara, F. Asano, H. Asoh et al., "Robust speech interface based on audio and video information fusion for humanoid HRP-2," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '04)*, pp. 2404–2410, Sendai, Japan, October 2004.

[18] F. Asano, K. Yamamoto, I. Hara et al., "Detection and separation of speech event using audio and video information fusion and its application to robust speech interface," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 11, pp. 1727–1738, 2004.

[19] Y. Matsusaka, H. Fujii, T. Okano, and I. Hara, "Health exercise demonstration robot TAIZO and effects of using voice command in robot-human collaborative demonstration," in *Proceedings of the IEEE/RSJ International International Symposium on Robot and Human Interactive Communication*, pp. 472–477, Toyama, Japan, 2009.

[20] J. McCarthy and P. J. Hayes, "Some philosophical problems from the standpoint of artificial intelligence," *Machine Intelligence*, vol. 4, pp. 463–502, 1969.

[21] M. Gleicher, "Retargetting motion to new characters," in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pp. 33–42, ACM, Orlando, FLa, USA, July 1998.

[22] M. Sipser, *Introduction to the Theory of Computation*, PWS Publishing, Boston, Mass, USA, 1997.

[23] "Reusable Dialog Components (RDC) Tag Library," http://jakarta.apache.org/taglibs/doc/rdc-doc/.

[24] A. Bouajjani, J. Esparza, and O. Maler, "Reachability analysis of pushdown automata: application to model-checking," in *Proceedings of the 8th International Conference on Concurrency Theory*, pp. 135–150, Warsaw, Poland, 1997.

[25] H. Asoh, Y. Motomura, I. Hara, S. Akaho, S. Hayamizu, and T. Matsui, "Combining probabilistic map and dialog for robust life-long office navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 807–818, Maui, Hawaii, USA, 1996.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 360925, 12 pages
doi:10.1155/2010/360925

Research Article

Cohesive Motion Control Algorithm for Formation of Multiple Autonomous Agents

Debabrata Atta,[1] Bidyadhar Subudhi,[1] and Madan M. Gupta[2]

[1] *Centre of Industrial Electronics & Robotics, Department of Electrical Engineering, National Institute of Technology Rourkela, Rourkela 768018, India*
[2] *Intelligent Systems Research Laboratory, College of Engineering, University of Saskatchewan, Saskatoon, SK, Canada S7N 5A9*

Correspondence should be addressed to Bidyadhar Subudhi, bidyadharnitrkl@gmail.com

Received 2 October 2009; Accepted 31 May 2010

Academic Editor: Noriyasu Homma

This paper presents a motion control strategy for a rigid and constraint consistent formation that can be modeled by a directed graph whose each vertex represents individual agent kinematics and each of directed edges represents distance constraints maintained by an agent, called follower, to its neighbouring agent. A rigid and constraint consistent graph is called persistent graph. A persistent graph is minimally persistent if it is persistent, and no edge can be removed without losing its persistence. An acyclic (free of cycles in its sensing pattern) minimally persistent graph of Leader-Follower structure has been considered here which can be constructed from an initial Leader-Follower seed (initial graph with two vertices, one is Leader and another one is First Follower and one edge in between them is directed towards Leader) by Henneberg sequence (a procedure of growing a graph) containing only vertex additions. A set of nonlinear optimization-based decentralized control laws for mobile autonomous point agents in two dimensional plane have been proposed. An infinitesimal deviation in formation shape created continuous motion of Leader is compensated by corresponding continuous motion of other agents fulfilling the shortest path criteria.

1. Introduction

There are several examples of coordinated team work of mobile agents in nature like food foraging by a group of ants, flocking birds, fish schooling for protection from enemies, and so forth. These examples give us a lesson that a particular task accomplished by a single mobile autonomous agent (like fish and birds) may be done more efficiently by a group of agents if they perform in a collaborative manner. During last thirty years or more, this fact has motivated many researchers in the field of control and automation significantly to contribute in several similar applications in industry. In some particular applications during their motion as whole, autonomous agents (e.g., robots, vehicles, etc.) need to maintain a particular geometrical shape for cohesive motion, called formation which satisfies some constraints like desired distance between two agents, desired angle between two lines joining two agents each. Examples of these types of formations are found in collective attack by a group of combat aircraft, search/rescue/surveillance/cooperative transportation by a troop of robots, underwater exploration/underwater inspection (like pipeline inspection) by a group of Autonomous Underwater Vehicles, attitude alignment of clusters of satellites, air traffic management system, automated highway system, and so forth.

In the area of formation control of autonomous agents like robots, the motion control strategies may be either a centralized one or it could be a decentralized control. In centralized mode of control [1, 2], the command for all agents of the group are assigned by the central command control board or a designated group leader for monitoring and control of all agents to guide them be placed at desired position. The centralized formation control could be a good scheme for a small group of robots, when it is implemented with a single computer and a single sensor to monitor and control the entire group. Control of large number of robots in a formation requires greater computational capacity and large amount communication. Due to these problems, decentralized control is recommended when formation involves a large number of agents. In the decentralized mode

of control [3–5], one agent of the group can be a leader and others are followers (or each agent of the group can be a leader and follower except a designated group leader and the two outmost agents) and as a follower, each agent generates its own commands autonomously (i.e., control law for each agent is provided separately such that each agent works autonomously) based on the relative measurement only from its neighbours without the need of an external supervisor, and the whole purpose of the formation motion is achieved.

Although two types of basic control strategies are discussed in the last paragraph, motion control scheme for formation may be modified depending upon some factors like agent dynamics, interagent information exchange structure, control goals in different applications, and so forth.

Formations are modeled using *formation graph* whose each vertex represents individual agent kinematics and each edge represents interagent constraint (e.g., desired distance) that must be maintained during motion of formation. Specifically, graph is used to represent coordinated behaviour among agents. Depending upon the pattern of information exchange in that coordination, two types of graphs are possible, that is, *directed* and *undirected*.

In undirected formation graph both of any pair of agents, constrained by an edge have equal responsibility to satisfy the constraint. For example, distance between any pair of agents is sensed by both the agents; that is, sensing is distributed. Therefore, structure of undirected graph suffers from various disadvantages. One of those disadvantages is information-based instability [6], which happens due to possibility of difference in distances measured by noisy sensors of any pair of agents. Communication requirement among agents is more. Therefore, external observer-based centralized control strategy is best suited for undirected formation graph. Control law is mainly focused on rigidity property of formation.

Formation graph is *rigid* [7] if distance between any pair of agents remains constant during any continuous motion of formation. A graph is said to be *minimally rigid* if it is rigid and if there is no rigid graph having the same vertices but fewer edges.

In directed formation graph, only one (called *follower*) of any pair of agents, constrained by an edge has responsibility to satisfy the constraint. Therefore, decentralized control strategy is best suited for directed formation graph.

A graph is *constraint consistent* [8] if every agent is able to satisfy all the constraints on it provided all others are trying to do so. A formation that both rigid and constraint consistent is termed as *persistent* graph.*Persistence* is a generalization to directed graphs of the undirected notion of rigidity. A persistent graph is *minimally persistent* if it is persistent and if no edge can be removed without losing persistence.

However, focus in this paper is in development of control strategy for directed graph-based formation. There are several papers in which basics of directed formation graph-related issues are discussed [7–10]. Definitions and theorems [7–10] with regard to undirected and directed graphs included in Section 2.

Digraph is called acyclic when no cycle is present in its sensing pattern [6, 8]. Control scheme for cyclic formation is more complicated than acyclic formation. Minimally persistent formation of autonomous agents may be formed in two ways. First one is *leader-follower* graph architecture constructed from an initial leader-follower seed by Henneberg Sequence with standard vertex additions or edge splitting [11]. Leader-follower type minimally persistent graph is always acyclic. Another type of construction by sequence of specific operation elaborated in [9] such that every intermediate construction is also persistent. Minimally persistent graph constructed by this method may have cycles. In this paper, control strategies for only leader-follower type formation constructed from sequence of vertex addition is described.

Although a number of research works have been directed in the area of cyclic formation graph, but still there remains scope of further work. In [12], Anderson et al. have proposed a distance preservation-based control law when cycles contain in the formation graph.

In most recent works [13, 14] or some previous works [6], distributed control is provided with exploiting gradient control law for each autonomous agent in a formation separately. In [5], formation control strategy of leader-follower and three coleader structures is set up based on discrete-time motion equations considering decentralized approach. In [15], Anderson et al. analyzed control of leader-follower structure in continuous domain assuming linearized system for small motion and stability aspects are also discussed. However, we proceed for the control of leader-follower formation in a different way; that is, our approach is based on optimization of some distances.

In the present paper, we consider the motion control scheme of the *leader-follower* structure type persistent formation in continuous domain that is based on optimization of a set of nonlinear objective functions under a set of equality constraints where each objective function (corresponding to each agent) corresponds to a specific constraints separately, as control scheme considers a decentralized approach. With advent of high speed computational platforms the solution associated with optimization procedure in the control generation is possible.

This paper is organized as follows. In Section 2, application of graph theory in formation control has been discussed. Problem formulation for formation control of multiple autonomous agents has been included in Section 3 followed by the development of control law, simulation, and conclusion in Sections 4, 5, and 6, respectively.

2. Application of Graph Theory in Formation Control

As briefly described about the graph theory in Section 1, it is observed that graph is the best way for proper understanding of information flow among agents in a formation. This makes groundwork for control engineers to design an efficient control scheme for the motion control of formation. In this paper, formation of leader-follower structure has been

considered and it may be extended to any number of agents. Therefore, it is worthwhile to understand how a leader-follower type formation graph is built up from an arbitrarily chosen number of agents. In this section, at the outset the pertinent definitions and theorems given in [7–10] about the rigidity and persistence are reviewed and subsequently the procedure of creating a leader-follower formation for a case of four numbers of autonomous agents is described.

2.1. Definitions Associated with Rigid Graph

2.1.1. Infinitesimally Rigid Graph. A representation p of an undirected graph $G(V,E)$ with vertex set V and edge set E is a function $p : V \to \Re^d$, where $d(2,3,\ldots)$ is the dimension of Euclidean space. Representation p is rigid if there exists $\varepsilon > 0$ such that for all realizations due to continuous deformations p' of distance set induced by p and satisfying, $d(p,p') < \varepsilon$ (where, $d(p,p') = \max \|p(i) - p'(i)\|$, where, $i \in V$), there holds $\|p'(i) - p'(j)\| = \|p(i) - p(j)\|$ for all $i,j \in V$. We simply say this phenomenon as congruence relationship between p and p'.

2.1.2. Generically Rigid Graph. A graph is said to be generically rigid if almost all realizations due to continuous deformations are rigid. This definition of rigidity is to exclude some undesirable situations like certain collections of vertices are collinear during continuous deformations.

2.1.3. Minimally Rigid Graph. A rigid graph is minimally rigid when no single edge can be removed without losing its rigidity.

2.1.4. Laman's Criteria (see [10]). If an undirected graph $G = (V,E)$ in $\Re^2$ with at least two vertices is rigid, then there exists a subset E' of edges such that $|E'| = 2|V| - 3$ and any subgraph $G'' = (V'',E'')$ of G' with at least two vertices satisfies $|E''| \le 2|V(E'')| - 3$, where $|V(E'')|$is number of vertices that are end-vertices of the edges in E''.

Lemma 1 (see [7]). *Let $G = (V,E)$ be a minimally rigid graph in $\Re^2$ and $G' = (V',E')$ a subgraph of G such that $|E'| = 2|V'| - 3$. Then, G' is minimally rigid.*

In directed graph, each agent is only aware of its own distance constraints and can move freely as long as these distance constraints are satisfied. Persistence is a directed notion and rigidity is an undirected notion. The properties of directed graphs are described below.

2.2. Definitions Associated with Persistent Graph.

Suppose, for a directed graph G, desired distances $d_{ij} > 0$ for all $(i,j) \in E$, edge set where $i,j \in V$, vertex set, and a realization p, then the following definitions are described.

2.2.1. Active Edge. The edge $(i,j) \in E$ is *active* if $\|p(i) - p(j)\| = d_{ij}$, that is, if the corresponding distance constraint is satisfied.

2.2.2. Fitting Position of a Vertex. The position of a vertex $i \in V$ is *fitting* for any desired distance set $\{d\}$ of G, if it is not possible to increase the set of active edges leaving i by changing the position of i while maintaining the positions of other vertices unchanged. Specifically, the position of vertex i, for a given realization p, is fitting if there is no $p' \in \Re^2$ for which the condition elaborated below is strictly satisfied:

$$\left\{(i,j) \in E : ||p(i) - p(j)|| = d_{ij}\right\} \subset \left\{(i,j) \in E : ||p' - p(j)|| = d_{ij}\right\}. \quad (1)$$

2.2.3. Fitting Realization of a Graph. A realization of a graph is a fitting realization for a certain distance set $\{d\}$ if all the vertices are at fitting positions for $\{d\}$.

2.2.4. Constraint Consistent Graph. A realization p of digraph G is constraint consistent if there exists $\varepsilon > 0$ such that any realization p' fitting for the distance set $\{d\}$ induced by p and satisfying $d(p,p') < \varepsilon$ is a realization of $\{d\}$. A graph is generically constraint consistent if almost all realizations are constraint consistent.

2.2.5. Persistent Graph. Realization p of the digraph G having desired distances $d_{ij} > 0$ for all $\{i,j\}$ is persistent if there exists $\varepsilon > 0$ such that every realization due to continuous deformation, p' fitting for the distance set induced by p and satisfying $d(p,p') < \varepsilon$ is congruent to p.

2.2.6. Generically Persistent Graph. A graph is generically persistent if almost all possible realizations are persistent (same as elaborated in case of generically rigid).

Theorem 1 (see [8]). *A realization is persistent if and only if it is rigid and constraint consistent.*

A graph is generically persistent if and only if it is generically rigid and constraint consistent.

2.2.7. Minimally Persistent Graph. A persistent graph is minimally persistent if it is persistent and if no edge can be removed without losing persistence.

Theorem 2 (see [8]). *A rigid graph is minimally persistent if and only if either of the two conditions elaborated below is satisfied.*

(i) *Out of all vertices, each of three vertices has one outgoing edge and each of rest vertices has two outgoing edges*

(ii) *Out of all vertices, one vertex has no outgoing edge; another one vertex has one outgoing edge and each of rest vertices has two outgoing edges.*

Theorem 3 (see [8]). *An acyclic digraph is persistent if all the conditions elaborated below are satisfied.*

Out of all vertices,

(i) *one vertex has one outgoing edge. This vertex represents Leader,*

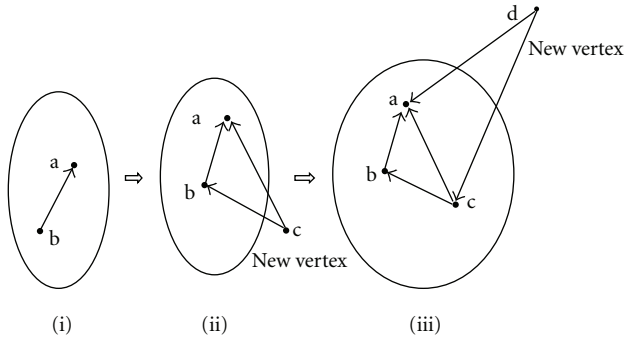


Figure 1: Henneberg construction for making leader-follower type formation from four agents "a", "b", "c", and "d" (i) "a", "b" are initial vertices with one edge directed towards "a" (leader) from "b" (first follower) (ii) "c" is new vertex added to (i) (iii) "d" is new vertex added to (b).

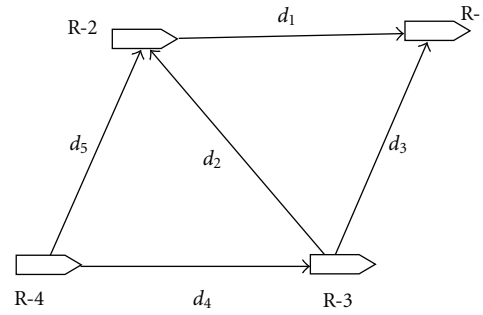


Figure 2: Quadrilateral formation of leader-follower structure. In this figure, the notations are as follows. R-1 denotes the leader and R-2 denotes the first follower, R-3 is for the ordinary follower-1 and R-4 is for ordinary follower-2.

(ii) *another vertex has one outgoing edge which must be incident to the Leader. This vertex represents first follower,*

(iii) *each of rest vertices must have two or more number of outgoing edges.*

2.3. Construction of Acyclic Minimally Persistent Graph. Acyclic minimally persistent graph is always constructed starting from a combination of two vertices, one is called leader and the other one is first follower, and an edge directed towards leader using Henneberg Sequence with only vertex additions.

2.4. Henneberg Construction (Directed Case) Containing Vertex Addition. It describes the sequence of graphs $G_2, G_3, \ldots G_{|V|}$, such that each graph G_{i+1} $(i \geq 2)$ can be obtained by a vertex addition starting from G_i, where i is number of vertices and $|V|$ is cardinality of vertex set of desired graph. Therefore, the procedure of drawing the graph using Henneberg sequence is described as

(i) start with a directed edge between two vertices.The vertex towards which edge is directed is called leader and remaining vertex is called first follower. The combination of these two a vertex with a directed edge is called initial leader-follower seed,

(ii) at each step of growing graph, add a new vertex,

(iii) join the new vertex to two old vertices (corresponding to leader and first follower) via two new edges, directed towards old vertices.

Figure 1 shows the construction procedure of formation of four agents.

3. Problem Formulation

For simplicity, we restrict our analysis only in quadrilateral formation of leader-follower structure taking into account four mobile autonomous point agents, in plane. This formation is shown in Figure 2. One among them is leader which has no outgoing edge; that is, it is free to move along a specified path and does not have any responsibility to maintain any distance constraint from other agents, the second one as the first follower which has one outgoing edge; that is, it requires to maintain only one desired distance constraint from the leader, the third one as ordinary follower-1 which has two outgoing edges; that is, it requires to maintain two distance constraints, one of which is directed towards leader and other one towards first follower, the fourth one as ordinary follower-2 which has two outgoing edges; that is, it requires to maintain two distance constraints, one of which is directed towards first follower and other one towards ordinary follower-1.

The different distance constraints among agents assumed are as follows: d_1 is the desired distance maintained by first follower from leader, d_2 is the desired distance maintained by ordinary follower-1 from first follower, d_3 is the desired distance maintained by ordinary follower-1 from leader, d_4 is the desired distance maintained by ordinary follower-2 from ordinary follower-1, and d_5 is the desired distance maintained by ordinary follower-2 from first follower.

We assume the desired distances among the agents satisfy noncollinear condition such that at least three point agents do not form a straight line.

Assumptions 1. (i) For simplicity, for each agent "i", the kinematic model [5] of unicycle nonholonomic point agent is considered as follows:

$$(\dot{x}_i, \dot{y}_i) = (v_i \cos\theta_i, v_i \sin\theta_i), \quad \dot{\theta}_i = \omega_i, \tag{2}$$

where $p_i(t) = (x_i(t), y_i(t))$ with $i = 1, 2, 3, 4$ denotes, the position of ith agent, $\theta_i(t)$, and $v_i(t)$ denote the orientation and translational velocity of the ith agent at each instant of time t, respectively. $(x_i(t), y_i(t))$ is with respect to an earth-fixed coordinate coordinate system.

(ii) Each agent can measure its position with respect to earth-fixed coordinate system by proper sensor arrangement.

(iii) First follower has position information of leader only with respect to its own point body system by using an active sensor (e.g., sonar).

(iv) Ordinary follower-1 has position information of leader and first follower only with respect to its own point body system by using an active sensor (e.g., sonar).

(v) Ordinary follower-2 has position information of first follower and ordinary follower-1 only with respect to its own point body system by using an active sensor (e.g., sonar).

(vi) Each agent (except leader) can achieve the positions of neighbours w.r.t earth-fixed coordinate system from the relative informations of its neighbours and its own position w.r.t. earth-fixed coordinate system.

Now, the objective is to provide a control scheme for this triangular formation (starting from a non-collinear arrangement) such that during the motion of three robots for any mission, desired interagent distances are preserved.

4. Development of Control Law

Here, we intend to develop set of decentralized control laws for overall formation. Therefore, for each agent, we provide separate control law for continuous movement in autonomous manner based on local knowledge only of direction of its neighbours. Control Law for each agent is derived by optimizing corresponding objective function with given constraints involving desired distance constraints of formation. The unknown variables to be solved are x and y coordinates (which are continuous function of time) of position (with respect to earth-fixed coordinate system) of corresponding agent for which objective function is derived.

Assumptions 2. (i) Interagent distances are sufficiently large so that initial collision among robots can be avoided.

(ii) Initially positions of robots are not collinear (already stated).

(iii) During motion of formation, failure of sensors do not occur.

(iv) There is no time delay in sensing.

(v) Control input in the form of translational velocity and angular velocity (discussed later in this section) calculated from final and initial position of any agent should be necessarily generated by controller of each agent.

When all the agents of given formation completes movement to a new set of position coordinates from an old set of position coordinates during certain period of time such that desired distances among agents are preserved for both set of positions and not any other distance preserving position set is available in between these two position sets during motion, then the movement of formation from the old set to new set of positions is called one complete displacement of formation.

Before proceeding to develop the control laws for all agents, it is assumed that at any time t each agent is maintaining its own distance constraints. Then how these agents move to their new positions for a complete displacement is discussed below.

4.1. Control Law for Leader. Leader does not need to maintain any distance constraint from any other agent in formation. A specified control action is provided for its dynamics such that it moves along a specified path (trajectory); that is, each position (at each instant of time t) coordinate is known (preprogrammed (known)/unknown) to computational system of the leader. Suppose at time t initial position coordinate for leader is assumed as $((x_{1\,\mathrm{In}}(t), y_{1\,\mathrm{In}}(t))$. Let the leader move to a new position $(x_{1f}(t), y_{1f}(t))$, that is, the final position (rest point), in very small period of time $\mathrm{d}t$ *such that continuity preserves between* $(x_{1\,\mathrm{In}}(t), y_{1\,\mathrm{In}}(t))$ & $(x_{1f}(t), y_{1f}(t))$; that is, *the distance between these two positions is sufficiently small.* This motion of the leader and corresponding movement of the first follower is shown in Figure 3. According to Figure 2, $\mathrm{d}\vec{s_1} = \mathrm{d}x_1\hat{i} + \mathrm{d}y_1\hat{j}$, where $\vec{s_1}(t)$, a vector field along the trajectory curve of the leader, $\hat{i}$ and $\hat{j}$ are unit vectors along x and y direction of the global coordinate system, and $\mathrm{d}x_1 = x_{1f} - x_{1\,\mathrm{In}}$ and $\mathrm{d}y_1 = y_{1f} - y_{1\,\mathrm{In}}$. It should be noted that $(x_{1\,\mathrm{In}}(t), y_{1\,\mathrm{In}}(t))$ and $(x_{1f}(t), y_{1f}(t))$ are always on $\vec{s_1}(t)$. Therefore, control input to reach its final position is

$$\vec{v_1}(t) = \frac{\mathrm{d}\vec{s_1}(t)}{\mathrm{d}t} = \frac{\mathrm{d}x_1}{\mathrm{d}t}\hat{i} + \frac{\mathrm{d}y_1}{\mathrm{d}t}\hat{j} = v_{1x}(t)\hat{i} + v_{1y}(t)\hat{j}, \tag{3}$$

where, $\mathrm{d}x_1/\mathrm{d}t = v_{1x}(t)$ and $\mathrm{d}y_1/\mathrm{d}t = v_{1y}(t)$.

The translational velocity control input during $\mathrm{d}t = \|\vec{v_1}\| = \sqrt{(v_{1x}(t))^2 + (v_{1y}(t))^2}$ meter/second.

The angular velocity control input (rad./sec) during same period of time is

$$\begin{aligned}
\omega_1(t) &= \tan^{-1}\left|\frac{\mathrm{d}y_1}{\mathrm{d}x_1}\right| \quad \text{when } \mathrm{d}x_1 \text{ is} + \text{ve},\ \mathrm{d}y_1 \text{ is } + \text{ve} \\
&= \left(\pi - \tan^{-1}\left|\frac{\mathrm{d}y_1}{\mathrm{d}x_1}\right|\right) \quad \text{when } \mathrm{d}x_1 \text{ is } - \text{ve},\ \mathrm{d}y_1 \text{ is } + \text{ve} \\
&= -\left(\pi - \tan^{-1}\left|\frac{\mathrm{d}y_1}{\mathrm{d}x_1}\right|\right) \quad \text{when } \mathrm{d}x_1 \text{ is } - \text{ve},\ \mathrm{d}y_1 \text{ is } - \text{ve} \\
&= -\tan^{-1}\left|\frac{\mathrm{d}y_1}{\mathrm{d}x_1}\right| \quad \text{when } \mathrm{d}x_1 \text{ is } + \text{ve},\ \mathrm{d}y_1 \text{ is } - \text{ve} \\
&= -\frac{\pi}{2} \text{ or } \frac{\pi}{2} \quad \text{when } \mathrm{d}x_1 = 0 \text{ and } \mathrm{d}y_1 \text{ is } + \text{ve or } - \text{ve}.
\end{aligned} \tag{4}$$

4.2. Control Law for First Follower. It may be noted that first follower has one outgoing edge; that is, it has to maintain only one distance constraint (desired distance d_1) and that is to leader. Initial and final position coordinates for the leader are $(x_{1\,\mathrm{In}}(t), y_{1\,\mathrm{In}}(t))$, $(x_{1f}(t), y_{1f}(t))$, respectively. Then, first follower senses the disturbance in position of leader; that is, it senses error in desired distance constraint (d_1) by sensing the final position of the leader staying at its initial position $(x_{2\,\mathrm{In}}(t), y_{2\,\mathrm{In}}(t))$. It tries to satisfy this distance constraint to leader. Therefore, suppose it moves to a rest point (final position) $(x_{2f}(t), y_{2f}(t))$ at the next instant of time dt after the instant during which the leader moves to its final position. During this movement of first follower, leader is assumed to be stationary at the position $(x_{1f}(t), y_{1f}(t))$. A condition is given to first follower such that only due to disturbance in position of leader, first follower changes its position.To maintain the cohesive motion with the leader,

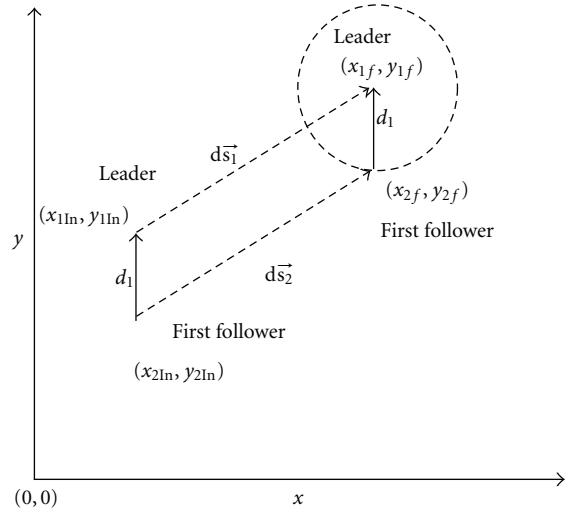


Figure 3: Motion of leader and first follower for a very small duration of time.

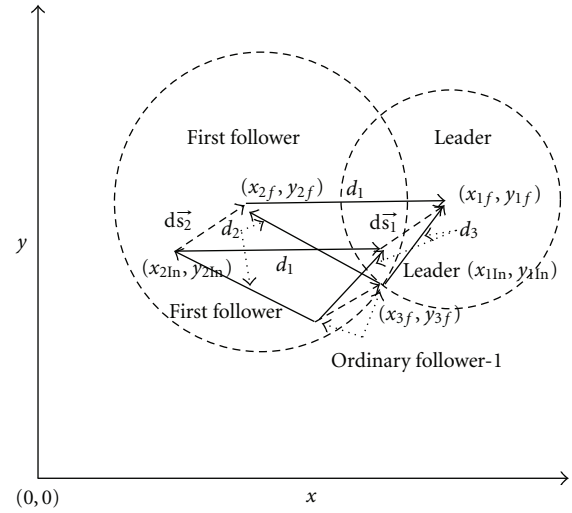


Figure 4: Motion of leader, first follower, and ordinary follower-1 for a very small duration of time.

distance of the first follower from the leader must be d_1 at final position of both the agents. It can be formulated as follows:

$$\left(x_{1f}(t) - x_{2f}(t)\right)^2 + \left(y_{1f}(t) - y_{2f}(t)\right)^2 - d_1^2 = 0, \quad (5)$$

where the values of x_{1f}, y_{1f}, and d_1 are known to computational system of first follower. But it is clear from (5) that locus of the position of first follower is a circle. Hence, its rest position may be at anywhere on this circle. First follower may take its rest position for which it crosses over the leader and may collide with leader. Undesirable consequence of this phenomenon is that ordinary follower may collide with leader and first follower both for maintaining the distant constraints from both of them. Hence, to provide a control avoiding this unsafe situation, a restriction to the motion of first follower must be imposed, such that it reaches to a.

In Figure 3, $\|\vec{ds_2}\|$ is defined as the distance between the final and initial position of first follower. Here, $\vec{s_2}(t)$ a vector field along the trajectory curve of first follower. It should be noted that $(x_{2\,\mathrm{In}}(t), y_{2\,\mathrm{In}}(t))$, and $(x_{2f}(t), y_{2f}(t))$ are always on $\vec{s_2}(t)$. Then, we have

$$\|\vec{ds_2}\|^2 = \left(x_{2f}(t), x_{2\,\mathrm{In}}(t)\right)^2 + \left(y_{2f}(t), y_{2\,\mathrm{In}}(t)\right)^2. \quad (6)$$

Therefore, $\|\vec{ds_2}\|$ must be minimum such that first follower moves along the shortest path to its final position. Hence, the first follower follows the leader maintaining safe motion. Now, we intend to propose a control law for motion of first follower satisfying the aforesaid conditions. Actually, the whole problem may be treated as an optimization problem, where minimization of objective function (7) under equality constraint (6) should be performed. And a control law based on this optimization is presented as

$$\vec{v_2}(t) = \frac{\vec{ds_2}}{dt} = \frac{dx_2}{dt}\hat{i} + \frac{dy_2}{dt}\hat{j} = v_{2x}(t)\hat{i} + v_{2y}(t)\hat{j}, \quad (7)$$

where $dx_2/dt = v_{2x}(t)$ and $dy_2/dt = v_{2y}(t)$ and $\hat{i}$ and $\hat{j}$ are unit vectors along x and y direction of the global coordinate system.

The translational velocity control input $= \|\vec{v_2}\| = \sqrt{(v_{2x}(t))^2 + (v_{2y}(t))^2}$ meter/second.

The angular velocity (rad/sec) control input is

$$\begin{aligned}
\omega_2(t) &= \tan^{-1}\left|\frac{dy_2}{dx_2}\right| \quad \text{when } dx_2 \text{ is } +\text{ve},\ dy_2 \text{ is } +\text{ve} \\
&= \left(\pi - \tan^{-1}\left|\frac{dy_2}{dx_2}\right|\right) \quad \text{when } dx_2 \text{ is } -\text{ve},\ dy_2 \text{ is } +\text{ve} \\
&= -\left(\pi - \tan^{-1}\left|\frac{dy_2}{dx_2}\right|\right) \quad \text{when } dx_2 \text{ is } -\text{ve},\ dy_2 \text{ is } -\text{ve} \\
&= -\tan^{-1}\left|\frac{dy_2}{dx_2}\right| \quad \text{when } dx_2 \text{ is } +\text{ve},\ dy_2 \text{ is } -\text{ve} \\
&= -\frac{\pi}{2} \text{ or } \frac{\pi}{2} \quad \text{when } dx_2 = 0 \text{ and } dy_2 \text{ is } +\text{ve or } -\text{ve}.
\end{aligned} \quad (8)$$

4.3. Control Law for Ordinary Follower-1. Ordinary follower-1 tries to satisfy two distance constraints; that is, it has two outgoing edges, the first one (d_2) is directed towards first follower and the second one (d_3) is directed towards leader. Let the leader and first follower be placed at their corresponding final positions. The ordinary follower senses the disturbances in position of leader and first follower; that is, it senses error in desired distance constraints d_2 and d_3 by sensing the final position of the first follower and leader, respectively. It tries to satisfy these distance constraints to first follower and leader (as shown in Figure 4). Therefore, suppose the ordinary follower moves to its final position

(rest point) at the next time instant of the time (dt) after the instant during which first follower moves to its final position. During this movement of ordinary follower, leader and first follower are assumed to be stationary at the position $(x_{1f}(t), y_{1f}(t))$ and $(x_{2f}(t), y_{2f}(t))$, respectively. At the final position, ordinary follower satisfies its distance constraints. Here, a condition is given to the ordinary follower such that only when disturbances in positions of both leader as well as first follower (not merely leader) occur; ordinary follower changes its position to final one. This final position is assumed $(x_{3f}(t), y_{3f}(t))$. The assumed initial position of the ordinary follower is $(x_{3\,\mathrm{In}}(t), y_{3\,\mathrm{In}}(t))$. Now, according to the desired distance constraints for it, two conditions are to be satisfied as given in

$$\begin{aligned}\left(x_{1f}(t) - x_{3f}(t)\right)^2 + \left(y_{1f}(t) - y_{3f}(t)\right)^2 - d_3^2 = 0,\\ \left(x_{2f}(t) - x_{3f}(t)\right)^2 + \left(y_{2f}(t) - y_{3f}(t)\right)^2 - d_2^2 = 0,\end{aligned} \tag{9}$$

where $x_{1f}, y_{1f}, x_{2f}, y_{2f}, d_2$, and d_3 are known to the computational system of ordinary follower. Hence, $\|\mathrm{d}\vec{s_3}\|$ is defined as the distance between the final and initial position of ordinary follower. Here, $\vec{s_3}(t)$ a vector field along the trajectory curve of ordinary follower. It should be noted that $(x_{3\,\mathrm{In}}(t), y_{3\,\mathrm{In}}(t))$ and $(x_{3f}(t), y_{3f}(t))$ are always on $\vec{s_3}(t)$. Then, we define

$$\|\mathrm{d}\vec{s_3}\|^2 = \left(x_{3f}(t), x_{3\,\mathrm{In}}(t)\right)^2 + \left(y_{3f}(t), y_{3\,\mathrm{In}}(t)\right)^2. \tag{10}$$

Actually, (9) are equations of two circles. They meet at two different points. The ordinary follower will follow the leader and first follower maintaining safe motion and moves to any one meeting point such that $\|\mathrm{d}\vec{s_3}\|$ is minimum. By maintaining $\|\mathrm{d}\vec{s_3}\|$ minimum, ordinary follower moves along shortest path to its final position. Now, it is the need to propose a control law for motion of first follower satisfying the aforesaid conditions. Actually, the whole problem may be treated as an optimization problem, where minimization of objective function (10) under equality constraint (9) should be performed. And a control law based on this optimization is presented as

$$\vec{v_3}(t) = \frac{\mathrm{d}\vec{s_3}}{\mathrm{d}t} = \frac{\mathrm{d}x_3}{\mathrm{d}t}\hat{i} + \frac{\mathrm{d}y_3}{\mathrm{d}t}\hat{j} = v_{3x}(t)\hat{i} + v_{3y}(t)\hat{j}, \tag{11}$$

where, $\mathrm{d}x_3/\mathrm{d}t = v_{3x}(t)$, $\mathrm{d}y_3/\mathrm{d}t = v_{3y}(t)$, $\hat{i}$ and $\hat{j}$ are unit vectors along x, and y direction of the global coordinate system.

The translational velocity control input $\|\vec{v_3}\| = \sqrt{(v_{3x}(t))^2 + (v_{3y}(t))^2}$ m/second.

Angular velocity (rad./sec) control input is

$$\begin{aligned}\omega_3(t) &= \tan^{-1}\left|\frac{\mathrm{d}y_3}{\mathrm{d}x_3}\right| \quad \text{when } \mathrm{d}x_3 \text{ is } +\text{ve},\ \mathrm{d}y_3 \text{ is } +\text{ve}\\ &= \left(\pi - \tan^{-1}\left|\frac{\mathrm{d}y_3}{\mathrm{d}x_3}\right|\right) \quad \text{when } \mathrm{d}x_3 \text{ is } -\text{ve},\ \mathrm{d}y_3 \text{ is } +\text{ve}\\ &= -\left(\pi - \tan^{-1}\left|\frac{\mathrm{d}y_3}{\mathrm{d}x_3}\right|\right) \quad \text{when } \mathrm{d}x_3 \text{ is } -\text{ve},\ \mathrm{d}y_3 \text{ is } -\text{ve}\\ &= -\tan^{-1}\left|\frac{\mathrm{d}y_3}{\mathrm{d}x_3}\right| \quad \text{when } \mathrm{d}x_3 \text{ is} +\text{ve},\ \mathrm{d}y_3 \text{ is } -\text{ve}\\ &= -\frac{\pi}{2} \text{ or } \frac{\pi}{2} \quad \text{when } \mathrm{d}x_3 = 0 \text{ and } \mathrm{d}y_3 \text{ is } +\text{ve or } -\text{ve}.\end{aligned} \tag{12}$$

4.4. Control Law for Ordinary Follower-2. Ordinary follower-2 tries to satisfy two distance constraints; that is, it has two outgoing edges, first one (d_4) is directed towards ordinary follower-1 and second one (d_5) is directed towards first follower. Let the first follower and ordinary follower-1 be placed at their corresponding final positions. The ordinary follower senses the disturbances in position of first follower and ordinary follower-1; that is, it senses error in desired distance constraints d_5 and d_4 by sensing the final position of the first follower and ordinary follower-1. It tries to satisfy these distance constraints to first follower and ordinary follower-1. Therefore, suppose the ordinary follower-2 moves to its final position (rest point) at the next instant of the time (dt) after the instant during which ordinary follower-1 moves to its final position. During this movement of ordinary follower-2–ordinary follower-1, first follower and leader are assumed to be stationary at the position $(x_{3f}(t), y_{3f}(t))$, $(x_{2f}(t), y_{2f}(t))$, and $(x_{1f}(t), y_{1f}(t))$, respectively. At the final position, the ordinary follower satisfies its distance constraints. Here, a condition is given to the ordinary follower such that only when disturbances in positions of both leader as well as first follower (not merely leader) occur, ordinary follower changes its position to final one. This final position is assumed $(x_{4f}(t), y_{4f}(t))$. The assumed initial position of the ordinary follower is $(x_{4\,\mathrm{In}}(t), y_{4\,\mathrm{In}}(t))$. Now according to the desired distance constraints for it, two conditions are to be satisfied as given in

$$\begin{aligned}\left(x_{2f}(t) - x_{4f}(t)\right)^2 + \left(y_{2f}(t) - y_{4f}(t)\right)^2 - d_5^2 = 0,\\ \left(x_{3f}(t) - x_{4f}(t)\right)^2 + \left(y_{3f}(t) - y_{4f}(t)\right)^2 - d_4^2 = 0,\end{aligned} \tag{13}$$

where $x_{2f}, y_{2f}, x_{3f}, y_{3f}, d_4$, and d_5 are known to the computational system of ordinary follower. Hence, $\|\mathrm{d}\vec{s_4}\|$ is defined as distance between the final and initial position of ordinary follower. Here, $\vec{s_4}(t)$ is a vector field along the trajectory curve of ordinary follower. It should be noted that $(x_{4\,\mathrm{In}}(t), y_{4\,\mathrm{In}}(t))$ and $(x_{4f}(t), y_{4f}(t))$ are always on $\vec{s_4}(t)$. Then we define

$$\|\mathrm{d}\vec{s_4}\|^2 = \left(x_{4f}(t), x_{4\,\mathrm{In}}(t)\right)^2 + \left(y_{4f}(t), y_{4\,\mathrm{In}}(t)\right)^2. \tag{14}$$

Actually, (13) are equations of two circles. They meet at two different points. The ordinary follower will follow the leader and first follower maintaining safe motion and moving to any one meeting point such that $\|d\vec{s_4}\|$ is minimum. By maintaining $\|d\vec{s_4}\|$ minimum, ordinary follower moves along shortest path to its final position. Now it is the need to propose a control law for motion of first follower satisfying aforesaid conditions. Actually, the whole problem may be treated as an optimization problem where minimization of objective function (14) under equality constraint (13) should be performed. And a control law based on this optimization is presented as

$$\vec{v}_4(t) = \frac{d\vec{s_4}}{dt} = \frac{dx_4}{dt}\hat{i} + \frac{dy_4}{dt}\hat{j} = v_{4x}(t)\hat{i} + v_{4y}(t)\hat{j}, \quad (15)$$

where $dx_4/dt = v_{4x}(t)$ and $dy_4/dt = v_{4y}(t)$ and $\hat{i}$ and $\hat{j}$ are unit vectors along x and y directions of the global coordinate system.

The translational velocity control input $\|\vec{v}_4\| = \sqrt{(v_{4x}(t))^2 + (v_{4y}(t))^2}$ m/second.

Angular velocity (rad./sec) control input is

$$\begin{aligned}
\omega_4(t) &= \tan^{-1}\left|\frac{dy_4}{dx_4}\right|, \quad \text{when } dx_4 \text{ is } +\text{ve},\ dy_4 \text{ is } +\text{ve} \\
&= \left(\pi - \tan^{-1}\left|\frac{dy_4}{dx_4}\right|\right), \quad \text{when } dx_4 \text{ is } -\text{ve},\ dy_4 \text{ is } +\text{ve} \\
&= -\left(\pi - \tan^{-1}\left|\frac{dy_4}{dx_4}\right|\right), \quad \text{when } dx_4 \text{ is } -\text{ve},\ dy_4 \text{ is } -\text{ve} \\
&= -\tan^{-1}\left|\frac{dy_3}{dx_3}\right|, \quad \text{when } dx_3 \text{ is } +\text{ve},\ dy_3 \text{ is } -\text{ve} \\
&= -\frac{\pi}{2} \text{ or } \frac{\pi}{2}, \quad \text{when } dx_3 = 0 \text{ and } dy_3 \text{ is } +\text{ve or } -\text{ve}.
\end{aligned} \quad (16)$$

Remarks 1. Hence, from the previous discussion it is concluded that for each complete displacement of considered quadrilateral formation, at the end of first instant of time dt the leader moves to its desired final position. Then, at the end of next instant dt (which is the second one) the first follower moves to its final desired position to maintain distance constraints to the leader, during which leader is kept stationary. At end of another instant dt (which is third one) the ordinary follower-1 reaches to its final position to maintain distance constraints to both leader and first follower. At end of another instant dt (which is fourth one) the ordinary follower-2 reaches to its final position to maintain distance constraints to both leader and first follower. Therefore, the agents are not reaching their corresponding final position exactly at the same time. Consequently, during the period from "after the starting of first instant" and "before the end of fourth instant" desired distances are not preserved among the agents, rather the first follower, ordinary follower-1 and ordinary follower-2 try to form up. So it may be concluded that after every $4 * dt$ time, the desired formation is obtained. Therefore, each agent start to move to its new position (to be in a new position set) after every $3 * dt$ time. That is there is a discontinuous motion that occurs for every agent. Therefore, for formation of n number of agents, after every $n * dt$ time the desired formation is maintained. Each agent start to move to its new position (to be in a new position set) after every $(n-1) * dt$ time. However, if the dt is chosen as very small we may assume that all the agents reach their corresponding final (new) positions during first instant of time dt (almost same time taken by leader to reach its desired final position) and during each complete displacement of formation, and continuous motion of formation is maintained. Consequently we may also assume all the agents move with continuous motion. Simulation results in the next section are also done based on this assumption.

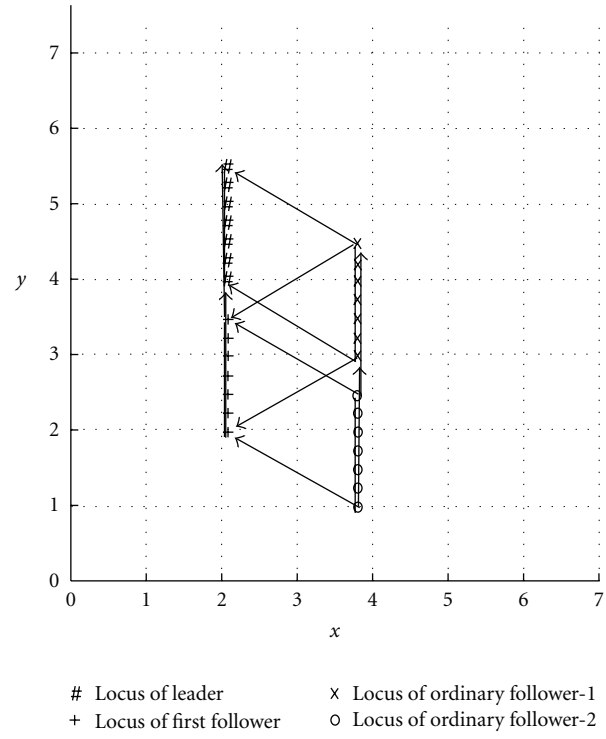


Figure 5: Straight line motion of formation of four agents with leader-follower structure.

5. Simulation Studies

The control laws (3), (7), (11), and (15) for different agents have been tested successfully via three cases of simulations for specified formation with consideration of $d_1 = d_2 = d_3 = d_4 = d_5 = 2$ meters. Length of each time instant is considered as 0.01 second for simulation during optimization as described in what follows.

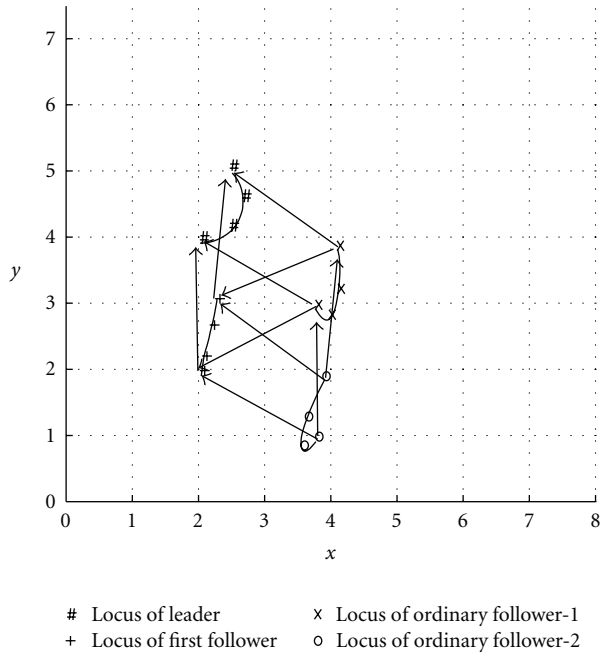


Figure 6: Motion of formation of four agents with leader-follower structure, with 0.09 degree (approximately 0.00157 radian) orientation at each instant of time provided in the motion of leader.

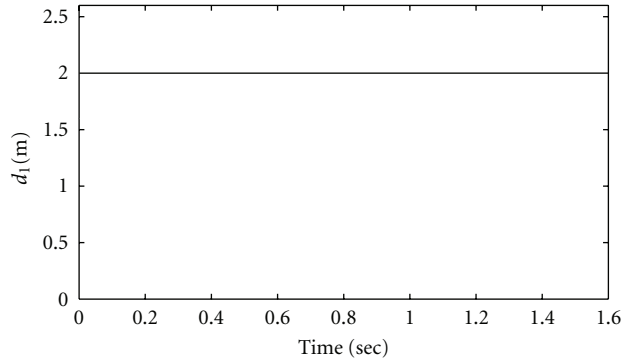


Figure 7: Plot of distance between leader and first follower versus time.

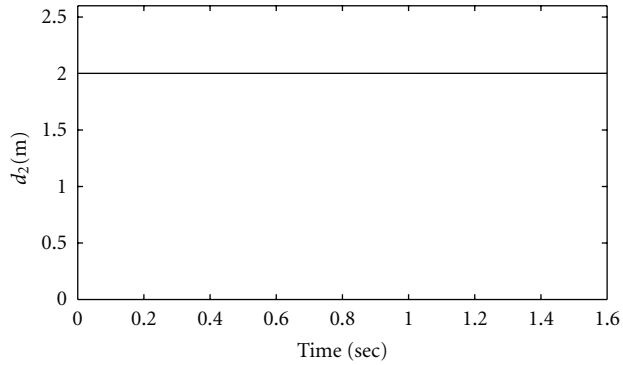


Figure 8: Plot of distance between first follower and ordinary follower 1 versus time.

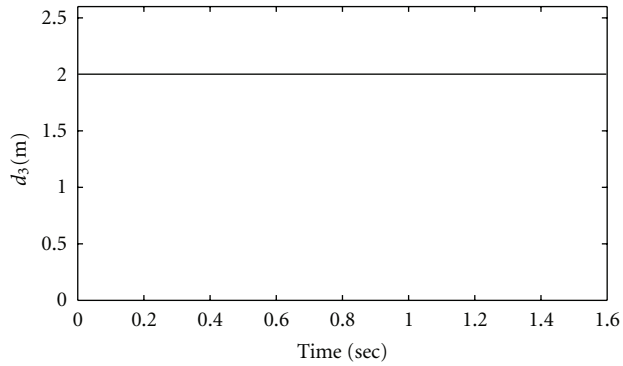


Figure 9: Plot of distance between leader and ordinary follower 1 versus time.

5.1. Specific Assumptions in Different Cases

5.1.1. Case I and Case II. In Case I and Case II, the assumptions are as follows:

(i) Initial position coordinates are $(2,4)$, $(2,2)$, $(3.732,3)$, and $(3.732,1)$ for leader, first follower, Ordinary Follower-1, and Ordinary Follower-2, respectively;

(ii) Translational velocity input to the leader is 1 meter/second;

(iii) Distance travelled by leader is 1.5 meter in each case;

5.1.2. Case III. Here the considerations are as follows:

(i) Initial position coordinates are $(2,4)$, $(2,2)$, $(3.732,3)$, and $(3.732,1)$ for leader, First follower, Ordinary Follower-1, and Ordinary Follower-2, respectively;

(ii) Leader is assumed to move along a sinusoidal path. The equations that describe that sinusoidal path of the leader is considered as

$$x(t) = 0.03t \text{ meter},$$
$$y(t) = \sin(0.03t) \text{ meter};$$
(17)

(iii) Time travelled by the leader is 100 second;

Control laws given in (3), (7), (11), and (15) require the final position of the corresponding agent at each instant of time during their motion. For the leader, the final position at each instant of time is available as the path is specified for it, but for other agents, these positions must be calculated. To find out the final position at each instant of time t, the controller in each case requires optimization of an quadratic objective function under one or two quadratic equality constraints as described in Section 4. Several optimization methods are available for this purpose. Our choice here is to exploit Sequential Quadratic Programming (SQP) as it is one of the most popular and robust algorithms for nonlinear continuous optimization. The method is based on solving a series of subproblems designed to minimize a quadratic model of the objective subject to a linearization

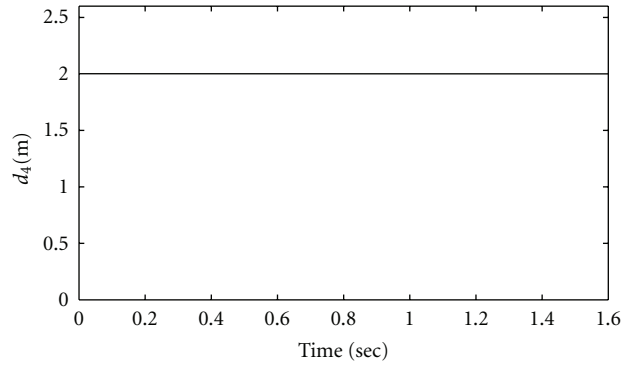


Figure 10: Plot of distance between ordinary follower 1 and ordinary follower 2 versus time.

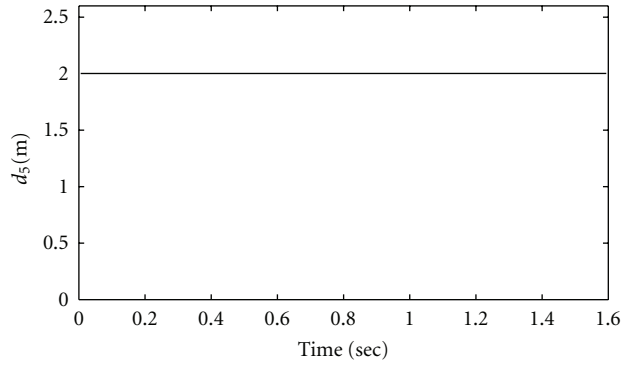


Figure 11: Plot of distance between first follower and and ordinary follower 2 versus time.

of the constraints. In the proposed controller, the objective functions are chosen as quadratic whilst the constraints are taken as nonlinear quadratic which can be linearized during course of optimization procedure. At the beginning of each instant of time t (i.e., at the beginning of a complete displacement of the whole formation), the position of each agent is used as the initial position in control law of that particular agent. This position coordinate is also assumed as starting point of that agent's complete iterative procedure (in optimization process using SQP) for finding out its final position.That iterative procedure follows the steps elaborated below:

(i) making a Quadratic Programming (QP) subproblem (based on a quadratic approximation of the Lagrangian function) using nonlinear objective function and equality constraints,

(ii) solving that Quadratic Programming (QP) sub problem at each iteration,

(iii) during (ii) updating an estimate of the Hessian of the Lagrangian at each iteration using the BFGS (Broyden–Fletcher–Goldfarb–Shanno) formula [16, 17],

(iv) quadratic Programming solution at each iteration performing appropriate Line Search using Merit Function [16–18].

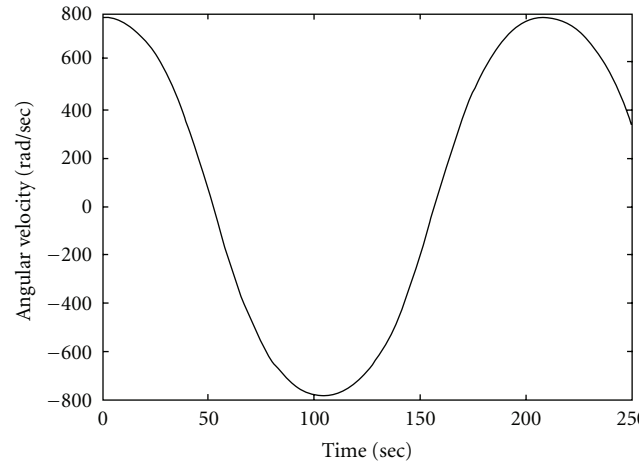


Figure 12: Plot of angular velocity (rad/second) versus time (second) for sinusoidal motion of leader.

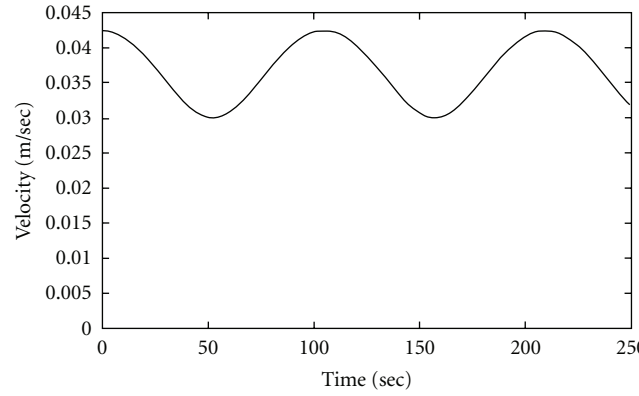


Figure 13: Plot of velocity versus time for sinusoidal motion of leader.

In this work, MATLAB optimization toolbox has been used for solving the above said optimization problem using Sequential Quadratic Programming. However, alternatively other packages such as NPSOL, NLPQL, OPSYC, and OPTIMA can be used.

5.2. Description of Simulation Result in Different Cases

5.2.1. Case I. We provide straight line motion to the Leader such that it moves along global x-axis. The paths followed by all agents during motion of formation are shown in Figure 5. Distances (d_1, d_2, d_3, d_4, and d_5) are observed maintained at specified values.

5.2.2. Case II. We provide 0.09 degree (approximately 0.00157 rad/sec) angular velocity input to the motion of leader.The translational velocity input is constant at 1 m/s. In this case complete path of the leader may be considered as part of a complete circle. For this case paths of all agents during motion are shown in Figure 6. From the simulation studies of Case II it is found that the distances among agents

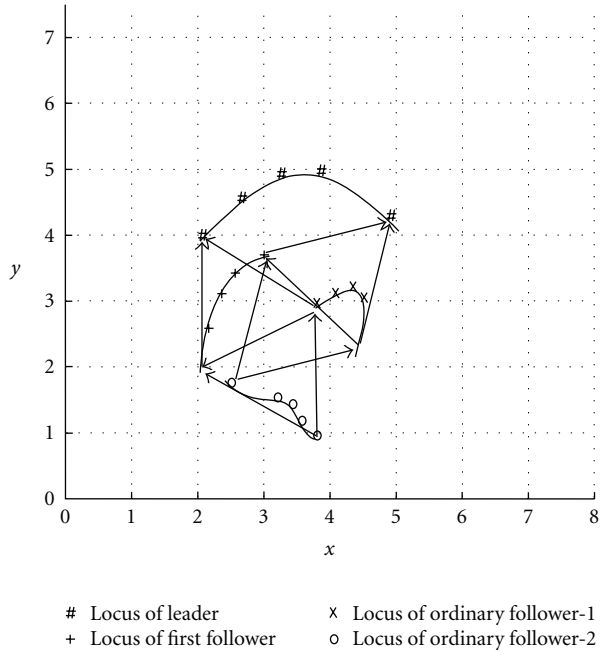


Figure 14: Motion of triangular formation of leader-follower structure, with sinusoidal motion of leader. In this figure during motion of formation, paths of leader, first follower, and ordinary follower are shown. During motion interagent distances are maintained at desired value.

are maintained at the desired values during the motion of the formation as in case of Case-I. Plots of d_1 versus t, d_2 versus t and d_3 versus t, d_4 versus t, d_5 versus t are shown in Figures 7, 8, 9, 10, and 11, respectively. These plots are also observed in Case-I also.

5.2.3. Case III. As the path of leader is sinusoidal the velocity and orientation (corresponding angular velocity) change at each instant. The changes in translational velocity and angular velocity (which are control inputs for leader) along sinusoidal path are shown in Figure 12, Figure 13, and Figure 10. for travelling time 250 seconds. For this case path followed by first follower and ordinary follower along with leader are shown in Figure 11. Here also in this case similar observations are made on the maintaining of the distance constraints during the motion of the formation as in Case I and Case II.

6. Conclusions

In this paper, a new algorithm using a set of decentralized control laws based on optimization (using Sequential Quadratic Programming) of distance constraints has been proposed for the motion control of a *leader-follower* structure type formation of multiple mobile autonomous agents. The effectiveness of the proposed control schemes have been demonstrated through a number simulation studies. During the motion of formation, the interagent distances are maintained at desired values.

The above described control design strategy may be extended to the formation of any number of autonomous agents in the form of leader-follower structure in which each agent (except First Follower) observes the distances of only two neighboring agents to which it needs to maintain distance constraints. The control strategy considering kinematics (other than unicycle nonholonomic point model) of each agent and phenomenon of obstacle avoidance are currently in contemplation. The future work in this context may be to pursue the stability study of proposed control design.

References

[1] R. Carelli, C. de la Cruz, and F. Roberti, "Centralized formation control of non-holonomic mobile robots," *Latin American Applied Research*, vol. 36, no. 2, pp. 63–69, 2006.

[2] C. de la Cruz and R. Carelli, "Dynamic modeling and centralized formation control of mobile robots," in *Proceedings of the 32nd Annual Conference on IEEE Industrial Electronics (IECON '06)*, pp. 3880–3885, November 2006.

[3] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[4] J. R. T. Lawton, R. W. Beard, and B. J. Young, "A decentralized approach to formation maneuvers," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 6, pp. 933–941, 2003.

[5] S. Sandeep, B. Fidan, and C. Yu, "Decentralized cohesive motion control of multi-agent formations," in *Proceedings of the 14th Mediterranean Conference on Control and Automation (MED'06)*, June 2006.

[6] J. Baillieul and A. Suri, "Information patterns and hedging Brockett's theorem in controlling vehicle formations," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 1, pp. 556–563, December 2003.

[7] C. Yu, J. M. Hendrickx, B. Fidan, B. D. O. Anderson, and V. D. Blondel, "Three and higher dimensional autonomous formations: rigidity, persistence and structural persistence," *Automatica*, vol. 43, no. 3, pp. 387–402, 2007.

[8] J. M. Hendrickx, B. D. O. Anderson, J.-C. Delvenne, and V. D. Blondel, "Directed graphs for the analysis of rigidity and persistence in autonomous agent systems," *International Journal of Robust and Nonlinear Control*, vol. 17, no. 10-11, pp. 960–981, 2007.

[9] J. M. Hendrickx, B. Fidan, C. Yu, B. D. O. Anderson, and V. D. Blondel, "Elementary operations for the reorganization of minimally persistent formations," in *Proceedings of the 17th International Symposium on. Mathematical Theory of Networks and Systems (MTNS '06)*, pp. 859–873, Kyoto, Japan, 2006.

[10] B. D. O. Anderson, C. Yu, B. Fidan, and J. M. Hendrickx, "Rigid graph control architectures for autonomous formations," *IEEE Control Systems Magazine*, vol. 28, no. 6, pp. 48–63, 2008.

[11] T. Eden, W. Whitely, B. D.O. Andersoon, A. S. Morse, and P. N. Belhumeur, "Information structures to secure Control of rigid formations with leader-follower architecture," in *Proceedings of the American Control Conference*, vol. 4, pp. 2966–2971, Portland, Ore, USA, June 2005.

[12] B. D. O. Anderson, C. Yu, S. Dasgupta, and A. Stephen Morse, "Control of a three-coleader formation in the plane," *Systems and Control Letters*, vol. 56, no. 9-10, pp. 573–578, 2007.

[13] M. Cao, A. S. Morse, C. Yu, B. D. O. Anderson, and S. Dasgupta, "Controlling a triangular formation of mobile autonomous agents," in *Proceedings of the 46th IEEE Conference on Decision and Control (CDC '07)*, pp. 3603–3608, December 2007.

[14] L. Krick, M. E. Broucke, and B. A. Francis, "Stabilization of infinitesimally rigid formations of multi-robot networks," *International Journal of Control*, vol. 82, no. 3, pp. 423–439, 2008.

[15] B. D. O. Anderson, S. Dasgupta, and C. Yu, "Control of directed formations with a leader-first follower structure," in *Proceedings of the 46th IEEE Conference on Decision and Control (CDC '07)*, pp. 2882–2887, New Orleans, La, USA, December 2007.

[16] M. J. D. Powell, "A fast algorithm for nonlinearly constrained optimization calculations," in *Numerical Analysis*, G. A. Watson, Ed., vol. 630 of *Lecture Notes in Mathematics*, Springer, London, UK, 1978.

[17] M. J. D. Powell, "The Convergence of variable metric methods for nonlinearly constrained optimization calculations," in *Nonlinear Programming 3*, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, Eds., Academic Press, Boston, Mass, USA, 1978.

[18] S. P. Han, "A globally convergent method for nonlinear programming," *Journal of Optimization Theory and Applications*, vol. 22, no. 3, pp. 297–309, 1977.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 465618, 8 pages
doi:10.1155/2010/465618

Research Article

Estimating Ground Inclination Using Strain Sensors with Fourier Series Representation

Wolfgang Svensson and Ulf Holmberg

Intelligent System Laboratory, IDE Department, Halmstad University, Box 823, 301 18 Halmstad, Sweden

Correspondence should be addressed to Ulf Holmberg, ulf.holmberg@hh.se

Received 1 November 2009; Accepted 25 January 2010

Academic Editor: Noriyasu Homma

An embedded measurement system for foot orthosis during gait is proposed. Strain gauge sensors were mounted on a foot orthosis to give information about strain in the sagittal plane. The ankle angle of the orthosis was fixed and strain characteristics were therefore changed when walking on slopes. With a Fourier series representation of the strain during a gait cycle, ground angle at different walking speeds and inclinations could be estimated with similar accuracy as previous studies using kinematically based estimators. Furthermore, if the angle of the mechanical foot ankle was changed, the sensing technique still could estimate ground angle without need for recalibration as opposed to kinematical sensors. This indicates that embedded strain sensors can be used for online control of future orthoses with inclination adaptation. Also, there would be no need to recalibrate the sensing system when changing shoes with different heel heights.

1. Introduction

Ankle and foot muscles disorders affect the human gait and are commonly treated with orthoses to partially compensate functional loss. With an ankle-foot orthosis, AFO, typical assisting functions are provide ankle stability during stance, stimulate push-off effect during late stance, keep the toes off ground during swing, assist poor functional muscles, and decrease pain by limiting motion. Orthoses have been passive and purely mechanical. But decreasing size and cost of electronics have made it possible for active solutions of assistance, for example, [1–3]. But existing systems are still limited in their capability of adapting to new inclining circumstances, that is, hills or heel height variation.

Studies have shown that when able-bodied walked uphill, gait was adjusted in the ankle foot system [4]. Downhill walking is considered to be different in the sense that joints have to absorb more energy caused by the combined forward and downward movement (see, e.g., [5–7]). This can be seen by a shorter stride length [5] and is compensated mostly by the knees and somewhat by the hip [4].

It is stated in [8] that people with reduced range of motion in one joint compensate by using their other joints. But with elderly, orthotic, and prosthesis users, these changes in flexion may not be possible. Many of the assisting devices have fixed ankle position and attempting to move the body's center of mass forward may cause a sense of *instability* when walking in inclinations. Our hypothesis is that *this restricted adaptation possibility causes an extra torque* on the AFO. This would therefore be measurable and useful for estimating the ground angle. In the future this could be used in ankle control for adapting to the ground inclination.

Portable gait measuring techniques are interesting both for gait analysis and active control. Automatic classification of gait phases has previously been done using various wearable sensors, for example, force resistive sensors (FSR) [9, 10], gyros [11], combinations of FSRs and gyros [12], accelerometers [13–15], or goniometers [16]. It has also been shown that upper-body mounted accelerometers together with artificial neural networks can be used for slope and speed estimation. Aminian et al. [17] used summary statistics (median, variance, etc.) for inclination estimation and Wang et al. [18] used wavelets to separate slope from level walking. Sabatini et al. [15] mounted accelerometers on the foot and used them as gravity sensors during stance.

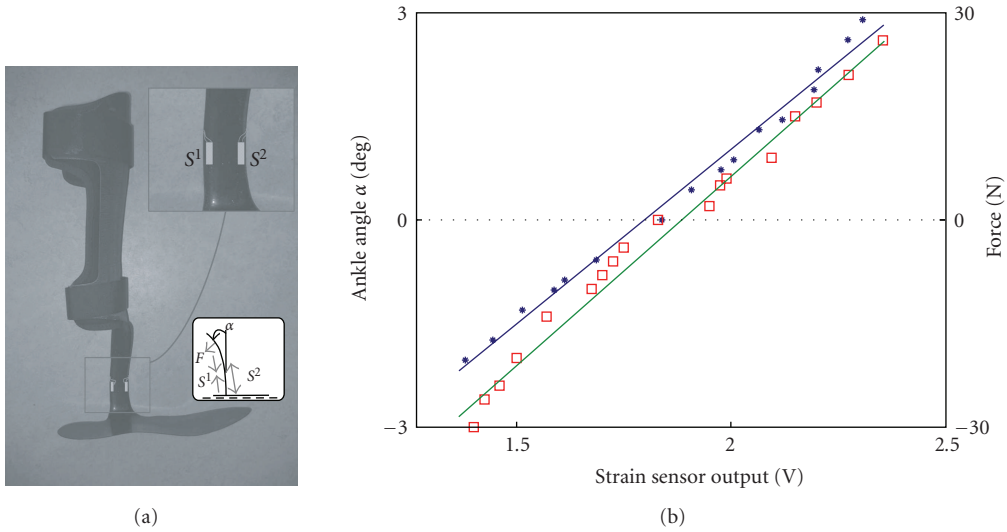


FIGURE 1: (a) Strain measurement setup. One strain gauge sensor S^1 is attached partly on the inside at the back and one S^2 partly on the inside at the front. Sensors S^3 and S^4 are attached at the corresponding positions but on the outside. (b) Measured ankle angle α (stars) and force F (squares) relation to AFO strain sensor output. Estimated linear relations: solid (α) and dashed (F).

Torque is typically estimated from measuring ground reaction forces and ankle angle. Instead of using previously mentioned sensors, the use of strain gauge sensors as an indirect measurement is investigated. Another feature of strain sensors is the small size that allows embedded constructions (see Figure 1). These sensors can be valuable for active control as well as for prosthetic and orthotic design.

Previous studies showed that gait cycle strain peak and mean value changed in inclination [19]. But similar changes in strain can also be caused by a changed gait speed. In this work we propose to take advantage of the fact that gait cycles are periodic. A Fourier series representation can then be used to code the information into fewer variables. This is then used for estimation of the ground inclination. It extends the work in [19] and shows that a fixed estimation model can be used even though the speed changes. In addition and more importantce, the estimation works also when the ankle angle is changed. This indicates that online ankle angle control would be possible using the strain signal and that no recalibration would be necessary when changing shoes to other heel heights.

2. Strain Measurement

2.1. Strain Sensor Setup. A solid light-weight carbon ankle-foot orthosis was used. Four strain gauge sensors were glued 60 mm above the orthotic sole as shown in Figure 1(a). The sensor signals were combined into one via a (full Wheatstone) bridge. The combined amplified sensor signal was sampled with an off-the-shelf *PIC* processor with a 10-bit AD-converter at 50 Hz sampling frequency. The signal was logged into a PC with *Sysquake* software using *Bluetooth* modules. The strain sensor signals are combined into one strain signal describing the angle behavior in the sagittal (vertical-longitudinal) plane during gait.

A torque applied on an AFO can be modeled as a force F, acting on a beam with one fix end as shown in Figure 1(a), where the beam bending angle α is proportional to the force for small angles. Instead of measuring the angle α and the force F or trying to reconstruct the corresponding torque, the attached strain sensors are used to produce information about the orthosis bending during gait.

Experiments were made to verify the strain signal being proportional to both applied force F and angle α in Figure 1(b).

2.2. Strain Signal Behavior. From Figure 1(b) it is seen that no strain (when the orthosis is unloaded) corresponds to a sensor output around 1.8 Volts. The sensor signal increased at plantar flexion and decreased at dorsiflexion; see Figure 2. At heel strike the strain increased to a peak from being unloaded as the foot was put down in front of the body. The foot was then lowered and the trunk moved forward decreasing the strain. The body continued forward and the leg tried to rotate with an ankle dorsiflexion. An oscillation in the sensor signal was sometimes seen around the time of heel lift. In the end of the stance phase, an abrupt decrease of the strain signal occurred. This was the largest load on the orthosis caused by the push-off. After toe-off, the inherent orthotic elasticity brought the orthosis to unload. Although the sensor setup was designed to mainly measure in the sagittal (vertical-longitudinal) plane it was also somewhat influenced by the natural motion in the coronal (vertical-transversal) plane during weight acceptance.

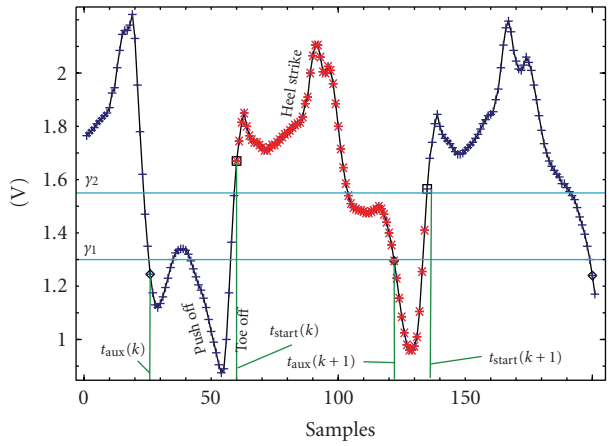


FIGURE 2: AFO strain signal output and extraction of strain signal for one gait cycle (red (∗)). Thresholds γ_1, γ_2 for finding the auxiliary times t_{aux} (first t when $S(t) < \gamma_1$) and t_{start} (first t when $S(t) > \gamma_2$) are indicated.

3. Methods

3.1. Strain Signal Extraction. In order to extract one gait cycle from the strain sensor output, an appropriate starting point is selected to be particularly easy to detect. This is chosen to be after toe-off since here the strain sensor signal is making an abrupt monotonous increase. However, there are sometimes oscillations around heel lift with the sensor signal increasing which should not be misinterpreted as toe lift (see Figure 2). By introducing two thresholds, the time instant just after toe-off can be found. The procedure is as follows.

Denote the strain sensor output $S(t)$ with sample index t (integer). Let $t_{start}(0)$ be any initial sampling instant. Find the start of gait cycle k, defined at $t_{start}(k)$ by using auxiliary sampling instant $t_{aux}(k)$, according to

$$t_{aux}(k) = \min_{t>t_{start}(k-1)} \{t \mid S(t) < \gamma_1\},$$
$$t_{start}(k) = \min_{t>t_{aux}(k)} \{t \mid S(t) < \gamma_1\}. \quad (1)$$

The *strain signal* (vector) at gait cycle k is now defined as

$$\mathbf{s}_k = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{pmatrix} = \begin{pmatrix} S(t_{start}(k)) \\ S(t_{start}(k)+1) \\ \vdots \\ S(t_{start}(k+1)-1) \end{pmatrix}. \quad (2)$$

The index k will be omitted for simplicity except when the gait number is specific for the expression.

3.2. Fourier Series Representation. Since the dimension (N) of the strain vector is changing with the walking speed, it is inconvenient to use the strain vector $\mathbf{s}$ directly in an estimation scheme. However, due to the periodicity of the strain with $S(t+N) = S(t)$, it would make sense to use Fourier series representation. Then the gait information in the strain signal is coded into fewer variables efficiently and with a vector representation with a fixed prechosen dimension.

Using n harmonics, the Fourier coefficients can be written in matrix form as

$$x = \begin{pmatrix} a_0 \\ a \\ b \end{pmatrix} = \begin{pmatrix} \frac{1}{N}\sum_{i=1}^{N} s_i \\ \frac{2}{N}\mathcal{C}^T\mathbf{s} \\ \frac{2}{N}\mathcal{S}^T\mathbf{s} \end{pmatrix}, \quad (3)$$

where the matrixes $\mathcal{C} \in \Re^{N\times n}$ and $\mathcal{S} \in \Re^{N\times n}$ have the elements $\mathcal{C}_{ij} = \cos(ij\Omega), \mathcal{S}_{ij} = \sin(ij\Omega)$, and $\Omega = 2\pi/N$. Letting $\mathbf{1} = [1 \cdots 1]^T \in \Re^{N\times 1}$, the basis functions can be put in a matrix where each column corresponds to a basis function

$$E = \begin{pmatrix} \mathbf{1} & \mathcal{C} & \mathcal{S} \end{pmatrix} \in \Re^{N\times(2n+1)}. \quad (4)$$

The strain can then be approximated as

$$\hat{\mathbf{s}} = Ex \in \Re^{N\times 1} \quad (5)$$

which is the orthogonal projection to the space spanned by the $2n+1$ basis functions, that is,

$$x = \arg\min_x ||\mathbf{s} - \hat{\mathbf{s}}||^2 \in \Re^{2n+1}. \quad (6)$$

Notice that the dimension of x is fixed and chosen ($= 2n+1$) and usually much smaller than the dimension of $\mathbf{s}$ ($= N$), which also varies from step to step.

3.3. Estimation of Inclination. Assume that the ground angle ϕ can be modeled linearly by the parameter vector θ from the Fourier coefficients x as

$$\phi_k = x_k^T\theta + \epsilon_k, \quad (7)$$

where ϵ_k is stochastic white noise and

$$\theta = \begin{pmatrix} \theta_1 & \cdots & \theta_{2n+1} \end{pmatrix}^T. \quad (8)$$

The model parameters θ are found using M steps data

$$\phi = \begin{pmatrix} \phi_1 & \cdots & \phi_M \end{pmatrix}^T,$$
$$X = \begin{pmatrix} x_1 & \cdots & x_M \end{pmatrix}^T \longrightarrow \phi = X\theta + \epsilon, \quad (9)$$
$$\epsilon = \begin{pmatrix} \epsilon_1 & \cdots & \epsilon_M \end{pmatrix}^T$$

by the least squares method as

$$\theta = \arg\min_\theta \left(\epsilon^T\epsilon\right) = (X^TX)^{-1}X^T\phi. \quad (10)$$

Inclination can then be estimated from strain according to

$$\hat{\phi}_k = x_k^T\theta. \quad (11)$$

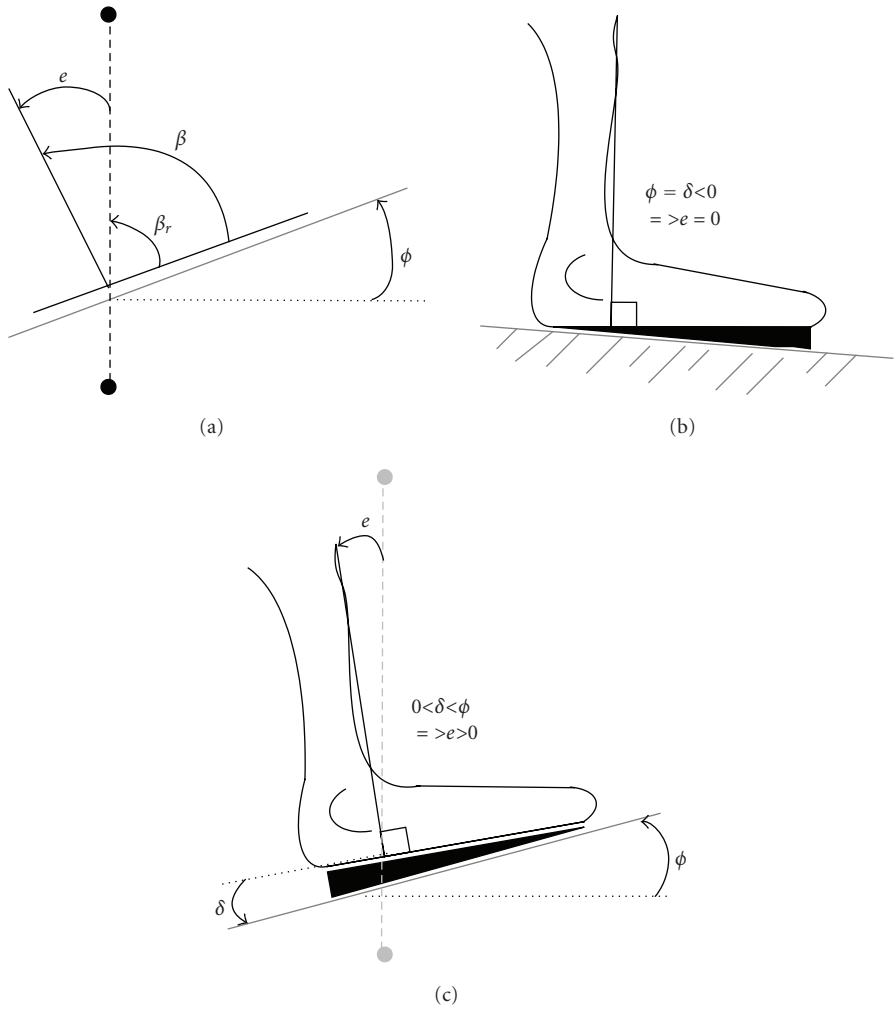


Figure 3: (a) Reference ankle angle β_r and actual angle β of an unloaded AFO give a control error e in inclinations. Below: Ankle position of a biological foot during mid-stance while using compensating keels with resulting control error e. (b) The keel angle δ is chosen so that $e = 0$ and (c) is the error shown as the angle between the actual orthosis and the vertical "desired" direction (dotted).

To quantify the estimation quality the root mean square error RMSE is used:

$$\text{RMSE} = \sqrt{\frac{\sum_{k=0}^{M-1} \left(\phi_k - \hat{\phi}_k\right)^2}{M}}. \tag{12}$$

3.4. Estimation of Control Error. The main motivation of this study was to find a way to estimate a control error suitable to be used for adjusting an ankle angle such that an orthosis or prosthesis can be designed to adapt to changes of ground slope inclinations. Previous studies have suggested a controller which tried to keep the lower leg motion adjusted to an imagined vertical line [20]. That suggests, as shown in Figure 3, that the AFO ankle angle β is adjusted so that it during mid-stance, when it is unloaded, becomes equal to β_r. Thus, the system has an error defined as

$$e(k) = \beta(k) - \beta_r(k) = \beta - \frac{\pi}{2} + \phi \tag{13}$$

and a control law is then

$$\beta(k+1) = \beta(k) - e(k). \tag{14}$$

Since the control error e is not known, it needs to be estimated. Notice that when $\beta = \pi/2$ it follows from (13) that $e = \phi$. *Assume the strain changes in the same way by the ground inclination as by a corresponding change in ankle angle.* If so, then the control error can be estimated by the same estimator as that used for inclination previously. Thus,

$$\hat{e}(k) = x_k^T \theta \tag{15}$$

which would be appropriate also for $\beta \neq \pi/2$.

3.5. Experiment. Data sampling during gait was conducted by letting three healthy male subjects, whose heights are 182–186 cm, walk continuously on a treadmill at different speeds and inclinations. Two test subjects A and B, wore their own walking shoes and one test subject C used sandals with heel straps.

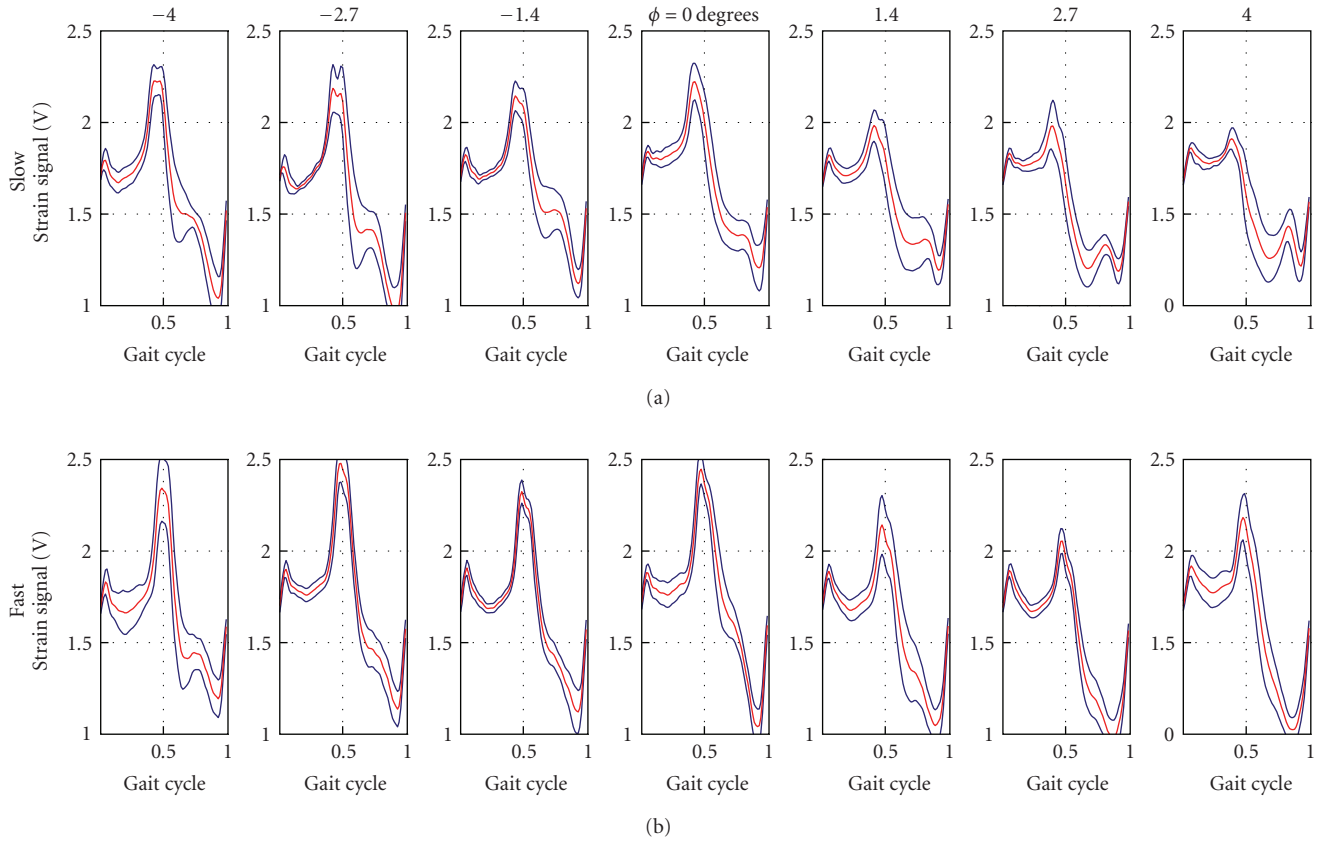


FIGURE 4: Mean strain signal (red) from test subject B, during a normalized gait cycle, from t_{start} to t_{start}, at seven different ground inclinations -4° to 4°, and two speeds: *slow* speed (0.5 m/s) and *fast* speed (1.0 m/s). The standard deviations are plotted in blue.

First, gait experiments were made by letting subject B walk on the treadmill at inclination angles $0^\circ, \pm 1.4^\circ, \pm 2.7^\circ$, and $\pm 4^\circ$. For each inclination two speed conditions were investigated, 0.5 m/s and 1.0 m/s, referred to hereafter as *slow* and *fast*. For each walking situation the gait cycles were extracted and the mean strain signal shape and its standard deviation were calculated. The purpose was to display the strain signal for various situations and attempt to explain intuitively characteristic differences.

Then, gait experiments were made on all subjects A, B and C, in order to investigate the performance of the inclination estimation technique. Ten different experiments were investigated for each one, consisting of the five inclinations $0^\circ, \pm 2.9^\circ$, and $\pm 4.7^\circ$ at the two speeds 0.5 m/s (slow) and 1.0 m/s (fast). Each walking condition was measured during 90 s. Before measuring, each subject walked 10 steps on the treadmill getting familiar with the current speed and inclination. Three different inclination estimators were calculated for each subject. One used only slow speed data, the second used only fast speed data, and the third used both slow and fast speed data for the calculation of the estimator model parameter θ. Since the third estimator is based on two different speeds, it becomes robust to speed variations. By comparing its performance to the estimators turned for one speed only, it is possible to find out what the robustness to speed variations costs in terms of degraded performance. Data were split as 60/40% for estimation and validation, respectively.

Finally, experiments were made to validate the estimation of the control error. The AFO used in the experiments had a fixed ankle angle that could not be adjusted. Instead the ankle was adjusted by inserting a keel as shown in Figure 3. Then the error in (13) can be formulated as

$$e = \phi - \delta, \tag{16}$$

where the δ is the keel angle. In this study the error was estimated from (15) and was compared with the calculated one using (16).

4. Results

4.1. Strain Signal Analysis. Uphill walking affected the strain signal curvature as expected. It was also observed that there was a considerable variation between steps, from heel strike to heel strike. If the ground angle increased, the plantarflexion peak was reduced in magnitude and almost disappeared completely for 4-degree uphill slow walking (Figure 4). This inclination also caused a large strain at dorsiflexion just before push-off. As previous studies have shown, an increase of dorsiflexion is a result of the compensation in the ankle foot system [4].

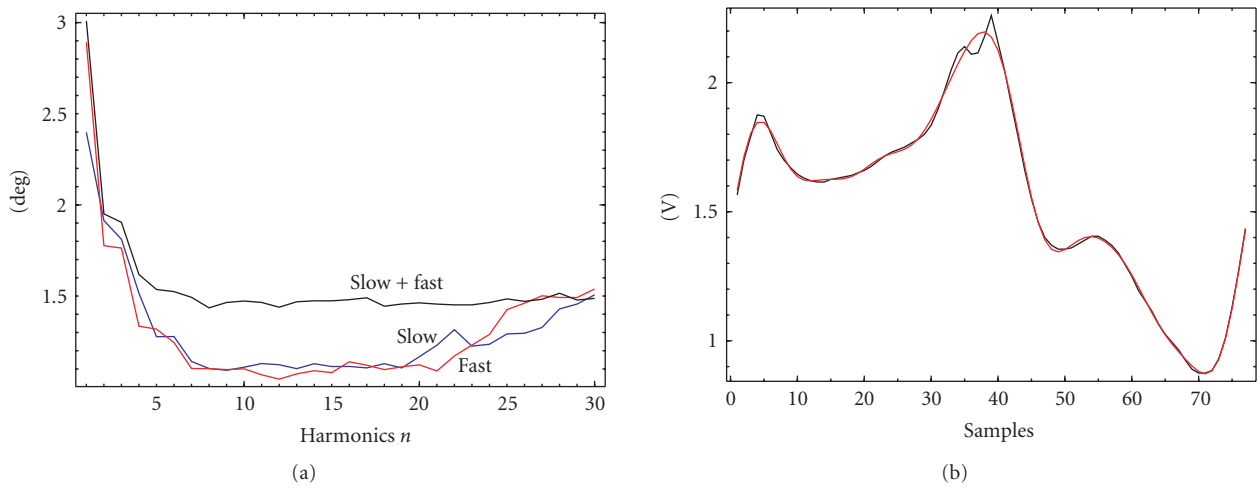


FIGURE 5: (a) RMSE for different Fourier series harmonics n, where the estimator is tuned for *slow*, *fast*, and the combined *slow* + *fast* walking speeds. No significant improvement is made for $n > 8$. For $n = 8$, the speed invariant estimator has RMSE 1.5° compared to 1.2° for the estimators specially tuned for slow or fast walking. (b) Measured strain signal $\mathbf{s}$ (black) and its Fourier series approximation $\hat{\mathbf{s}}$ (red), using $n = 8$ harmonics.

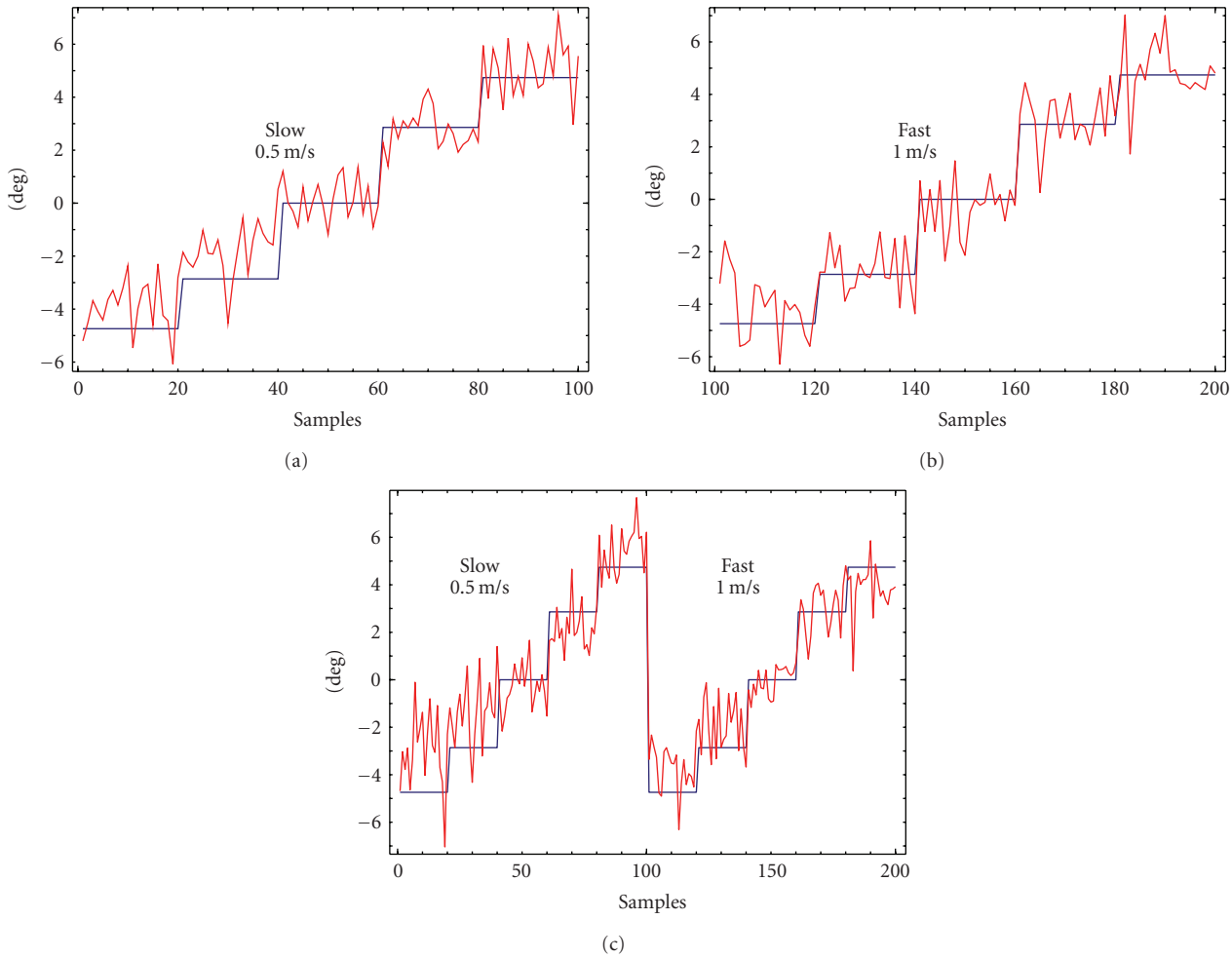


FIGURE 6: Ground angle estimation for subject B, tuned for (a) slow speed, (b) fast speed, and (c) slow + fast speeds.

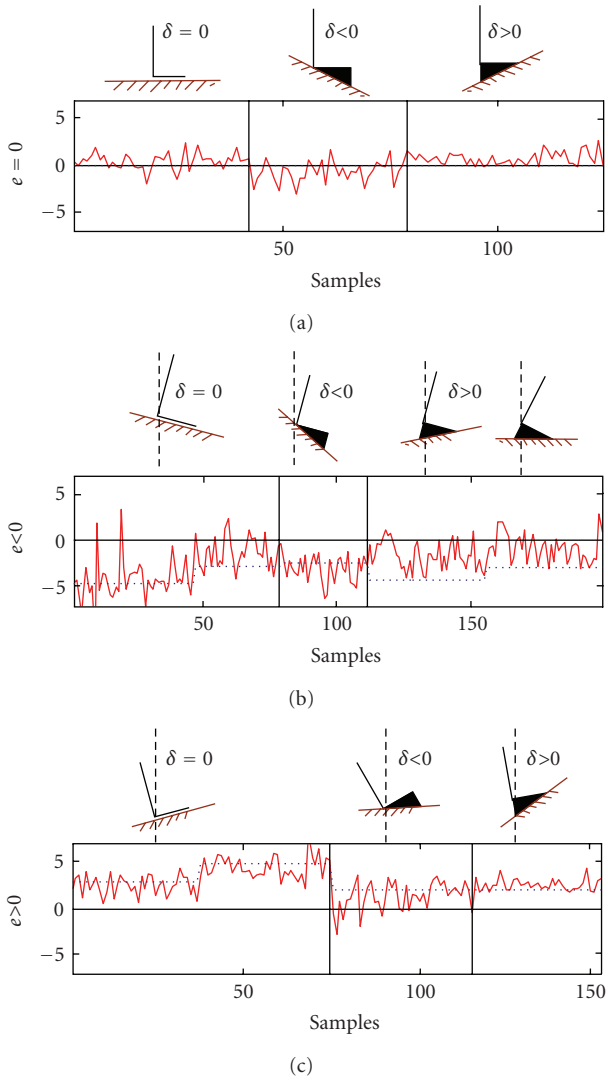


FIGURE 7: Control error of test subject B at different inclinations and control angles (δ: keel angles), theoretical (dotted) calculated from (16) and estimated (solid) during *fast* walking using (15).

In downhill walking, the opposite foot has a lower position than the measured foot during weight transfer. Thus, the dorsiflexion should increase, which also can be observed in the strain signal. But there seems to be a difference between slow and fast walking in that the push-off peak is increased for slow walking while the dorsiflexion strain has increased magnitude mostly at mid-stance for fast walking (Figure 4, $\phi < 0$). No significant difference in plantarflexion could be observed.

The differences between inclination and horizontal walking occur both at *fast* and *slow* walking. With increased speed, the step length increases which would cause a larger foot blade to ground ankle at heel strike. This is observed by the increase in plantarflexion for all inclinations (Figure 4, compare *fast* to *slow*). A faster gait also needs larger push-off force, especially at uphill. This is seen as an increased dorsiflexion (Figure 4, compare *fast* to *slow* for $\phi > 0$).

4.2. Estimation of Inclination. The ground angle estimation performance varied with the number of chosen Fourier harmonics n. RMSE for different harmonics n of the gait cycle is shown in Figure 5(a), where it drops rapidly to a minimum around $n = 8$. For higher harmonics, no significant improvement of the performance is seen Therefore the number of harmonics was chosen to be $n = 8$, resulting in $\dim(x) = 2n + 1 = 17$. The three different estimators, tuned for slow, fast, and the combination slow + fast walking speeds, can also be evaluated in Figure 5(a). At $n = 8$, the RMSE is 1.2° for the single speed estimators while it degrades to 1.5° for the combined estimator based on both speeds. A rather modest cost to pay to get robustness against speed variations. The Fourier series approximated strain signal $\hat{s}$ for $n = 8$ is compared to the measured strain signal s in Figure 5(b). Clearly, most of the strain signal behavior is captured.

The three different inclination estimators, tuned for slow, fast, and the combined slow + fast walking situations are shown in Figures 6(a), 6(b), and 6(c), respectively. It is seen in Figure 6(c) that the combined estimator based on the two speeds mostly degrades at slow speed downhill.

The results for the other subjects A and C are similar and therefore not shown for the interest of brevity. The only difference in performance was noticed for subject C who walked using sandals, but surprisingly only with degraded performance for uphill walking (RMSE = 2.3°).

4.3. Estimation of Control Error. The control error estimator was validated at different inclinations and control angles (keel angles). The results show in Figure 7 that the error has similar precision as when estimating ground inclination. Walking with a keel which has the same angle as the ground clearly centralizes the estimations around zero. The estimator also finds the right sign of the error for other walking situations, which is essential for control law (14).

5. Conclusions

An embedded measurement system for foot orthosis during gait is proposed. Strain gauge sensors were mounted on a foot orthosis in order to give information about strain in the sagittal plane. It was shown that the extracted strain signal for each gait cycle can be efficiently coded into a vector representation based on Fourier series. This vector has a fixed and lower dimension than the original strain signal. It can be used in linear regression techniques for estimation of ground slope inclination. The proposed estimation scheme uses the entire gait cycle as opposed to kinematically based estimators that need a stationary phase in order to extract inclination information from accelerometers. A drawback could be that the strain signal-based estimator needs to be tuned for each person's walking style. A particular interesting feature, though, is that a control error defined as the unloaded orthosis deviation from the vertical line can be estimated. The orthosis used here was not hinged and the ankle angle could not be changed. A keel was used to artificially change the ankle angle to a "controlled" position. It was then verified

that the inclination estimator produced estimates of the control error. This error is thereby controllable. A future development of a hinged actuated orthosis would be able to take a advantage of the proposed technique for online ankle angle adaptation. Also, as opposed to kinematically based estimators this approach does not need to be recalibrated when changing shoes with other heel heights. Further tests on disabled are now needed for more detail evaluation.

References

[1] J. A. Blaya and H. Herr, "Adaptive control of a variable-impedance ankle-foot orthosis to assist drop-foot gait," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 12, no. 1, pp. 24–31, 2004.

[2] J. C. Moreno, F. J. Brunetti, J. L. Pons, J. M. Baydal, and R. Barberà, "Rationale for multiple compensation of muscle weakness walking with a wearable robotic orthosis," in *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1914–1919, Barcelona, Spain, April 2005.

[3] A. B. Zoss, H. Kazerooni, and A. Chu, "Biomechanical design of the berkeley lower extremity exoskeleton (bleex)," *IEEE/ASME Transactions on Mechatronics*, vol. 11, no. 2, pp. 128–138, 2006.

[4] A. H. Hansen, D. S. Childress, and S. C. Miff, "Roll-over characteristics of human walking on inclined surfaces," *Human Movement Science*, vol. 23, no. 6, pp. 807–821, 2004.

[5] A. Leroux, J. Fung, and H. Barbeau, "Postural adaptation to walking on inclined surfaces: I. Normal strategies," *Gait & Posture*, vol. 15, no. 1, pp. 64–74, 2002.

[6] M. S. Redfern and J. DiPasquale, "Biomechanics of descending ramps," *Gait & Posture*, vol. 6, no. 2, pp. 119–125, 1997.

[7] M. Kuster, S. Sakurai, and G. A. Wood, "Kinematic and kinetic comparison of downhill and level walking," *Clinical Biomechanics*, vol. 10, no. 2, pp. 79–84, 1995.

[8] A. S. McIntosh, K. T. Beatty, L. N. Dwan, and D. R. Vickers, "Gait dynamics on an inclined walkway," *Journal of Biomechanics*, vol. 39, no. 13, pp. 2491–2502, 2006.

[9] B. T. Smith, D. J. Coiro, R. Finson, R. R. Betz, and J. McCarthy, "Evaluation of force-sensing resistors for gait event detection to trigger electrical stimulation to improve walking in the child with cerebral palsy," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, no. 1, pp. 22–29, 2002.

[10] M. M. Skelly and H. J. Chizeck, "Real-time gait event detection for paraplegic FES walking," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 9, no. 1, pp. 59–68, 2001.

[11] K. Aminian, B. Najafi, C. Büla, P.-F. Leyvraz, and P. Robert, "Spatio-temporal parameters of gait measured by an ambulatory system using miniature gyroscopes," *Journal of Biomechanics*, vol. 35, no. 5, pp. 689–699, 2002.

[12] I. P. I. Pappas, M. R. Popovic, T. Keller, V. Dietz, and M. Morari, "A reliable gait phase detection system," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 9, no. 2, pp. 113–125, 2001.

[13] R. Williamson and B. J. Andrews, "Gait event detection for FES using accelerometers and supervised machine learning," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 3, pp. 312–319, 2000.

[14] A. T. M. Willemsen, F. Bloemhof, and H. B. K. Boom, "Automatic stance-swing phase detection from accelerometer data for peroneal nerve stimulation," *IEEE Transactions on Biomedical Engineering*, vol. 37, no. 12, pp. 1201–1208, 1990.

[15] A. M. Sabatini, C. Martelloni, S. Scapellato, and F. Cavallo, "Assessment of walking features from foot inertial sensing," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 3, pp. 486–494, 2005.

[16] S. K. Ng and H. J. Chizeck, "Fuzzy model identification for classification of gait events in paraplegics," *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 4, pp. 536–544, 1997.

[17] K. Aminian, P. Robert, E. Jéquier, and Y. Schutz, "Estimation of speed and incline of walking using neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. 44, no. 3, pp. 743–746, 1995.

[18] N. Wang, E. Ambikairajah, N. H. Lowell, and B. G. Celler, "Accelerometry based classfication of walking patterns using time-frequency analysis," in *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Lyon, France, August 2007.

[19] W. Svensson and U. Holmberg, "Ground angle estimator from an ankle foot orthosis based on strain sensing and Fourier series," in *Proceedings of IEEE International Conference on Mechatronics and Automation (ICMA '08)*, pp. 203–206, Takamatsu, Japan, August 2008.

[20] W. Svensson and U. Holmberg, "An autonomous control system for prosthetic foot ankle," in *Proceedings of the 4th IFAC Symposium on Mechatronic Systems*, pp. 856–861, Heidelberg, Germany, September 2006.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 581840, 10 pages
doi:10.1155/2010/581840

Research Article

Prediction Control for Brachytherapy Robotic System

Ivan Buzurovic, Tarun K. Podder, and Yan Yu

Medical Physics Division, Department of Radiation Oncology, Thomas Jefferson University, Philadelphia, PA 19107, USA

Correspondence should be addressed to Ivan Buzurovic, buzurovic@ieee.org

Received 2 November 2009; Accepted 8 March 2010

Academic Editor: Noriyasu Homma

In contemporary brachytherapy procedure, needle placement at desired location is challenging due to a variety of reasons. We have designed and fabricated an image-guided robot-assisted brachytherapy system to improve the needle placement and seed delivery. In this article we have used two different predictive control strategies in order to investigate the needle insertion efficacy and system dynamics during prostate brachytherapy. First, we used neural network predictive control (NNPC) to predict an insertion force. The NNPC uses the linearized state-space model of the robotic system to predict future system performances. Second, we used feedforward model predictive control (MPC) which allows the controller to compensate the influence of a measured disturbance's impact immediately rather than waiting until the effect appears in the system. Feedback control problem for the contact force regulation is considered. The simulation results and experiments for both cases are presented and compared.

1. Introduction

LOW-DOSE rate (LDR) prostate brachytherapy is a method of delivering radiotherapy by implanting radioactive sources into and around the prostate gland [1]. This procedure is performed under the guidance of transrectal ultrasound (TRUS) images. In traditional brachytherapy procedures, the needles are inserted transperineally under the guidance of transrectal ultrasound images. Both the needle and the ultrasound are operated manually. The seeds are deposited using a manual applicator. In order to increase accuracy of the needle placement and seed delivery, we have developed a fully automated robotic system for prostate brachytherapy [2]. Using the robotic approach we are able to record the needle insertion forces and motion trajectories measured during the actual brachytherapy needle insertion while implanting radioactive seeds in the prostate gland [3, 4]. These insertion forces are significantly responsible for needle deviation from the desired trajectories and target movements. Also, needle insertion causes soft tissues to deform and it is often difficult to obtain precise imaging data during insertion. Since the target organ is deformable, sometimes it is necessary for the next needle trajectory to be updated according to the magnitude of the deformation. During brachytherapy needle insertion forces deform the prostate tissue and displace the targeted seed positions [5]. The prostate deformation and possible calcification within the gland can cause additional needle deflection. Several authors discussed these problems and suggested methods and procedures to improve the needle insertion efficacy.

In order to reduce tissue deformation, the authors have studied the effect of different trajectories for a 2-DOF robot performing needle insertion in soft tissue [6]. They have compared tissue deformation and infinitesimal force per tissue displacement for different trajectories. In the articles [7, 8] the needle insertion motion planning system is presented. The system is based on an interactive simulation of needle insertion in deformable tissues and numerical optimization to reduce placement error. The authors described a 2D physically based, dynamic simulation of needle insertion that uses a finite-element model of deformable soft tissues and models needle cutting and frictional forces along the needle shaft. The investigation of flexible needle insertion was presented in [9]. A method for an interactive virtual needle insertion simulation has been suggested in [10]. In [11, 12] a force model for needle insertion and a model that predicted the soft tissue deformation caused by needle insertion were presented. A novel method for the evaluation and correction of brachytherapy needle deflection was introduced in [13]. The authors discussed about the influence of needle rotation

to the insertion force and needle deformation. It has been reported that robot used in conjunction with an optimized needle insertion technique benefits patients with increased accuracy, leading to a more successful outcome and reduced complications [14]. To improve needle insertion outcome, the needle insertion point, needle heading, and needle depth are optimized by minimizing the distance between a rigid needle and a number of targets in the tissue as it is shown in [15]. In [16] the authors suggested fast needle insertion to minimize tissue deformation and damage.

Generally, to improve needle placement and seed deposition in brachytherapy procedure several methods have been presented in the literature, such as parameter optimization, different needle rotation techniques, robotic insertion, force modeling, and needle steering techniques. In the previous studies, we concluded that the proper selection of the translational and rotational velocities may reduce tissue deformation and target movements by reducing insertion force [17–19]. Therefore, it can be concluded that the insertion force has a dominant influence on the needle insertion, seed deposition precision, and dosimetry distribution in brachytherapy procedure.

In our initial work [20] we have investigated tracking problem for the same robotic system using neural network approach. The force prediction model was introduced as well. In this article we have introduced novel methods to control the brachytherapy robot with the lowest tissue deformation and highest seed precision delivery as the ultimate goals. In order to achieve the desired dynamic behavior of the robotic system, both control strategies that we implemented in the robotic system have a force prediction module. For the first one we have used an artificial neural network (ANN) and neural network predictive control (NNPC). The second one was a feedforward model predictive control (MPC). The purpose of both approaches is to control the reactive force which is responsible for tissue displacement. When the reactive force is minimized, tissue displacement decreases. The force prediction control was also used to minimize the effect of system time-delay. The mathematical model of the system should include the contact force. Consequently, the purpose of both NNPC for force prediction and feedforward MPC is to optimize insertion force which should result in less tissue displacement and deformation, on one side, and higher seed deposition precision, on the other side. The experiments and results for both control approaches are discussed in the following parts.

In the first approach, we used an NNPC to predict insertion force in order to achieve real-time adaptive needle control. We have tested whether the system is able to predict the insertion force. The implemented optimization algorithm then computes the control signals that optimize the future system performances. It is shown in the literature that the force prediction technique for needle insertion can improve insertion outcome. The measurement and prediction of insertion force and needle fracture force using experimental approach were investigated in [21, 22]. Prediction of how the skin deforms upon insertion by microneedles is described in [23]. Applications of ANN in medical robotics are used in the following studies. Nonetheless, no application of the neural networks in the needle insertion optimization has been reported so far. ANN is applied in early detection of prostate cancer [24, 25]. In miniature robotic surgical systems ANN is used in conjunction with real-time visual feedback to "learn" the inverse system dynamics and control the manipulator endpoint trajectory [26]. The real-time control of a robot arm using recorded neurons in the motor cortex together with mathematical transformations including neural networks is presented in [27]. The overview of the ANN applications in many disciplines, especially in medicine can be found in [28]. In the article [29] the authors solved the problem of achieving high-accuracy positioning of a medical robot using neural network for the patient positioning system. Another robotic system for cardiac surgery which uses a recurrent neural network with adaptive internal states is described in [30]. In [31] it is shown that the neural network demonstrated robust adaptability to all of the observed breathing patterns while the linear filter failed in a significant percentage of the cases.

In the second approach (the MPC) the control strategy was to compare predicted robot states to a set of objectives and to adjust the actuators to achieve the objectives respecting the robot's constraints. Such constraints included the actuators' physical limits, needle deflection, tissue deformation and displacement, and the needle contact forces. Therefore, the robotic system is modeled to take the contact with an environment into account. For a described case, robot dynamics is usually called constrained system dynamics. Mathematical models of robotic systems give rise to a mathematical system of differential equations and algebraic equations that can be viewed as a singular system of differential equations [32]. While the manipulator end-effector is in contact with the environment, system dynamics changes significantly. Notwithstanding, it is essential to maintain an appropriate control over the working regime not only force and torques arising due to contact, but also the position and orientation of the end-effector [33]. In this approach, the controller predicts how much each output will deviate from its setpoint within the prediction horizon. It multiplies each deviation by the output's weight and computes the weighted sum of squared deviations. The critical step for MPC is choosing the weights. Because of the contact force influence the controller cannot be exclusively focused on setpoint tracking. In that case, the large manipulated-variable adjustment cannot be implemented. Contact forces are capable to accelerate equipment wear or can lead to control system instability. The described problem can be solved by decomposition of the system to its slow and fast subsystems.

2. System Description

We developed a 16- (DOF-) robot-assisted brachytherapy system, [2]. This robotic system is divided into three subsystems: the cart, the supporting platform, and the surgery module, with 3, 6, and 7 DOF, respectively, Figure 1. The supporting platform connects the surgery module to the cart. The surgery module consists of a 2-DOF ultrasound probe

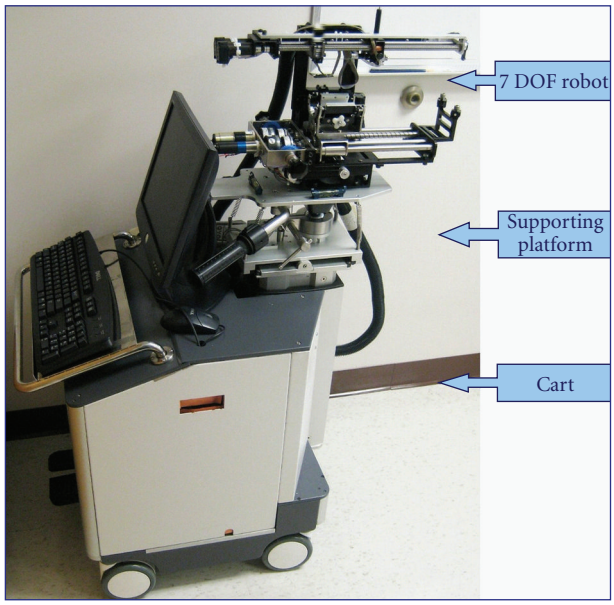


Figure 1: Robot-assisted brachytherapy system (Euclidian). Three basic subsystems are shown: 7DOF robot subsystem (surgery module), supporting platform and cart.

driver and a 5-DOF needling module. The ultrasound (US) module can be translated and rotated independently by two DC servomotors fitted with high-resolution optical encoders and gearboxes. In this current study, we have investigated the 5-DOF-needling module which consists of a gantry and a needle driver. The gantry connects the needle-driving module to the positioning platform. The gantry has two translation motions and one rotational motion (pitching). The needle driver subsystem consists of a hollow needle (cannula) and a solid needle (stylet) driven separately by two DC servomotors. The cannula rotates continuously or partially using another tiny DC motor. The basic task of these parts is to deliver the exact prescribed dose of radioactive seeds with high precision level into the human prostate. The seeds are delivered through the cannula. During the operation the stylet pushes the seeds through the cannula according to the control algorithm and prescribed surgery plan.

Also, the system is designed to take ultrasound images during the operation to update the real-time radiation dose distribution, seed position, and number of needles to be inserted into the prostate, depending on the surgery plan. Dedicated software for 3D imaging and control is developed to support the surgery procedure.

3. Mathematical Framework

3.1. System Dynamics for NNCP. Prior to implementing one of the suggested controllers to the system, it is essential to investigate the system behavior. Therefore, in this section, we have presented a mathematical model of the robotic system together with the mathematical model of the actuators. The derivation of dynamic equations is performed using

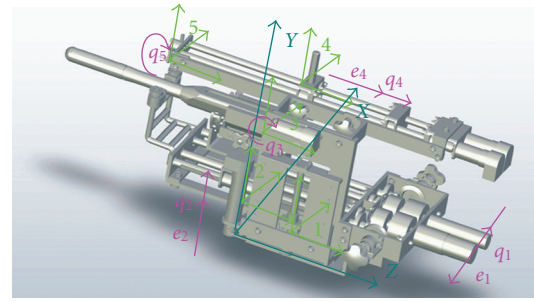


Figure 2: 5-DOF surgery module, absolute and local coordinate systems together with generalized coordinates q_i.

the energy-based Lagrangian dynamic formulation. The mechanical part of the robotic subsystem is represented as a five-link open kinematic chain. All links are rigid bodies. We assumed that all materials for the system are homogenous. The generalized coordinates $(q_1, q_2, q_3, q_4, q_5)$ are adopted to describe each joint movement; see Figure 2. The general form of dynamic equation can be written as follows:

$$\sum_{\alpha=1}^{n} a_{\alpha\gamma}(q)\ddot{q}^{\alpha} + \sum_{\alpha=1}^{n}\sum_{\beta=1}^{n} \Gamma_{\alpha\beta,\gamma}(q)\dot{q}^{\alpha}\dot{q}^{\beta} = Q_{\gamma}^{g} + Q_{\gamma}^{u}, \quad (1)$$

where $\gamma = 1, \ldots, 5$, $a_{\alpha\gamma(q)}$ stands for the metrics tensor coefficients, $\Gamma_{\alpha\beta,\gamma(q)}$ denotes the corresponding coefficients, and Q_g, Q_u are the generalized forces components. The former is a component due to gravitation and the latter is a component that corresponds to the actuator forces and torques. Each robot joint is motorized by a direct current motor, depending on the torque. Mathematical relationship between the output torque and the input motor torque for each motor is

$$N_V N_m J_M \ddot{\theta} + F\dot{\theta} + M = C_n N_m I_R. \quad (2)$$

Equation of the electrical equilibrium in the rotor circuit is:

$$L_R \dot{I}_R + R_R I_R + C_E N_V \dot{\theta} = U, \quad (3)$$

where θ is the output rotation angle, M is the output torque of the motor, I_R is the rotor current, L_R is the motor's inductance, and U is the motor's voltage. Other values are constant numbers and they depend on the motor type and motor characteristics. They are: N_V—output reduction ratio (ratio of the angular velocity at the output to the rotation speed at the input of the reductor), N_m—torque reduction ratio (ratio of the torques at the input and output of the reductor), J_M—rotor torque, F—friction coefficient of the motor, C_n—motor's mechanical constant (ratio of the motor shaft torque to the rotor current), R_R—electrical resistance of the rotor circuit and C_E—counter electromotive force coefficient. The electrical schema of the DC motors is represented in Figure 3. Based on (2) and (3) the mathematical model of the each actuator can be obtained. For the purpose of the further modeling it is necessary to express the equation of the actuator in the state space. We adopted the state vector as $\hat{x} = (\theta, \dot{\theta}, I_R)^T$. The order of the actuator's equation is

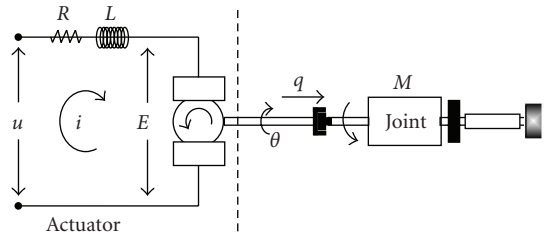


Figure 3: Electric schema for DC motor. Each robot joint is motorized by direct current motor.

three. Without losing any precision the model of the actuator can be described with an order being equal to two because the inductance is $L_R \approx 0$. In this case, the nonlinear motor behavior is neglected. A dynamic equation for the actuators is as follows:

$$\dot{\hat{x}}_i = A_i \hat{x}_i + b_i U_i + f_i M_i, \tag{4}$$

where A_i is the motor's system matrix having dimension 2×2, and b_i and f_i are 2×1 vectors. Matrices A_i, b_i, and f_i can be represented as

$$A_i = \begin{bmatrix} 0 & 1 \\ 1 & -\dfrac{1}{J_{Mi}}\left(\dfrac{F_i}{N_{Vi}N_{Mi}} + \dfrac{C_{Mi}C_{Ei}}{R_{Ri}}\right) \end{bmatrix},$$

$$b_i = \begin{bmatrix} 0 \\ \dfrac{C_{Mi}}{N_{Vi}J_{Mi}R_{Ri}} \end{bmatrix}, \tag{5}$$

$$f_i = \begin{bmatrix} 0 \\ -\dfrac{1}{N_{Vi}J_{Mi}N_{mi}} \end{bmatrix}.$$

The robotic system consists of the mechanical part (1) and actuators (4). The relationship between the actuator and manipulator's mathematical model is twofold, via coordinates and via torques: $\theta_i = t_i q_i$, where t_i is a transfer coefficient. For the described case, $t_i = 1$. Similarly, the actuator's torque is equal to the torque on the joint $M_i = \tau_i$. Equation (1) can be rewritten as follows:

$$M(q)\ddot{q} + G(q, \dot{q}) = \tau, \tag{6}$$

where M is the inertia matrix, G is a matrix that represents the centrifugal, coriollis, and gravitational influence to the system, and τ is the generalized torque vector. The state space vector of the whole system is adopted as

$$\begin{aligned} x &= [x_1 \quad x_2 \quad \cdots \quad x_{2n-1} \quad x_{2n}]^T \\ &= [q_1 \quad \dot{q}_1 \quad \cdots \quad q_n \quad \dot{q}_n]^T, \end{aligned} \tag{7}$$

where n is the number of generalized coordinates, that is, number of DOF of the surgery module. Now, based on the equation $\theta_i = t_i q_i$, it can be calculated for each actuator $\dot{q}_i = T_i \hat{x}_i$ and for the whole system $\dot{q} = Tx$, keeping in mind

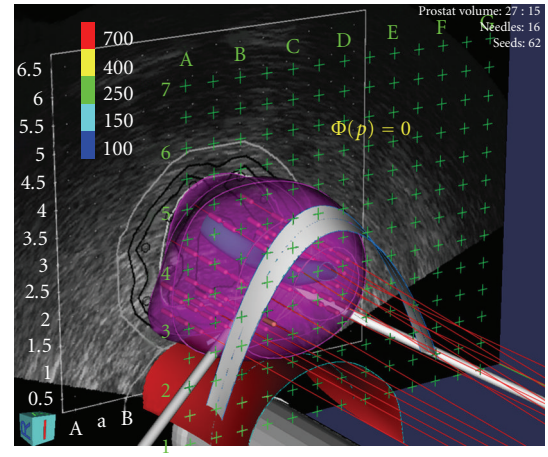


Figure 4: 3D representation of the contact surface (prostate gland) which is variable and it is different for each case. Mathematical equation of contact surface is calculated using Euclidian robotic system software.

that $T = \text{diag}(T_i)$, $T_i = (0\, t_i)$. Based on (6) the mathematical equation of the mechanical part in the state space form is

$$\tau = M(x)T\dot{x} + G(x). \tag{8}$$

Using the suggested relationships between the actuators and mechanical part via the coordinates and torques, the system model can be expressed by the following matrix equation:

$$\dot{x} = Ax + Bu + F\tau, \tag{9}$$

where $A = \text{diag}(A_i)$ having order $2n$, and B and F are $2n \times n$ matrices described by the equations $B = \text{diag}(b_i)$ and $F = \text{diag}(f_i)$. Vector u is the input vector $u = (U_1 \cdots U_n)^T$. Combining (6) and (8) and solving it in terms of τ the following expression is obtained:

$$\tau = (I_n - M(x)TF)^{-1}[M(x)T(Ax + Bu) + G(x)]. \tag{10}$$

The final form of the state space nonlinear dynamic equation is calculated by putting (10) into (9):

$$\dot{x} = \hat{A}(x) + \hat{B}(x)u, \tag{11}$$

where the matrices from (11) are represented by (12):

$$\begin{aligned} \hat{A}(x) &= \left(A + F(I_n - M(x)TF)^{-1}M(x)TA\right)x \\ &\quad + F(I_n - M(x)TF)^{-1}G(x), \\ \hat{B}(x) &= B + F(I_n - M(x)TF)^{-1}M(x)TB. \end{aligned} \tag{12}$$

3.2. System Dynamics for MPC. Assume that the contact surface is a scalar function $\Phi : R^m \to R^1$ (Figure 4):

$$\Phi(p) = 0. \tag{13}$$

The contact force vector is

$$f = D^T(p)\lambda. \tag{14}$$

λ is a scalar multiplier for the constraint function and $D(p)$ is the gradient of the constraint function, described as in (15):

$$D(p) = \frac{\partial \Phi(p)}{\partial p}. \tag{15}$$

In the previous equation $p \in \Re^3$ is a position vector from the fixed reference frame to the constrained surface. Additional constraint is $p = \Phi(q)$ at the contact point. The matrix function $J(q)$ is defined as the Jacobian matrix function:

$$J(q) = \frac{\partial \Phi(q)}{\partial q}. \tag{16}$$

The influence of the contact force to the system can be described as

$$M(q)\ddot{q} + G(q, \dot{q}) = \tau + J^T(q) f, \tag{17}$$

where $M(q)$ denotes the inertia matrix function and G is the vector function which describes coriolis, centrifugal, and gravitational effects. q is a vector of the generalized coordinates and τ is the torque vector. Influence of the external contact forces is described by factor f. Combining (13)–(17) the mathematical model of the system is

$$M(q)\ddot{q} + G(q, \dot{q}) = \tau + J^T(q) D^T(M(q))\lambda(q, \dot{q}, \tau). \tag{18}$$

After an appropriate transformation, (18) is transformed to its matrix form:

$$\begin{bmatrix} I & 0 & 0 \\ 0 & M(q) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \dot{q} \\ \ddot{q} \\ \dot{\lambda} \end{pmatrix} = \begin{bmatrix} 0 & I & 0 \\ 0 & 0 & J^T D^T \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} q \\ \dot{q} \\ \lambda \end{pmatrix} + \begin{bmatrix} 0 \\ -G(q, \dot{q}) - M(q) \\ \Phi(p) \end{bmatrix} + \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix} \tau. \tag{19}$$

System (19) is linearized at the surrounding point where the needle is inserted into the patient. The linearized mathematical model of the system is obtained, as follows:

$$E\dot{x}(t) = Ax(t) + Bu(t) + d, \tag{20}$$

with singular matrix $E \in \Re^{n \times n}$, and vector d which represents the unmeasured disturbances such as needle deflection, tissue deformation, and displacement, $A \in \Re^{n \times n}$, $B \in \Re^{m \times r}$. x and u represent the system state-space vector and control vector, respectively, $x \in \Re^n$, $u \in \Re^r$.

The system defined by (20) is known as a singular system, descriptor system, semistate system, system of differential-algebraic equations, or generalized state space system. They arise naturally in many physical applications, such as electrical networks, aircraft and robotic systems, neutral delay and large-scale systems, economics, optimization problem, and constrained mechanics. The matrices of linearized system (20) are defined as follows:

$$A = \begin{bmatrix} 0 & I & 0 \\ \dfrac{\partial}{\partial q}(G - J^T D^T \lambda)\Big|_0 & 0 & J^T D^T|_0 \\ DJ|_0 & 0 & 0 \end{bmatrix},$$
$$E = \begin{bmatrix} I & 0 & 0 \\ 0 & M(q_0) & 0 \\ 0 & 0 & 0 \end{bmatrix},$$
$$B = \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}, \tag{21}$$
$$u = \delta\tau,$$
$$d = \begin{bmatrix} 0 \\ \Delta\tau \\ 0 \end{bmatrix}.$$

Now it is possible to find subspaces S and F, as in [34], having in mind that for state space X the system (20) can be represented as the direct sum of S and F, $X = S \oplus F$. As a result of the matrix transformation $\mathbf{M}$ applied to system (20), the slow and fast subsystems can be described by a mathematical formulation:

$$\dot{x}_s = L_s x_s + B_s u + d_s,$$
$$L_f \dot{x}_f = x_f + B_f u + d_f, \tag{22}$$

with $L_s = \mathbf{M}A \mid S$, $L_f = \mathbf{M}A \mid F$, $B_s = P\mathbf{M}B$, $B_f = Q\mathbf{M}B$, $d_s = P\mathbf{M}d$, and $d_f = Q\mathbf{M}d$ for some linear transformations Q and P represented by matrices Q and P, respectively. For that case, the initial conditions are chosen to satisfy $x_{0s} = Px_0$ and $x_{0f} = Qx_0$. The solution of system (10) is $x = x_s + x_f$ as in [35]:

$$x_s = e^{L_s t} x_{0s} + \int_0^t e^{L_s(t-\tau)} B_s u(\tau) d\tau + \int_0^t e^{L_s(t-\tau)} d(\tau) d\tau,$$
$$x_f = -\sum_{i=0}^{v-1} L_f^i B_f u^i. \tag{23}$$

Now it is possible to apply the NNPC and MPC controllers and to simulate dynamical behavior of the system when two types of the predictive approaches are considered.

4. Results

4.1. Optimization Goals and Experiments. In the NNPC approach the ANN predicts insertion force. The adaptive optimization module was responsible to minimize the insertion force by recomputing the optimization parameters.

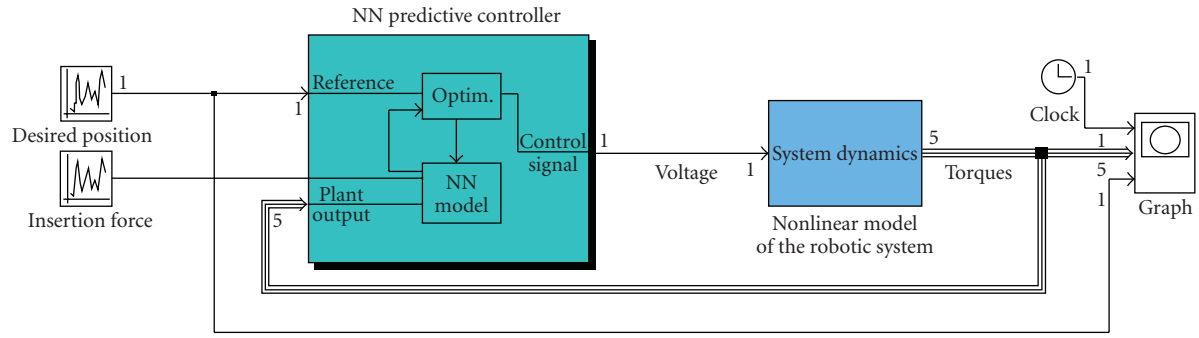


Figure 5: Predictive control system scheme: Neural network model is used to predict future plant performances. The controller calculates the control input that optimizes robotic system performances. After system identification procedure, robot model is used to predict future performances.

In the MPC approach the slow subsystem was stabilized with state feedback

$$u = K_{sl}x_s. \tag{24}$$

A stabilizing feedback K_{sl} for the slow subsystem was obtained by minimization of the following expression:

$$\min \int_0^\infty \left(x_x^T Q x_s + u^T P u\right) dt. \tag{25}$$

For data collection, we used the fully automated Robotic Brachytherapy System—EUCLIDIAN (Endo-Uro Computed Lattice for Intratumoral Delivery, Implantation, Ablation with Nanosensing); see Figure 1. The whole equipment was integrated into the system. For cannula, we used Rev. A4, 304 Stainless steel needle (r = 2.98 mm ± .01 mm with 15° bevel angle) and for stylet Rev. A4, 304 Stainless steel needle Vita Needle Company. The phantom was made of polyvinylchloride (PVC) plastic and PVC+ hardener in the ratio 80% to 20%, respectively (MF Manufacturing, TX). Force measurements were performed using two single-axis force sensors (Model 13, Honeywell Sensotech, Columbus, OH), one at each of the proximal ends of the stylet and cannula, and one 6-axis force-torque sensor (Nano17, ATI Industrial Automation, Apex, NC) at the distal end of the cannula. The deposited seeds were rounded stainless steel dummy seeds (BARD). The position of the needle tip and, consequently, the depth of deposition into the phantom were measured using optical encoders FAULHABER MicroMo HEDM5500J series 5500, attached to the cannula and stylet motors.

4.2. Force Prediction Using NNPC. Neural network controller uses a neural network model to predict future system responses to potential control signals (Figure 5). The first task of the system was to train the neural network to represent the forward dynamics of the robotic system. An optimization algorithm then computes the control signals that optimize the future system parameters. For the MPC the dynamic model of the robotic system (11) is used to predict the future system behavior. An optimization algorithm is used to select the control input that optimizes the future performances. The error between the predicted force value and measured value is used as an ANN training signal.

A multilayer backpropagation neural network is employed to the NNPC scheme to model the nonlinear relationships between the robotic system input (actuators voltage) and the insertion force, in order to adapt the system to a variety of operating conditions and to acquire a more flexible learning ability. The differences of the predicted force and real-time measured values are within the margin of the 1.8% which may be satisfactory for the presented application, as in Figure 6. The prediction error was shown in Figure 7. The time delay for such a complex nonlinear system could be modified if stronger system performaces are required. The predictive neural network had 3 hidden layers. The number of the time steps within the prediction errors that were minimized was 8. The number of the time steps within the control increments that are minimized was 2. In the presented simulation, 350 training epochs were used together with *trainlm* training function. It was found that the optimal update interval of 0.3s was good enough in the prediction procces, but our anticipation was that the suggested system performances might benefit if the future algorithms are applied in order to get a faster update rate.

In Figure 8 the insertion force for both cases was represented: nonoptimized insertion and insertion when the ANN network approach is employed. It can be concluded that for optimized case the insertion force is smooth without peaks. Using the presented method, it is not possible to completely decrease the force during needle trajectory but high force gradient can be avoided. Consequently, it is possible that the tissue deformation and damage can be decreased.

4.3. Insertion Force Control and MPC. The purpose of this approach is to control the reactive insertion force on one side and to predict and compensate the impact of the measured and unmeasured disturbances rather than waiting until the effect appears at the output of the system on the other side. Because of the fact that this procedure required an accurate robot system model, it was necessary to obtain a more precise model. That is a reason for adopting the robotic system model as a singular system of differential equations. In [33] it is shown that the impulsive behavior of the system can be

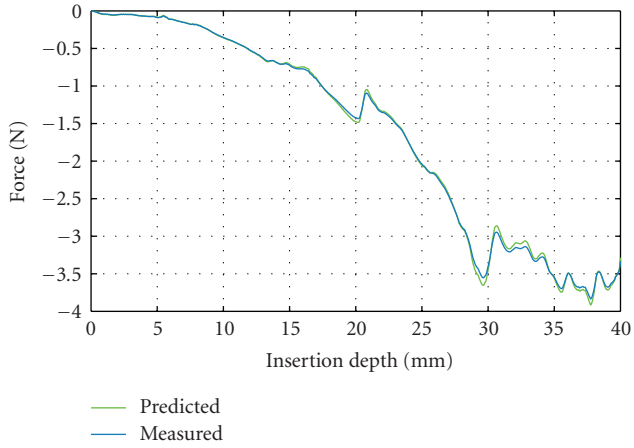


Figure 6: Insertion force prediction results using ANN for representative insertion: measured force (blue) and predicted force (green).

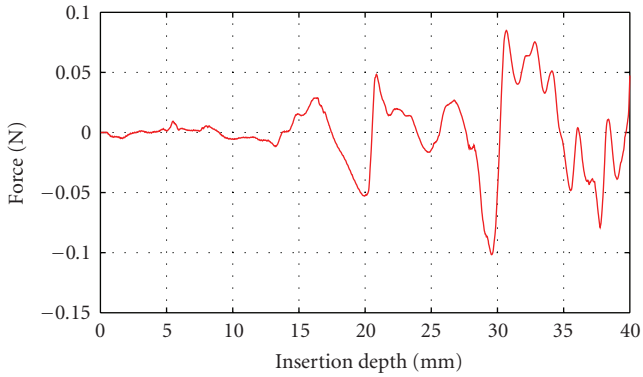


Figure 7: Insertion force prediction error.

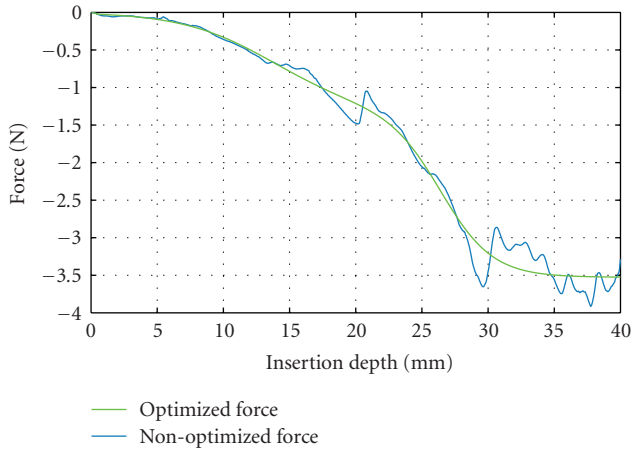


Figure 8: Optimized (green) and nonoptimized (blue) insertion forces measured by force sensor installed in the robotic system.

avoided using appropriate initial conditions, defined by $x_{0s} = Px_0$ and $x_{0f} = Qx_0$. By using the described approach the fast subsystem will not induce impulsive behavior. Moreover, it can be concluded as stated previously and from (23)

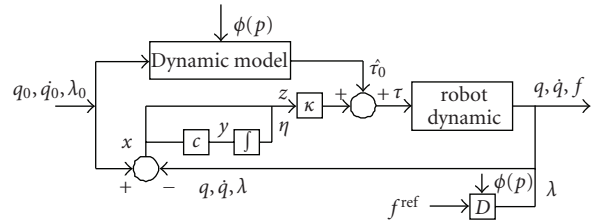


Figure 9: Insertion force control scheme for the slow subsystem. Disturbance d is given to MPC and its effect is predicted and compensated before the effect appears at the system output.

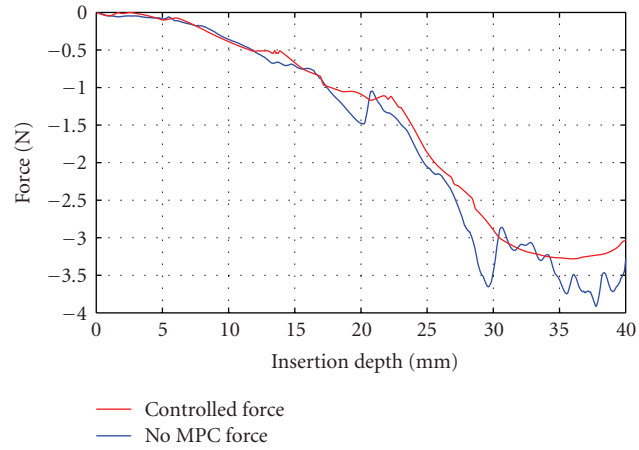
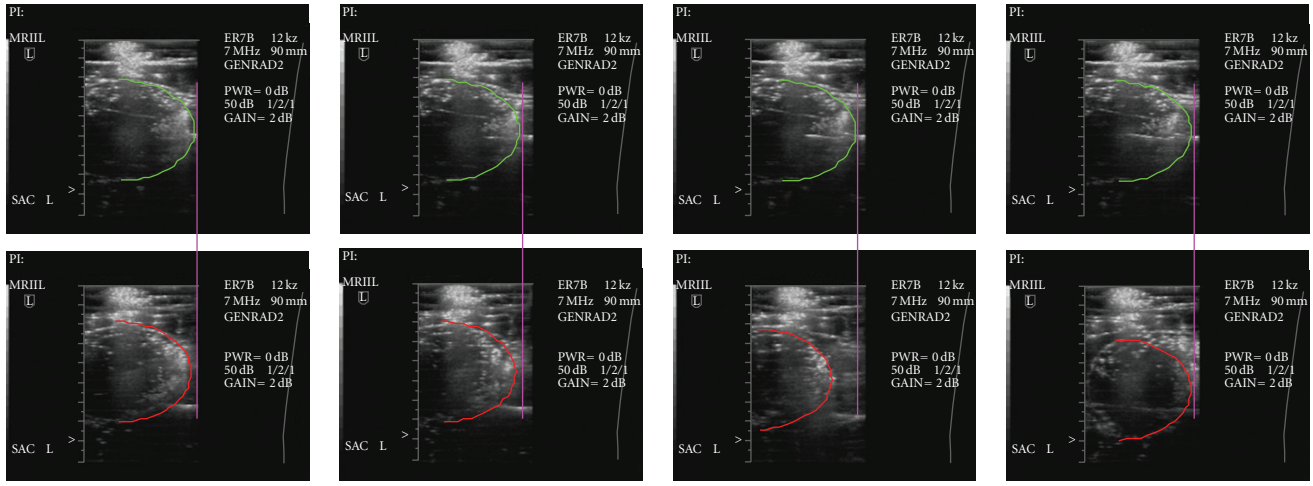


Figure 10: Controlled force (red) and insertion force when MPC is not implemented (blue).

that there is little need to find fast feedback to eliminate the impulsive behavior. The necessary task was to find an appropriate feedback for the slow subsystem. The control scheme for the slow subsystem is represented in Figure 9, as suggested in [34].

Furthermore, when the MPC controller is applied, the main objective is to hold the insertion force at the predefined acceptable reference value, by adjusting the control signal from actuators in order to minimize the prostate displacement, that is, the reactive force which acts upon the tissue. Using this approach it is possible to decrease the insertion force during insertion trajectory. The needle displacement is an unmeasured disturbance and the controller provides feedback compensation for such disturbances. For the insertion force the controller provides feedforward compensation. Various noise effects can corrupt the measurements. The noise could vary randomly with a zero mean or could exhibit a nonzero, drifting bias. The MPC uses a filtering method for removing estimated noise component. At the beginning of each sampling instant, the controller estimates the current system state. Accurate knowledge of the state improves prediction accuracy which improves controller performances. As a result of passive insertion force control, multiplier λ changes values during the insertion and the force is minimized based on (25). During the insertion passive force control becomes active and keeps passive insertion force close to the predefined minimal value. When



(a) (b) (c) (d)

Figure 11: Tissue displacements for the MPC (green) and for open-loop control approach (red). Disturbance d is given to MPC and its effect is predicted and compensated before effect appears at the system output.

the MPC control approach was implemented, it is possible to decrease the insertion force, as it is shown in Figure 10. Also, peaks during the insertion are minimized. These conclusions give us a reason to believe that tissue deformation can be decreased better than using the NNPC approach. The previous hypothesis was proven after the experiments, as in Figure 11. Figure 11 shows the result of tissue deformation for both insertion approaches, the MPC and classic insertion with open loop control. It can be observed that the open-loop controller is not able to predict insertion force and consequently to minimize tissue displacement. A system with the MPC has better optimization characteristics and faster response. Consequently, tissue deformation is less compared to the open-loop case as well as to the NNPC approach. For the needle displacement, the results are similar. As a result of the experiments, the needle displacement, and therefore seed deposition precision, for insertion depth of 8 cm in the first case (NNPC) was between 0.8 and 1.1 mm. For the MPC it was 0.2–0.4 mm. The results in the second approach were better because the MPC includes an influence on the needle deflection as an unmeasured disturbance. Presented feedback compensation method resulted in needle deflection which ensures clinically acceptable precision for radioactive seed deposition (less then ± 2 mm, according to [36]).

5. Conclusion and Future Work

In this article, we used the neural network predictive control (NNPC) to predict an insertion force. The NNPC uses the nonlinear model of the robotic system to predict future system performances. The controller then calculates the control input vector (motors voltages) that optimizes the robot performances based on predicted force values over finite time interval. The control system is designed to maintain practical stability even when the force gradient is high. It is shown that the neural network approach is suitable for improving performances of the robotic system with complex dynamic behavior.

In the second approach the MPC method was applied. This method shows better results than the NNPC method. There are two reasons for that. First is that the mathematical model of the system used for NNPC does not take the insertion force into account. The insertion force was measured but the robot dynamics does not depend directly on the insertion force. Based on the experiments, the results show that this method has a slower response, and sometimes it is not good enough for displacement compensation. Actually, the compensation of the displacement for the NNCP method is performed by decreasing force gradient. It was a consequence of the insertion force prediction and optimization of the insertion procedure. That is a second reason why MPC was more suitable for this kind of application. On the other hand, the mathematical model in the MPC case includes the insertion force. Moreover, the system is described by using a singular system of differential equations. That approach gives a more accurate mathematical model which is one of the basic prerequisites for the MPC method. An influence of the insertion force can be changed by observing changes of the Lagrange multiplier λ. Here, not only the control of the passive insertion force (the reaction force) were implemented, but also the disturbances are taken into account. All disturbances were divided into two groups, measured and unmeasured. The insertion force was a measured disturbance to the system and the needle and tissue displacement belonged to another group of disturbances. Together with the control of the passive force, position control and velocity control were applied as well.

The second approach (MPC) can include more procedure specific criteria such as needle acceleration or rotation for each specific encounter, or patent specific criteria such as age, prostate volume, activity, or even prostate density or texture. Based on the presented results an effective motion

planning algorithm will be developed as well as the improved predictive and optimization algorithm.

Acknowledgments

This study is supported by the Department of Defense (DoD) under Grant no. W81XWH-06-1-0227, and the National Cancer Institute (NCI) under Grant no. R01-CA091763.

References

[1] D. Hodgson, P. Acher, and D. Cahill, "An update on localized prostate cancer," *International Journal of Clinical Practice*, vol. 2, no. 61, pp. 315–319, 2007.

[2] Y. Yu, T. Podder, Y. Zhang, et al., "Robot-assisted prostate brachytherapy," in *Proceedings of the 9th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI '06)*, vol. 4190 of *Lecture Notes in Computer Science*, pp. 41–49, Copenhagen, Denmark, October 2006.

[3] T. Podder, D. Clark, J. Sherman, et al., "In vivo motion and force measurement of surgical needle intervention during prostate brachytherapy," *Medical Physics*, vol. 33, no. 8, pp. 2915–2922, 2006.

[4] T. K. Podder, J. Sherman, D. Fuller, et al., "In-vivo measurement of surgical needle intervention parameters: a pilot study," in *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS '06)*, pp. 3652–3655, New York, NY, USA, August-September 2006.

[5] E. Dehghan and S. E. Salcudean, "Needle insertion point and orientation optimization in non-linear tissue with application to brachytherapy," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 2267–2272, Roma, Italy, April, 2007.

[6] N. Abolhassani, R. Patel, and M. Moallem, "Trajectory generation for robotic needle insertion in soft tissue," in *Proceedings of the 26th Annual International Conference of the IEEE EMBS*, pp. 2730–2733, San Francisco, Calif, USA, September, 2004.

[7] R. Alterovitz, K. Goldberg, J. Pouliot, R. Taschereau, and I.-C. Hsu, "Sensorless planning for medical needle insertion procedures," in *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, vol. 4, pp. 3337–3343, Las Vegas, Nev, USA, October 2003.

[8] R. Alterovitz, K. Y. Goldberg, J. Pouliot, and I.-C. Hsu, "Sensorless motion planning for medical needle insertion in deformable tissues," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 2, pp. 217–225, 2009.

[9] C.-K. Chui, S.-H. Teoh, C.-J. Ong, J. H. Anderson, and I. Sakuma, "Integrative modeling of liver organ for simulation of flexible needle insertion," in *Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV '06)*, pp. 1–6, Singapore, December 2006.

[10] S. P. DiMaio and S. E. Salcudean, "Interactive simulation of needle insertion models," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 7, pp. 1167–1179, 2005.

[11] A. M. Okamura, C. Simone, and M. D. O'Leary, "Force modeling for needle insertion into soft tissue," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 10, pp. 1707–1716, 2004.

[12] J. R. Crouch, C. M. Schneider, J. Wainer, and A. M. Okamura, "A velocity-dependent model for needle insertion in soft tissue," in *Proceedings of the 8th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI '05)*, vol. 3750 of *Lecture Notes in Computer Science*, pp. 624–632, Palm Springs, Calif, USA, October 2005.

[13] G. Wan, Z. Wei, L. Gardi, D. B. Downey, and A. Fenster, "Brachytherapy needle deflection evaluation and correction," *Medical Physics*, vol. 32, no. 4, pp. 902–909, 2005.

[14] M. A. Meltsner, N. J. Ferrier, and B. R. Thomadsen, "Observations on rotating needle insertions using a brachytherapy robot," *Journal of Physics in Medicine and Biology*, vol. 52, no. 19, pp. 6027–6037, 2007.

[15] E. Dehghan and S. E. Salcudean, "Needle insertion parameter optimization for brachytherapy," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 303–315, 2009.

[16] M. Mahvash and P. E. Dupont, "Fast needle insertion to minimize tissue deformation and damage," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '09)*, pp. 3097–3102, Kobe, Japan, May 2009.

[17] I. Buzurovic, T. Podder, K. Yan, et al., "Parameter optimization for brachytherapy robotic needle insertion and seed deposition," *Medical Physics*, vol. 35, no. 6, p. 2865, 2008.

[18] T. K. Podder, D. P. Clark, D. Fuller, et al., "Effects of velocity modulation during surgical needle insertion," in *Proceedings of the 27th Annual International Conference of the Engineering in Medicine and Biology Society (EMBS '05)*, vol. 7, pp. 5766–5770, Shanghai, China, September 2005.

[19] T. K. Podder, J. Sherman, D. P. Clark, et al., "Method to reduce force and target movement during surgical needle interventions," in *Proceedings of European Medical & Biological Engineering Conference*, vol. 11, pp. 4315–4320, Prague, Czech Republic, 2005.

[20] I. Buzurovic, T. K. Podder, and Y. Yu, "Force prediction and tracking for image-guided robotic system using neural network approach," in *Proceedings of IEEE Biomedical Circuits and Systems Conference (BIOCAS '08)*, pp. 41–44, Baltimore, Md, USA, November 2008.

[21] S. P. Davis, B. J. Landis, Z. H. Adams, M. G. Allen, and M. R. Prausnitz, "Insertion of microneedles into skin: measurement and prediction of insertion force and needle fracture force," *Journal of Biomechanics*, vol. 37, no. 8, pp. 1155–1163, 2004.

[22] S. P. Davis, M. G. Allen, and M. R. Prausnitz, "The mechanics of microneedles," in *Proceedings of Annual International Conference of the IEEE Engineering in Medicine and Biology (EMBS '02)*, vol. 1, pp. 498–499, Houston, Tex, USA, October 2002.

[23] N. Roxhed, T. C. Gasser, P. Griss, G. A. Holzapfel, and G. Stemme, "Penetration-enhanced ultrasharp microneedles and prediction on skin interaction for efficient transdermal drug delivery," *Journal of Microelectromechanical Systems*, vol. 16, no. 6, pp. 1429–1440, 2007.

[24] B. Djavan, M. Remzi, A. Zlotta, C. Seitz, P. Snow, and M. Marberger, "Novel artificial neural network for early detection of prostate cancer," *Journal of Clinical Oncology*, vol. 20, no. 4, pp. 921–929, 2002.

[25] H. Zhang, Z. Zhang, and A. W. Partin, "Neural network based systems for prostate cancer stage prediction," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 3, pp. 659–662, Como, Italy, 2000.

[26] J. M. Stevens and G. D. Buckner, "Actuation and control strategies for miniature robotic surgical systems," *Journal of Dynamic Systems, Measurement and Control*, vol. 127, no. 4, pp. 537–549, 2005.

[27] J. K. Chapin, K. A. Moxon, R. S. Markowitz, and M. A. L. Nicolelis, "Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex," *Nature Neuroscience*, vol. 2, no. 7, pp. 664–670, 1999.

[28] J. E. Dayhoff and J. M. DeLeo, "Artificial neural networks: opening the black box," *Cancer*, vol. 91, supplement 8, pp. 1615–1635, 2001.

[29] L. Qian and C. Mavroidis, "Identification of the end-effector positioning errors of a high accuracy large medical robot using neural networks," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '98)*, vol. 2, pp. 951–958, Victoria, Canada, October 1998.

[30] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber, "A system for robotic heart surgery that learns to tie knots using recurrent neural networks," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, pp. 543–548, Beijing, China, October 2006.

[31] M. J. Murphy and S. Dieterich, "Comparative performance of linear and nonlinear neural networks to predict irregular breathing," *Journal of Physics in Medicine and Biology*, vol. 51, no. 22, pp. 5903–5914, 2006.

[32] N. H. McClamroch, "Singular systems of differential equations as dynamic models for constrained robot systems," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, pp. 21–28, San Francisco, Calif, USA, 1986.

[33] J. K. Mills, "Constrained manipulator dynamic models and hybrid control," in *Proceedings of the 3rd International Symposium on Intelligent Control*, vol. 1, pp. 424–429, Arlington, Va, USA, August 1988.

[34] D. Cobb, "Descriptor variable systems and optimal state regulation," *IEEE Transactions on Automatic Control*, vol. 28, no. 5, pp. 601–611, 1983.

[35] J. K. Mills and A. A. Goldenberg, "Force and position control of manipulators during constrained motion tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 1, pp. 30–46, 1989.

[36] R. Nath, L. L. Anderson, J. A. Meli, A. J. Olch, J. A. Stitt, and J. F. Williamson, "Code of practice for brachytherapy physics: report of the AAPM radiation therapy committee task group no. 56," *Medical Physics*, vol. 24, no. 10, pp. 1557–1598, 1997.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 397865, 11 pages
doi:10.1155/2010/397865

Research Article

Support Vector Machine for Behavior-Based Driver Identification System

Huihuan Qian,[1,2] Yongsheng Ou,[1] Xinyu Wu,[1] Xiaoning Meng,[2] and Yangsheng Xu[1,2]

[1] *Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China*
[2] *Department of Mechanical and Automation Engineering, Chinese University of Hong Kong, Hong Kong*

Correspondence should be addressed to Yongsheng Ou, yoo205@lehigh.edu

Received 1 November 2009; Accepted 5 March 2010

Academic Editor: Zeng-Guang Hou

We present an intelligent driver identification system to handle vehicle theft based on modeling dynamic human behaviors. We propose to recognize illegitimate drivers through their driving behaviors. Since human driving behaviors belong to a dynamic biometrical feature which is complex and difficult to imitate compared with static features such as passwords and fingerprints, we find that this novel idea of utilizing human dynamic features for enhanced security application is more effective. In this paper, we first describe our experimental platform for collecting and modeling human driving behaviors. Then we compare fast Fourier transform (FFT), principal component analysis (PCA), and independent component analysis (ICA) for data preprocessing. Using machine learning method of support vector machine (SVM), we derive the individual driving behavior model and we then demonstrate the procedure for recognizing different drivers by analyzing the corresponding models. The experimental results of learning algorithms and evaluation are described.

1. Introduction

Automobiles are by now indispensable to our personal lives, as well as to the activities of business, public services, and even national security, but the problem of car thefts is a reality and it threatens the automobile security seriously. According to the National Insurance Crime Bureau, a vehicle is stolen every 25 seconds in the U.S.A. Each year, over 1.2 million vehicles are stolen across the country, causing the loss of 8 billion US dollars. Therefore, the work on vehicle security is significant.

Not surprisingly, many classical intelligent technologies are already well established within the automotive industry for vehicle security. GM has developed the *OnStar* system [1], which can supply the vehicle information to the infrastructure once the vehicle owner reported for a theft. However, it has no solution to how to detect the theft by vehicle automatically. There are a large number of vehicle companies which have commercialized their intelligent electronic car keys [2], however, the level of "intelligence" is only limited to activating the vehicle remotely. The deeper research for intelligent recognition of a vehicle theft will boost the vehicle intelligence significantly.

In the past decade, a number of groups have participated in the research of intelligent vehicles, which have led to projects including ARGO [3], ROVA [4], and MOSFET [5]. These vehicles mainly apply machine vision techniques to perform road-condition analysis and assist people in driving.

In recent years, the paradigm of learning human behaviors has attracted considerable amount of attention. It is difficult to describe the desired instructions into specific and proper code statements. In the past decade, several researchers have proposed various experimental designs and applications [6–8]. Significant research towards learning skills directly from humans on intelligent vehicles has been conducted primarily by the Navlab group at the CMU [9] and our group at the CUHK [10, 11].

Support vector machine (SVM) has recently become popular in the machine learning. SVM is a new learning-by-example paradigm spanning a broad range of classification, regression, and density estimation problems. This systematic approach motivated by statistical learning theory combines ideas from various scientific branches such as mathematical programming, exploiting the quadratic programming for convex optimization, functional analysis,

indicating adequate methods for kernel representations, and machine learning theory, exploring the large maximum classifiers concept [12]. It was first introduced by Vapnik and co-workers and is described in more detail in [13, 14]. The roots of this approach, the so-called support vector (SV) methods of constructing the optimal separating hyperplane for pattern recognition, were already presented and had been used in machine learning in [15]. The SV technique was generalized for nonlinear separating surfaces in [16], and was further extended for constructing decision rules in the nonseparable case [17]. The training task involves optimization of a convex cost function conveying to a technique without local minima.

SVM has been applied to many areas, such as pattern recognition, regression, equalization [18, 19]. It is adopted in applications such as dynamic robot control [20], space robot control [21], image classification [22], human dynamic gait recognition [23], and so on.

In this paper, we focus on the research of utilizing dynamic human behavior models for vehicle security (preventing vehicles from being stolen) application. A methodology of modeling dynamic human behaviors is proposed. By learning from driving performances, the intelligent classifier can be embedded into an IC-based car key, through which the vehicle security system can identify valid drivers based on the ways the vehicle are driven and the drivers behave. When an illegitimate driver come to use the car and the demonstrated driving behaviors do not match the specified model, the car will be enabled to automatically stop running and deliver alarm signals accordingly. In [24, 25], we constructed a steering-by-wire vehicle and its steering interphase, which will be able to work as an on-road test platform for vehicle security.

We highlight the following aspects of our system in this paper: First, live biometrical features in dynamic human behaviors are adopted in the system, which brings the enhanced security to the proposed security system. Second, since we collect the signals directly from human driving controls, which include steering, acceleration and braking, we do not utilize other car dynamics and environmental variables such as the car's yaw angle with respect to the road, lateral offset to the road's center, and the road curvature. Therefore, no complicated sensor is required, which brings to the system robust and efficient performance in realtime. Third, the intelligent security system is easy to install on a normal vehicle by adding on functional modules. No complicated requirements means little space and time needed for system installation and drivers are likewise not distracted by the addition of the in-car system. Finally, we develop a methodology to capture and analyze the characteristics of human behaviors into computational representations. It is easily scalable for other applications.

The paper is organized as follows. Firstly, technical descriptions of the experimental hardware and software platforms are presented. In Section 3, we then extract features from the data collected from the experimental platform. Thereafter, we utilize SVM to classify the features of human driving behavior data in Section 4. Section 5 studies and analyzes the experimental results. Section 6 concludes the paper.

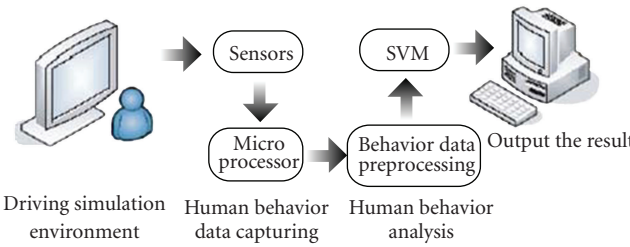


Figure 1: Architecture of the experimental system.

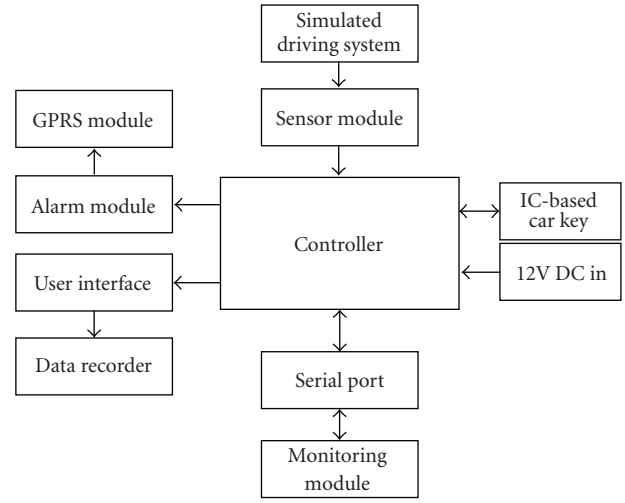


Figure 2: Architecture of the system hardware implementation.

2. Experimental Platform and Data Collection

2.1. Overview. In this section, the technical descriptions of the implemented hardware and software platforms are presented. Figure 1 shows the architecture of the experimental intelligent vehicle security system. We design an experimental platform which consists of three parts, (1) a real-time graphic simulator offering full controls including steering as well as the brake and acceleration pedals, (2) a sensory system with a processor circuit board to capture human driving behavior data, and (3) an analysis system to model and identify human behavior. While the human subject is "driving" on the driving simulator in part 1, the signals of his/her driving performance are captured by part 2 and then transferred to part 3 for modeling processes.

Figure 2 shows the hardware architecture of the proposed intelligent vehicle security system. The sensory module is implemented in the vehicle (we use simulated driving system instead) to measure the values of steering, braking, and acceleration. The data is sent to the monitoring module in realtime with the driver models embedded in the IC-based car key. Then the recognition result from the monitoring module is sent to user interface incar and to the driving data recorder as well. If the illegitimate driver is detected,

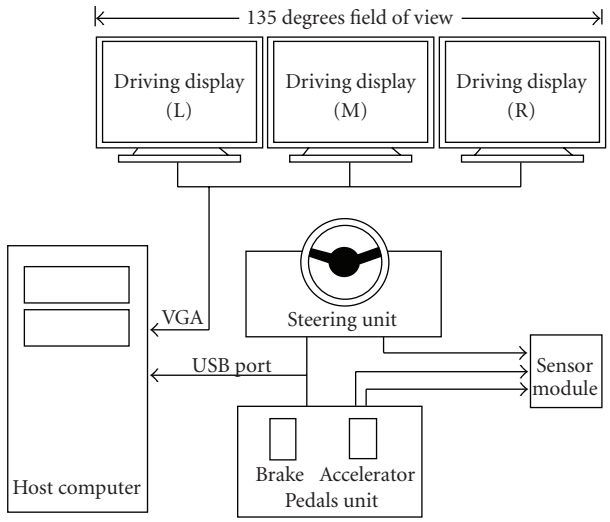


Figure 3: Diagram of the simulated driving system.

the alarm module will be active and the alarm message can be delivered through the wireless communication network.

2.2. Driving Simulation Subsystem. Figure 3 shows the diagram of proposed driving simulation subsystem. In this system, a simulated driving environment which bears substantial resemblance to a comparable real driving task is developed. Although some aspects of a real driving task cannot be modeled well in a simulated driving environment, including driving control reality variable road conditions, and, we choose a simulated driving task because it embodies the qualities which meet that criteria for comparing and identifying individual driving behavior. Moreover, the focus of this paper is the analysis of human behaviors itself.

In the simulation subsystem, we apply one PC to offer the simulated driving environment for the driver, including rendered 3-D graphics display as well as realistic controls, which are the steering wheel and pedals for acceleration and brake. We adopt a set of commercial racing game controllers from Logitech (in Figures 4(a) and 4(b)) and the racing game named "Need for Speed: Underground" from Electronic Arts. The racing game provides the features of 3-D road scenes with dynamic traffic stream (in Figure 4(c)). Human subject uses the front view and game controller to drive in the simulated environment, as if sitting in the driving seat in a real car. In this experiment, the host PC used to run the racing game is based on Pentium 4 2.0 G and 1024 MB RAM, the graphics card is Matrox Parhelia 128 MB which supports 3 monitors at a time for wide angle display.

2.3. Data Sensing and Capturing Subsystem. In the data capturing subsystem, a processor circuit board (in Figure 4(d)) is utilized to sense and gather the individual driving behavior data from the driving environment simulation subsystem. With the driving control sensing device, 3 analog signals are gathered and then processed by an A/D converter at the sampling time of 100 ms and then the digitized values are sent to the microcomputer (ATmega8535L). The received data $X(t)$ can be represented by

$$X(t) \triangleq \{a(t), b(t), c(t)\}, \tag{1}$$

where $a(t)$ represents the normalized acceleration value, $b(t)$ represents the normalized braking value, and $c(t)$ represents the normalized steering value.

The $X(t)$ is a time sequence of 3-Dimensional data (shown in Figure 5) and is transferred to the human behavior analysis subsystem through the RS232 port for further processing.

2.4. Data Analysis Subsystem. In the human behavior analysis subsystem, the methods introduced in the following sections are applied to the retrieved human behavior data. For our goal in identifying the drivers from their driving skills, the human behavior model library of each driver is generated from the corresponding behavior data input. Once the models are ready, we implement them as the classifier in the system in response to the real-time individual driving performance.

Before modeling human driving behaviors, we apply data preprocessing methods towards data collected from the previous subsystem. Fast Fourier transform (FFT), principal component analysis (PCA), and independent component analysis (ICA) are investigated in this paper. The output of this data preprocessing module is for the support vector machine (SVM) modeling and evaluation. These methodologies are presented as follows.

3. Feature Extraction

In this section, we apply data preprocessing methods towards data collected from the aforementioned experimental platform. It is necessary and important to apply data reduction and feature selection in data preprocessing for human behaviors modeling because failures in feature selection reduces the efficiency of the system performance significantly, even bad feature selection causes the failure of whole recognition procedures. Among several feature extraction methods, fast Fourier transform (FFT), principal component analysis (PCA), and independent component analysis (ICA) are investigated in this paper.

3.1. Fast Fourier Transform. To determine the extent of preprocessing human behaviors, we consider factors such as the existence of a preprocessing algorithm, its necessity, its complexity, and its generality. We select fast Fourier transform. In fact, if we have a function given by $x(t)$ whose Fourier transform is $X(f)$, when we shift $x(t)$ by a constant time, T, that is, $x(t-T)$, its Fourier transformation is

$$X(f)e^{-j2\pi fT}, \tag{2}$$

that is, time shifting affects phase only; the magnitude remains constant throughout.

Although the Fourier transform does not explicitly show the time localization of frequency components, the time

(a) (b)

(c) (d)

Figure 4: (a) Steering wheel; (b) Acceleration and brake pedals; (c) Simulated driving scene; (d) Processor circuit board.

localization can be presented by suitably prewindowing the signal in time domain [26]. Accordingly, short time Fourier transform (STFT) [27] of a signal $x(t)$ is defined as

$$\mathrm{STFT}_x^{\gamma}(t,f) = \int_{\tau} [x(\tau)\gamma(\tau - t)]e^{-j2\pi f\tau}d\tau. \quad (3)$$

STFT at time t is the Fourier transform of the signal $x(\tau)$ multiplied by a shifted analysis window $\gamma(\tau - t)$ centered around t. All integrals are from $-\infty$ to ∞. Because multiplication by the relatively short window $\gamma(\tau - t)$ effectively suppresses the signal outside a neighborhood around the analysis time point $t = \tau$, the STFT is simply a local spectrum of the signal $x(\tau)$ around analysis window t. The windows can be overlapped to prevent loss of information. Although human behavior is a nonstationary stochastic process over a long interval, it can be considered stationary over a short time interval. Thus, STFT should give a good spectral representation of the human behaviors during that time interval.

3.2. Principal Component Analysis. The method can be described in brief as follows suppose that we have two sets of training samples: A and B. The number of training samples in each set is N. Φ_i represents each eigenvector produced by principal component analysis (PCA). Each of the training samples, including positive samples and negative samples, can be projected into an axis extended by the corresponding eigenvector. By analyzing the distribution of the projected $2N$ points, we can roughly select the eigenvectors which have more human behavior information. The following is a detailed description of the process.

(1) For a certain eigenvector Φ_i, compute its mapping result according to the two sets of training samples. The result can be described as $\lambda_{i,j}$ $(1 \le i \le M, 1 \le j \le 2N)$.

(2) Train a classifier f_i using a simple method such as Perception or Neural Network which can separate $\lambda_{i,j}$ into two groups: specific valid driver or others with a minimum error $E(f_i)$.

(3) If $E(f_i) < \theta$, then we delete this eigenvector from the original set of eigenvectors.

M is the number of eigenvectors and $2N$ is the total number of training samples. θ is the predefined threshold. The remaining eigenvectors are selected.

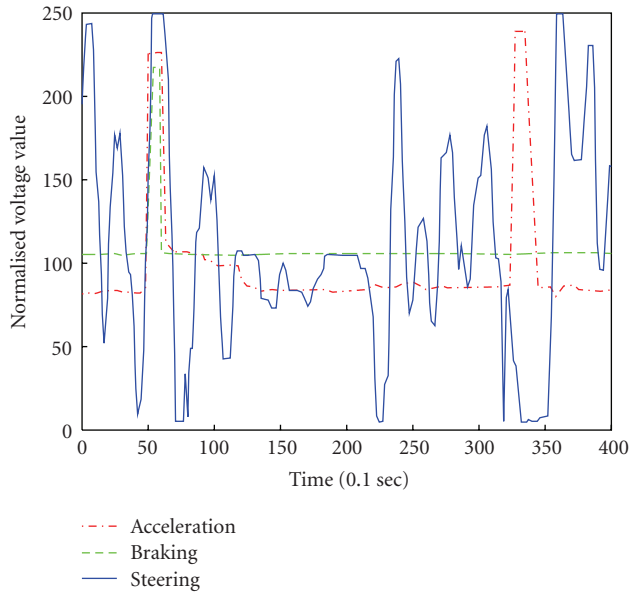


Figure 5: Human driving behavior data.

3.3. Independent Component Analysis. Apart from PCA, we also propose using independent component analysis (ICA) to reduce the dimensions of the data inputs for human behavior modeling. Independent component analysis is a statistical method which transforms an observed multidimensional vector into components that are statistically as independent as possible.

A fixed-point algorithm is employed for independent component analysis [28]. The goal of the ICA algorithm is to search for a linear combination of the prewhitened data $x_i'(t)$, where $y = w^T x'(t)$, such that the negentropy (non-gaussianity) is maximized. w is assumed to be bounded to have a norm of 1 and g' is the derivative of g. The fixed point algorithm [28] is as follows:

(1) Generate an initial random vector $w_{k-1}, k = 1$

(2) $w_k = E\{x' g(w_{k-1}^T x')\} - E\{g'(w_{k-1}^T x')\} w_{k-1}$

(3) $w_k = w_k / \|w_k\|$

(4) Stop if converged ($\|w_k - w_{k-1}\|$ is smaller than a certain defined threshold). Otherwise, increment k by 1 and return to step (2).

If the process converges successfully, the vector w_k produced can be converted to one of the underlying independent components by $w_k^T x'(t)$, $t = 1, 2, \ldots, m$. Due to the whitening process, the columns of B are orthonormal. By projecting the current estimates on the subspace orthogonal to the columns of the matrix B which are found previously, we are able to retrieve the components one after the other.

4. Learning via Support Vector Machine

In this paper, SVM is applied within the framework of modeling human behaviors for intelligent vehicle security application. Inherent complexities and the nonlinearity of human dynamic behavior make mathematical modeling difficult, hindering the use of conventional methods for process modeling and condition monitoring.

4.1. Mathematical Description. In SVM, the basic idea is to map the data X into a high-dimensional feature space $\mathcal{F}$ via a nonlinear mapping Φ, and to do linear classification or regression in this space

$$\mathrm{f}(\bar{x}) = (\omega \cdot \Phi(\bar{x})) + b \quad \text{with } \Phi : R^N \longrightarrow \mathcal{F}, \omega \in \mathcal{F}, \tag{4}$$

where b is a threshold. Thus, linear regression in a high-dimensional (feature) space corresponds to nonlinear regression in the low-dimensional input space R^N. Note that the dot product in (4) between ω and $\Phi(\bar{x})$ would have to be computed in this high-dimensional space (which is usually intractable), if we are not able to use the kernel that eventually leaves us with dot products that can be implicitly expressed in the low-dimensional input space R^N. Since Φ is fixed, we determine ω from the data by minimizing the sum of the empirical risk $R_{\mathrm{emp}}[f]$ and a complexity term $\|\omega\|^2$, which enforces flatness in feature space

$$R_{\mathrm{reg}}[f] = R_{\mathrm{emp}}[f] + \lambda \|\omega\|^2 = \sum_{i=1}^{l} C(f(\bar{x}_i) - y_i) + \lambda \|\omega\|^2, \tag{5}$$

where l denotes the sample size $(\bar{x}_1, \ldots, \bar{x}_l)$, $C(\cdot)$ is the penalty term and λ is a regularization constant. For a large set of loss function, (5) can be minimized by solving a quadratic programming problem, which is uniquely solvable. It can be shown that the vector ω can be written in terms of the data points

$$\omega = \sum_{i=1}^{l} (\alpha_i - \alpha^*) \Phi(\bar{x}_i), \tag{6}$$

with α_i, α_i^* being the solution of the aforementioned quadratic programming problem. α_i and α_i^* have an intuitive interpretation as forces pushing and pulling the estimate $f(\bar{x}_i)$ towards the measurements y_i. Taking (6) and (4) into account, we are able to rewrite the whole problem in terms of dot products in the low-dimensional input space

$$\begin{aligned} \mathrm{f}(\bar{x}) &= \sum_{i=1}^{l} (\alpha_i - \alpha_i^*)(\Phi(\bar{x}_i) \cdot \Phi(\bar{x})) + b \\ &= \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(\bar{x}_i, \bar{x}) + b, \end{aligned} \tag{7}$$

where α_i, α_i^* are Lagrangian multipliers, and $\bar{x}_i$ are support vectors.

In (7), we introduce a kernel function $K(\bar{x}_i, \bar{x}_j) = \Phi(\bar{x}_i) \cdot \Phi(\bar{x}_j)$. As explained in [29], any symmetric kernel function K satisfying Mercer's condition corresponds to a dot product in some feature space.

4.2. Approach. We propose to use SVM to model human driving behaviors. We consider human driving data $X(t)$ as the input vector of human dynamic behavior features as the time sequence. Since the SVM has an ability for classification, we use the human behavior data to "train" the SVM classifiers in the human dynamic behavior features space. For each driver, we want to design an SVM model to separate them from the other drivers. The task is to build up models across individuals by using the SVM training procedure. Then, since a dual class SVM model is capable of classifying two different classes of data only, in our applications, more than one dual class SVM is utilized and they are organized in a hierarchical manner. If there are n drivers to be recognized, $n - 1$ SVMs will be connected together as shown in Figure 6.

In any predictive learning task, such as classification, an appropriate representation of examples as well as the model and parameter estimation method should be selected to obtain a high level of performance of the learning machine. Traditional statistical approach to estimating models from data is based on parametric estimation. The basic fact that an assumption of an underlying dependency with a simple known parametric form is an ensuing need, limits its applicability in practice. Recent approaches allow a wide class of models of varying complexity to be chosen. The task of learning then amounts to selecting the model of optimal complexity and estimating parameters from training data. Under the SVM approach, the parameters to be usually chosen are the following.

(1) The penalty term C which determines the tradeoff between the complexity of the decision function and the number of training examples misclassified.

(2) The mapping function Φ.

(3) The kernel function such that $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$.

5. Experimental Study

In this section, we conduct experiments based on the proposed methodology for recognition of driver identities by analyzing the driving performances. In order to estimate the performance of the proposed system, we invite 7 human subjects to attend the experiment, who are Meng, Ou, Ye, Huang, Wang, Wu, and Shen. They are asked to "drive" on the designed experimental platform individually. The raw data of their driving behaviors is collected by the Data Sensing and Capturing Subsystem. The data recorded is to be analyzed by the Data Analysis Subsystem aforementioned. Our objective is to identify the driving data by trained SVM models. We use the accuracy rate of the SVM classifications to evaluate the performance of the proposed system. The experimental results of applying different data preprocessing methods and choosing different parameters of SVM when modeling human driving behaviors are shown in what follows.

5.1. Preprocess Data Analysis. In the first series of experiments, we run different data preprocessing methods for the optimization. The raw data is captured at a rate of 10 Hz and overlapping windows are applied on the data to cut the data into segments. Each segment is 40 seconds long and can be considered as a matrix of size 400×3.

We then apply FFT, PCA, and ICA to reduce the input size to the SVM for classification. The following steps are performed on each data segment.

(1) Apply FFT of order 20 to transform each column of data of size 400 into 20, so the result retrieved is a matrix of size 20×3, as each data segment. Then, align the data into a single row as a 1×60 vector.

(2) Divide the raw data matrix into 10 parts by time sequence and align these 40×3 matrix to a 40×30 matrix. Extract two features from the gained data matrix using PCA. With 2 PCs, a 2×30 feature retrieved is aligned to form a 1×60 vector.

(3) Divide the raw data matrix into 10 parts by time sequence and align these 40×3 matrix to a 40×30 matrix. Extract two features from the gained data matrix using ICA. With 2 ICs, a 2×30 feature retrieved is aligned to form a 1×60 vector.

We compare the data preprocessing using PCA and ICA with FFT. We simply train an SVM to distinguish one tester from all testing data, which is Meng's, to evaluate the performances of three methods of feature selection and data reduction. We have 2 groups of data containing 348 raw data segments totally, 104 segments representing the behaviors of driver Meng (the authorized driver) and 244 segments representing non Meng (the unauthorized drivers). These segments are sent to SVM for training and testing. Due to the aforementioned rules, each segment is processed to a 1×60 vector as the input to SVM.

Three data preprocessing methods are tested independently and the SVM testing results are shown in Table 1 for comparing the ability of feature selection of each method. As seen, FFT achieves the best performance among those three feature selections and data reduction methods, so we choose FFT as the data preprocessing for the further procedures.

Next we examine the different parameters of FFT to the classification results. Table 2 shows the test results of classification using different sampling times (length of data segment) and different FFT orders (size of input vector). Different sampling times from 10 seconds to 160 seconds and different FFT orders of 5, 10, and 20 are conducted. When using the sampling time t and FFT orders n, it means each data segment is a matrix of size $10t \times 3$ as the original sampling rate of the hardware is 10 Hz. The FFT transforms each column of data of size $10t$ into n and the result retrieved is a matrix of size $n \times 3$. Then the data is aligned into a single row as a $1 \times 3n$ vector for the SVM training and testing.

Table 2 shows the average success rate of identifying driver Meng from all testing data, using different sampling times and FFT orders. Different combination of sampling time and FFT order affects different classification rates.

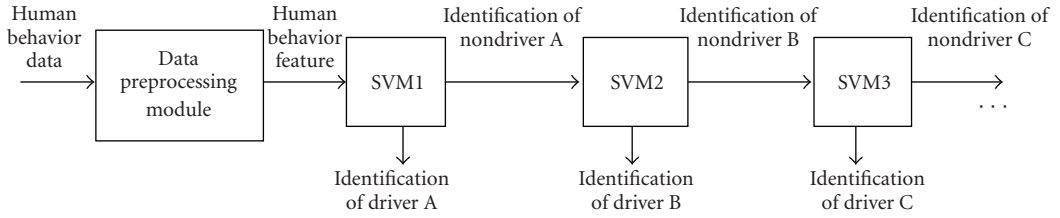


Figure 6: Block diagram of SVMs network classifier.

Table 1: Test results using different preprocessing methods.

	Number of errors (authorized driver)	Number of errors (unauthorized drivers)	Average success rate
FFT	22	46	80.46%
PCA	39	82	65.23%
ICA	47	79	63.79%

Table 2: Test results using different sampling time and FFT orders.

Sampling time	FFT (order 5)	FFT (order 10)	FFT (order 20)
10 sec.	64.92%	66.83%	64.68%
20 sec.	66.51%	66.51%	60.29%
40 sec.	69.23%	69.23%	78.85%
80 sec.	63.46%	55.77%	51.92%
160 sec.	65.38%	53.85%	65.38%

Shorter sampling time and smaller FFT order may cause the loss of important features involved in the human behavior signal, while longer time and larger order may lead to a mixed redundant signal to the classification procedure, both of them lower than the recognition rate. By comparing the success rate, sampling time at 40 seconds and FFT order of 20 achieve the best results. In addition, the data segment length of 40 seconds, namely, the time interval of the system examining the current driver, is efficient for not causing huge data segments for computation and providing adequate sampling frequency for testing driving performance in realtime.

5.2. Models Design. Training SVM requires the selection of parameters which influence the ensuing model performance. Therefore, to achieve a good model those parameters have to be chosen correctly. Examples, as stated earlier, are (1) cost function C and (2) the mapping function Φ. In the first part of our experiments, we have considered Gaussian radial basis function (RBF) as the kernel function. The RBF kernel is very advantageous in complex nonseparable classification problems due to its ability of nonlinear input mapping. It has the property that $\Phi(x) \cdot \Phi(y) = \exp(-\gamma \|x - y\|^2)$, and subsequently γ (defined as $1/2\sigma^2$ being the kernel width) is an important parameter to be chosen.

In this series of experiments, we run the SVM classifier with several values of C and γ somehow trying to determine which combination of parameters might be the best for a "good" model. That is, the one that could better express the causal relation among variables which govern the quality within the driving platform. This is accessed through the evaluation of performance accuracy. One possible way is to divide the original data into a data training set and into a validation data set for model evaluation. Figure 7 shows the testing accuracy as a function of kernel γ both parameterized with C for identification of driver Meng. Figure 8(a) shows the variation of number of SVs versus γ and C and Figure 8(b) shows the variation of number of learning iterations versus γ and C. The stopping tolerance for solving the optimization problem is set to 10^{-3}.

In the second part of our experiments, we consider a polynomial kernel as the kernel function. It has the property that $\Phi(x) \cdot \Phi(y) = (\gamma \times x' \times y + c)^d$, where $c = 0$ and $d = 3$ as the default settings in this experiment, and subsequently γ (defined as $1/2\sigma^2$ being the kernel width) is an important parameter to be chosen.

Figure 9 shows the testing accuracy as a function of kernel γ both parameterized with C for identification of driver Meng. Figure 10(a) shows the variation of number of SVs versus γ and C and Figure 10(b) shows the variation of number of learning iterations versus γ and C. The stopping tolerance for solving the optimization problem is set to 10^{-3}.

From the results shown above, larger C corresponds to a smaller number of SVs as well as a larger number of training iterations. Larger γ corresponds to poorer balanced classification, which means the deflective classifier model causing the lower accuracy on one side of the classification destination but higher accuracy on the other side, as well as the lager C doing so. Further explanation is required for these results taking into account both C and γ parameters. For nonseparable data, the penalty term $C \sum_{i=1}^{l} (f(\overline{x}_i - y_i))$ is able to reduce the training errors in the working data set. Therefore, the margin is an indicator of the generalization accuracy. In the absence of a method to compute the best tradeoff between the regularization term and the training errors, the balance sought by the SVMs technique is hard to find. Thus, a larger C corresponds to a higher penalty of training errors and clearly overfitting occurs. On the other

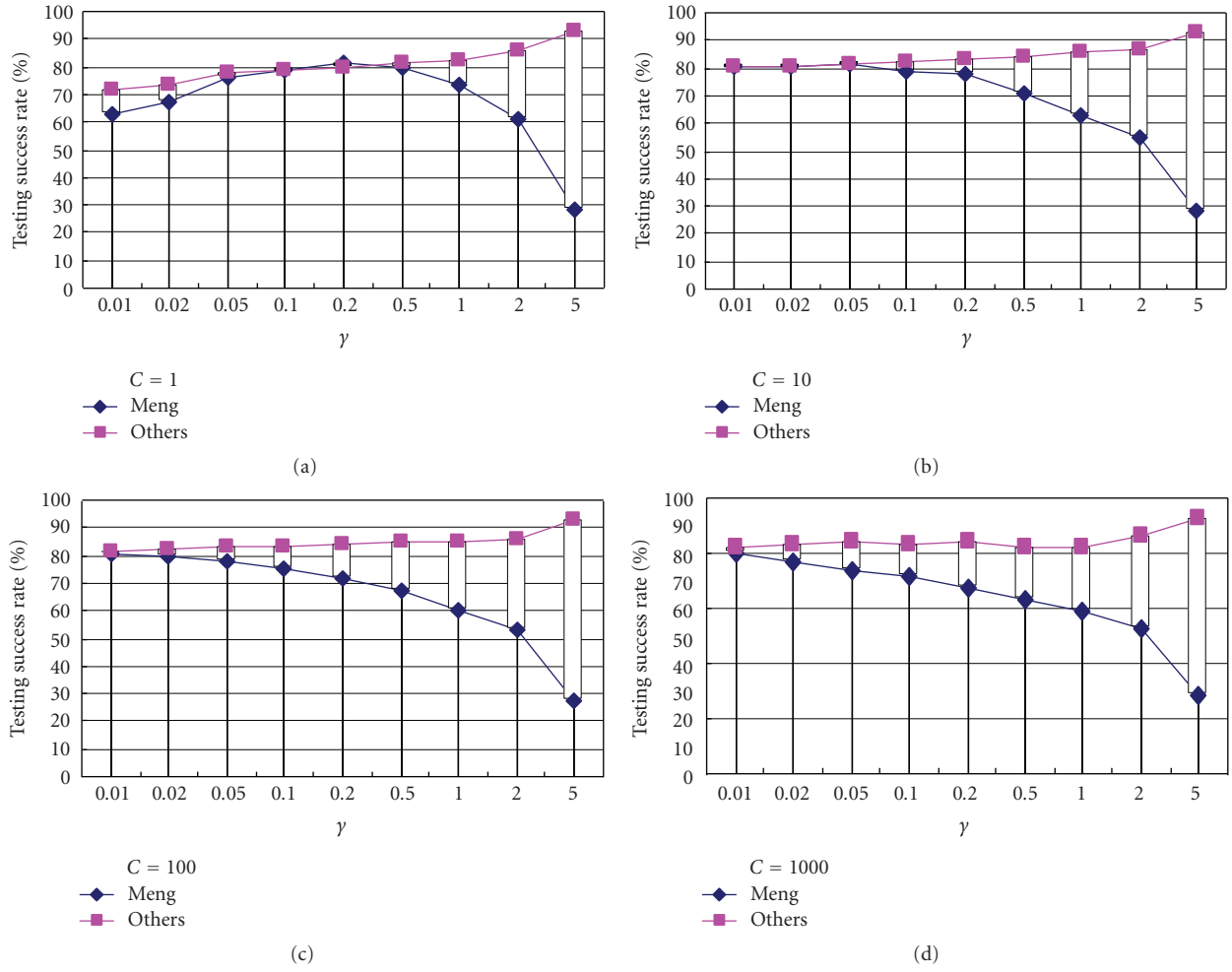


Figure 7: Success rate of model Meng versus Others parameterized with γ.

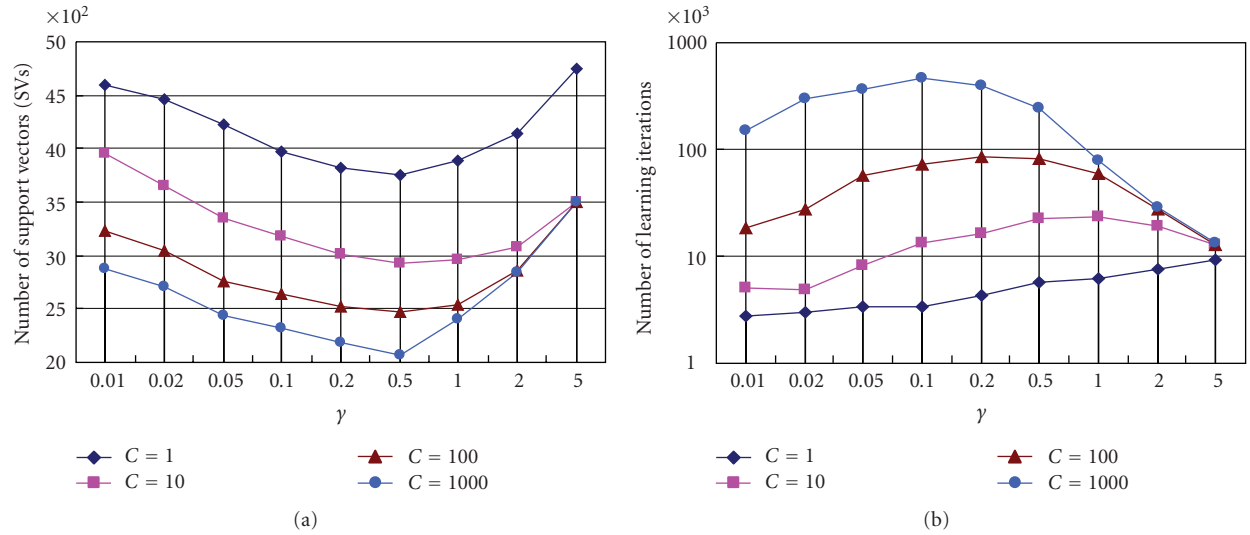


Figure 8: (a) Number of SVs versus γ; (b) Number of learning iterations versus γ (model Meng versus Others parameterized with C).

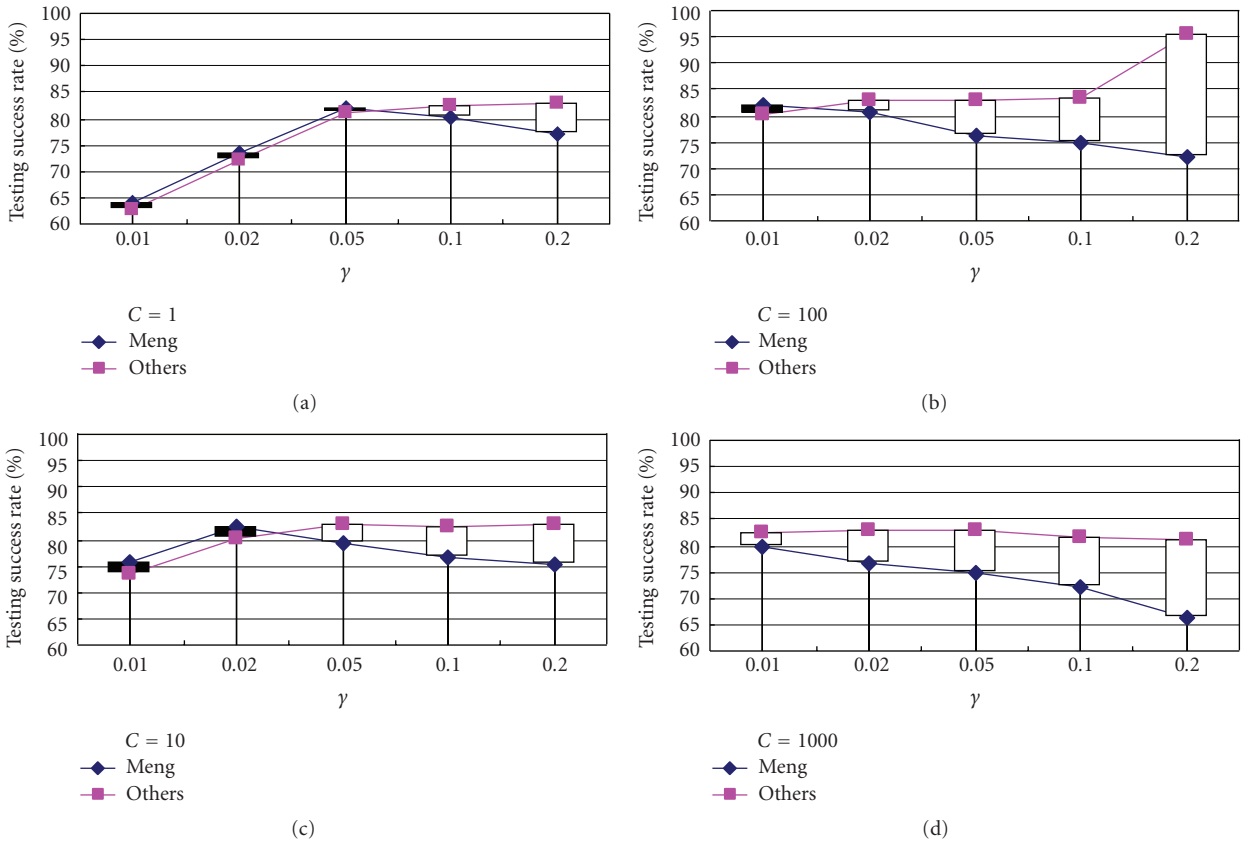


Figure 9: Success rate of model Meng versus Others parameterized with γ.

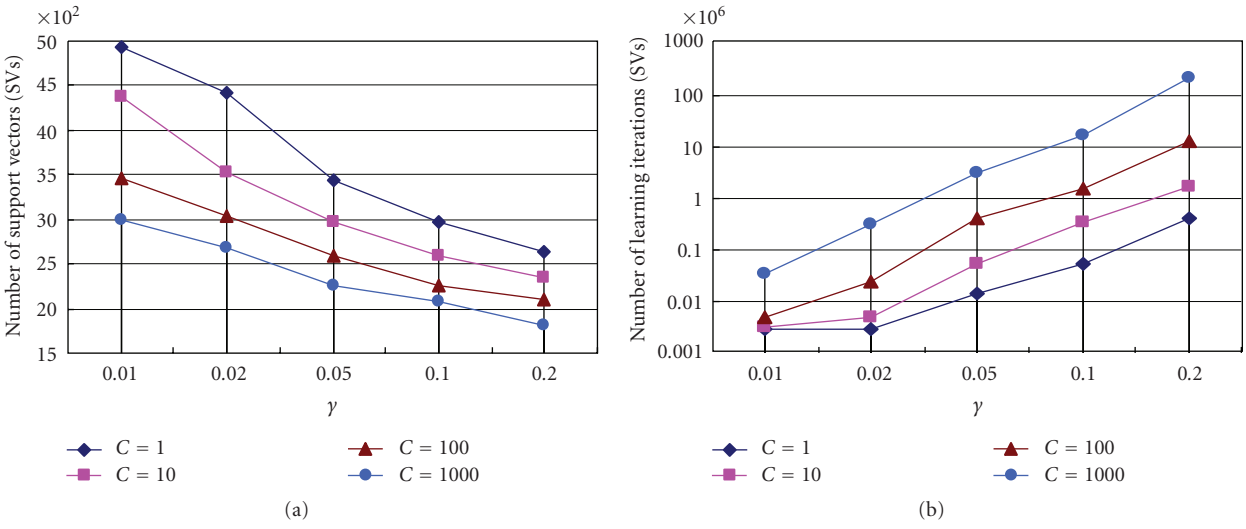


Figure 10: (a) Number of SVs versus γ; (b) Number of learning iterations versus γ (model Meng versus Others parameterized with C).

hand, the higher when the kernel parameter γ becomes, the greater the variety of the decision boundaries that can be formed originating a more complex model. The added flexibility initially decreases the generalization error as the model can better fit the data. However, there is the danger that this can lead to overfitting as well.

Due to the requirements of the proposed system, we aim to achieve a high classification accuracy as well as low computational consumption. The aim of the identification system is for the vehicle to judge if it is his own driver, so we set the Meng's success rate with higher priority. It is found experimentally that $C = 10$ and $\gamma = 0.02$ in the polynomial

Table 3: Test results on driver identifications via SVM.

Meng	Ou	Ye	Huang	Wang	Wu	Shen
81.45%	93.62%	77.08%	84.62%	79.31%	90.00%	89.47%

kernel has the highest success rate for Meng. With the same method, we train 7 SVM models for all 7 human subjects and utilize the SVM network to form a multiple classifier. To test with our testing samples and evaluate with the identification accuracy, we derive the final results in Table 3. As seen, we achieve the average success rate over 85%.

5.3. Discussion. In this section, we demonstrate that SVM is a feasible parametric model for our proposed application. The first aspect investigated is to use preprocessing methods for feature extraction from large human dynamic behavior data for modeling purposes. The extension of the implementation is to the data sets in a larger scale and different methods of problem multiclass formulation. The feature extraction method based on FFT is found to be able to give the best data reduction results compared to PCA and ICA in the presented experiments of modeling human driving behaviors through SVM. FFT establishes a one-to-one mapping between the time domain and the frequency domain and preserves information from the original signal, ensuring that important features are not lost as a result of the transformation. Under the experimental criteria in this paper, FFT is proved to have a better performance to model human dynamic behavior for driver identification than PCA and ICA. Although PCA and ICA are often used for input reduction, it is not always useful because the variance of a signal is not necessarily related to the importance of the variable. Human behavior contains much signals at lower frequencies and FFT can retain the energy at this frequent area but PCA and ICA work bad as there are too many isotropically distributed clusters. By reducing the redundancy in the input data, the training process of the human driving behavior model becomes more efficient. After the unnecessary information is removed from the inputs, not only the key characteristics of the human behavior data can be retained, but also the modeling power of the SVM is actually improved.

Besides choosing preprocessing methods, the SVM model design is an important issue in this section. We have discussed the application of the multiclass SVM's classifiers and compared them with different SVM parameters to identify different drivers. The basic idea of SVM is to determine the structure of the classifier by minimizing the bounds of the training error and generalization error. The SVs close to the boundary decision surface determine the efficacy of the classifier. Based on the results from our application, SVM with polynomial kernel achieves the better performance. Our results demonstrate that the SVMs have the potential to obtain a reliable distinction among our testing human subjects, individual identification can be recognized with the multiclass SVM's classifiers with a success rate of over 85%, which verifies that the proposed SVM modeling method is valid and useful against the vehicle thefts problem.

6. Conclusion and Future Work

In this paper, we focus on the research of utilizing dynamic human behavior models for vehicle security (preventing vehicles from being stolen) application. By learning from driving performances, the intelligent classifier can be embedded into an IC-based car key, through which the vehicle security system can identify the valid drivers based on the ways the vehicle are driven and the drivers behave.

6.1. Conclusions. We proposed the innovative idea on driver identification system for detecting vehicle theft based on dynamic human behaviors. The dynamic and stochastic feature is difficult to be handled by traditional mathematical methods. We compared FFT, PCA and ICA in the data preprocessing, and proved FFT has better performance to process human dynamic behavior. Thereafter, machine learning method based on SVM is applied. We discussed the application of the multi-class SVM classifiers and compared the performance of different SVM parameters. SVM with polynomial kernel performs better than other functions.

6.2. Future Work. Choosing the best parameters, especially if a systematic approach is not used and/or the problem knowledge do not aid for proper selection, can be time consuming since we have to rely upon guessing and trial-and-error techniques. Therefore, an interactive grid search model selection method can be studied, which may further enhance the accuracy.

Acknowledgments

This work is partially supported by Hong Kong RGC CUHK417605, Hong Kong ITF ITP/003/09AP, GHP/006/09SZ, the grant from Key Laboratory of Robotics and Intelligent System, Guangdong Province (2009A060800016), the Knowledge Innovation Program of the Chinese Academy of Sciences Grant No. KGCX2-YW-156, and the grant from Shenzhen Hong Kong Innovation Circle.

References

[1] "General Motor, OnStar," 2005, http://www.onstar.com/us_english/jsp/index.jsp.

[2] http://en.wikipedia.org/wiki/Electronic_key.

[3] A. Broggi, *Automatic Vehicle Guidance: The Experience of the ARGO Autonomous Vehicle*, World Scientific, Singapore, 1999.

[4] S. M. Smith, "ALTRUISM: interpretation of three-dimensional information for autonomous vehicle control," *Engineering Applications of Artificial Intelligence*, vol. 8, no. 3, pp. 271–280, 1995.

[5] S. L. M. Beauvais, C. Kreucher, and S. Lakshmanan, "Building world models for mobile platforms using heterogeneous sensors fusion and temporal analysis," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems (ITSC '97)*, pp. 230–235, 1997.

[6] H. Asada and S. Liu, "Transfer of human skills to neural net robot controllers," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2442–2448, 1991.

[7] M. Nechyba, *Learning and Validation of Human Control Strategy*, Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, Pa, USA, 1998.
[8] Y. Xu and Y. Ou, *Control of Single Wheel Robots, Springer Tracts in Advanced Robotics*, Springer, Berlin, Germany, 2005.
[9] D. A. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*, Kluwer Academic, Boston, Mass, USA, 1994.
[10] M. Nechyba and Y. Xu, ""Learning and transfer of human real-time control strategies," *Journal of Advanced Computational Intelligence*, vol. 1, no. 2, pp. 137–154, 1997.
[11] M. C. Nechyba and Y. Xu, "Stochastic similarity for validating human control strategy models," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 3, pp. 437–451, 1998.
[12] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*, MIT Press, Cambridge, Mass, USA, 2002.
[13] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.
[14] B. Schőlkopf, C. Burges, and A. Smola, *Advances in Kernel Methods Support Vector Learning*, MIT Press, Cambridge, Mass, USA, 1999.
[15] V. Vapnik and A. Chervonenkis, "A note on one class of perceptrons," *Automatic Remote Control*, vol. 25, 1964.
[16] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory (COLT '92)*, pp. 144–152, 1992.
[17] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
[18] D. J. Sebald and J. A. Bucklew, "Support vector machine techniques for nonlinear equalization," *IEEE Transactions on Signal Processing*, vol. 48, no. 11, pp. 3217–3226, 2000.
[19] T. B. Trafalis and H. Ince, "Support vector machine for regression and applications to financial forecasting," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN '00)*, vol. 6, pp. 348–353, Como, Italy, 2000.
[20] Y. Ou, H. Qian, and Y. Xu, "Support vector machine based approach for abstracting human control strategy in controlling dynamically stable robots," *Journal of Intelligent and Robotic Systems*, vol. 55, no. 1, pp. 39–54, 2009.
[21] P. Huang and Y. Xu, "SVM-based learning control of space robots in capturing operation," *International Journal of Neural Systems*, vol. 17, no. 6, pp. 467–477, 2007.
[22] X. Wu, Y. Ou, H. Qian, and Y. Xu, "A real time face classification and counting system," *International Journal of Information Acquisition*, vol. 5, no. 1, pp. 1–10, 2008.
[23] M. Chen, B. Huang, and Y. Xu, "Intelligent shoes for abnormal gait detection," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '08)*, pp. 2019–2024, Pasadena, Calif, USA, May 2008.
[24] H. Qian, T. L. Lam, W. Li, C. Xia, and Y. Xu, "System and design of an omni-directional vehicle," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pp. 389–394, Bangkok, Thailand, February 2008.
[25] T. L. Lam, H. Qian, Y. Xu, and G. Xu, "Omni-directional steer-by-wire interface for four wheel independent steering vehicle," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1383–1388, 2009.
[26] W. H. Press, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge, UK, 2nd edition, 1992.
[27] F. Hlawatsch and G. F. Boudreaux-Bartels, "Linear and quadratic time-frequency signal representations," *IEEE Signal Processing Magazine*, vol. 9, no. 2, pp. 21–67, 1992.
[28] A. Hyvarinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 626–634, 1999.
[29] S. Bernhard, C. J. C. Burges, and A. Smola, *Advanced in Kernel Methods Support Vector Learning*, MIT Press, Cambridge, Mass, USA, 1998.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 743010, 9 pages
doi:10.1155/2010/743010

Research Article

The New Evolution for SIA Rotorcraft UAV Project

Juntong Qi, Dalei Song, Lei Dai, Jianda Han, and Yuechao Wang

State Key Laboratory of Robotics, Shenyang Institute of Automation, CAS Nanta Street 114, Shenyang 110016, China

Correspondence should be addressed to Juntong Qi, qijt@sia.cn

Received 31 October 2009; Accepted 23 March 2010

Academic Editor: Zeng-Guang Hou

This paper describes recent research on the design, implement, and testing of a new small-scaled rotorcraft Unmanned Aerial Vehicle (RUAV) system—ServoHeli-40. A turbine-powered UAV weighted less than 15 kg was designed, and its major components were tested at the Shenyang Institute of Automation, Chinese Academy of Sciences in Shenyang, China. The aircraft was designed to reach a top speed of more than 20 mps, flying a distance of more than 10 kilometers, and it is going to be used as a test-bed for experimentally evaluating advanced control methodologies dedicated on improving the maneuverability, reliability, as well as autonomy of RUAV. Sensors and controller are all onboard. The full system has been tested successfully in the autonomous mode using the multichannel active modeling controller. The results show that in a real windy environment the rotorcraft UAV can follow the trajectory which was assigned by the ground control station exactly, and the new control method is obviously more effective than the one in the past year's research.

1. Introduction

Unmanned aerial vehicles (UAVs) are useful for many applications where human intervention is considered difficult or dangerous. Traditionally, the fixed-wing UAV has been served as the unit for these dangerous tasks because the control is easy. Rotary-wing UAV, on the other hand, can be used in many different tasks where the fixed-wing one is unable to achieve, such as vertical take-off/landing, hovering, lateral flight, pirouette, and bank-to-turn. Due to the versatility in maneuverability, helicopters are capable to fly in and out of restricted areas and hover efficiently for long periods of time. These characteristics make RUAV applicable for many military and civil applications.

However, the control of RUAV is difficult. Although some control algorithms have been proposed [1–6], most of them were verified by simulation instead of real experiments. One reason for this is due to the complicate, nonlinear, and inherently unstable dynamics, which has cross coupling between main and tail rotor, and lots of time-varying aerodynamic parameters. Another reason is that the flight test is in high risk. If an RUAV lost its control, it would never be stabilized again.

Based on our UAV research in [7], this paper details the development of a new unmanned helicopter (UAV) test bed—ServoHeli-40 (Figure 1) and the advanced control experiments performed toward achieving full autonomous flight. The experimental platform which has 40 kilograms takeoff weight is designed and finished by our research group in Shenyang Institute of Automation, Chinese Academy of Sciences. The brief of this paper is as follows: the ServoHeli-40 platform is introduced in Section 2. The introduction of sensor package is in Section 3. The modeling of the UAV helicopter system is presented in Section 4. In Section 5, we introduce active modeling scheme as a baseline control of the platform. In the end, there will be a conclusion of our work and also some discussions about future research issues.

2. ServoHeli-40 Platform Description

The entire experiment was implemented on the ServoHeli-40 small-size helicopter platform (Figure 1).

ServoHeli-40 aerial vehicle is a high-quality helicopter, which is changed by us using an RC technical grade helicopter operating with a remote controller. The modified

Figure 1: ServoHeli-40 small-size helicopter platform.

Table 1: Physical characteristics of ServoHeli-40 small-size helicopter.

Length	2.12 m
Height	0.73 m
Main rotor diameter	2.15 m
Stabilizer bar diameter	0.75 m
Rotor speed	1450 rpm
Dry weight	20 kg
Engine	2-stroke, air cooled
Flight time	45 min

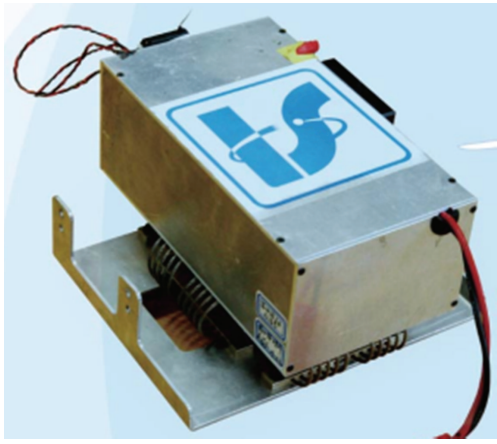


Figure 2: Airborne control box.

Figure 4: Hemisphere OEM GPS.

Figure 3: Crossbow IMU.

Figure 5: HMR3000 digital compass.

system allows a payload of more than 10 kilograms, which is sufficient to take the whole airborne avionics box and the communication units for flight control. The fuselage of the helicopter is constructed with sturdy duralumin, and composite body and the main rotor blades are replaced with heavy-duty fiber glass reinforced ones to accommodate extra payloads. The vehicle is powered by a ZENOAH engine which generates 9 hp at about 10000 rpm, a displacement of 80 cc, and practical angular rate ranging from 2,000 to 16,000 rpm. The full length of the fuselage is 2120 mm as well as the full width of it is 320 mm. The total height of the helicopter is 730 mm, the main rotor is 2150 mm, and the tail rotor is 600 mm.

Designing the avionics box and packing the box appropriately under the fuselage of the helicopter are two main tasks to implement of this UAV helicopter system. In the actual flight environment, the weight and the size of the avionics box are strict limited. Our airborne control box, which is shown in Figure 2, is a compact aluminum alloy package mounted on the landing gear. The center of gravity of the box lies on the IMU device, where it is not the geometry center of the system that keeps the navigation data

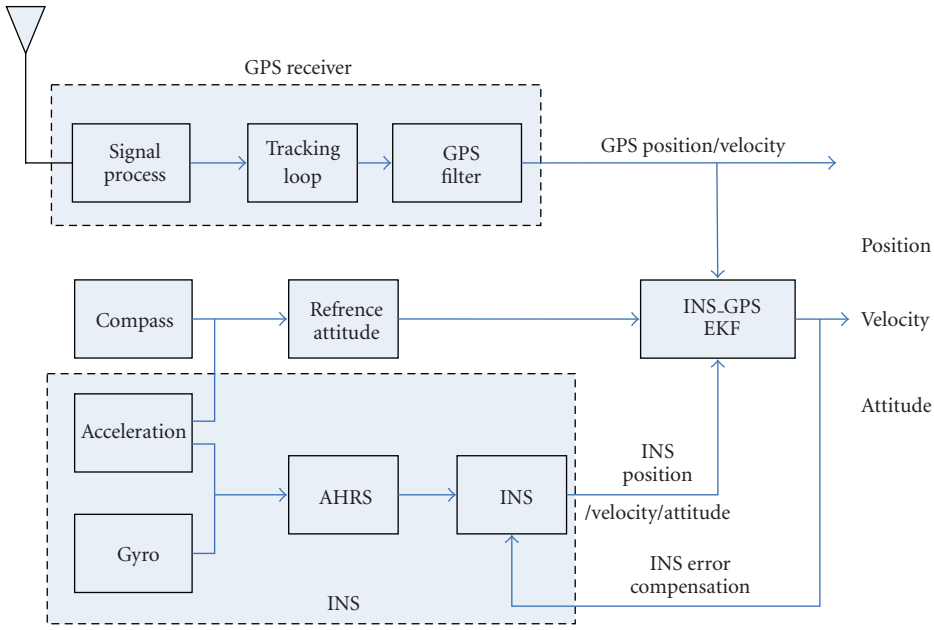


Figure 6: GPS-INS Navigator Structure.

from IMU accurate. The compass and the IMU are taken as the horizontal center of the gravity of the avionics system and the other components are installed on the same line.

The platform, which is fixed with a 3-axis gyro, a three-axis acceleration sensor, a compass, and a GPS, can save the data of velocities, angular rates, Euler angle accelerations, and positions into an SD card through an ARM processor. An Extended Kalman Fliter is used to estimate the sensors values. There is a CPLD used for sampling control inputs from the remote control of the pilot. The rotor speed is controlled by a Governor, an electronic unit for engine control. Table 1 shows the physical characteristics of ServoHeli-40 small-size helicopter. The type of the sensors and the method for navigation will be described in the next section.

3. Sensors of the ServoHeli-40

3.1. Sensors for Attitude and Position Estimation. In order to navigate following a desired trajectory while stabilizing the vehicle, the information about helicopter position, velocity, acceleration, attitude, and the angular rates should be known to the guidance and control system. The rotorcraft UAV system is equipped with sensors including inertial sensor unit, GPS, digital compass, rotor speed sensor, air-press altimeter, and ultrasonic sensor to obtain above accurate information about the motion of the helicopter in association with environmental information.

The Crossbow IMU300, which is shown in Figure 3, is a six-axis measurement system designed to measure the linear acceleration along three orthogonal axes and rotation rate around three orthogonal axes. It employs on-board digital processing to provide application-specific outputs and to compensate for deterministic error sources within the unit. Solid-state MEMS sensors make the IMU300 product responsive and reliable.

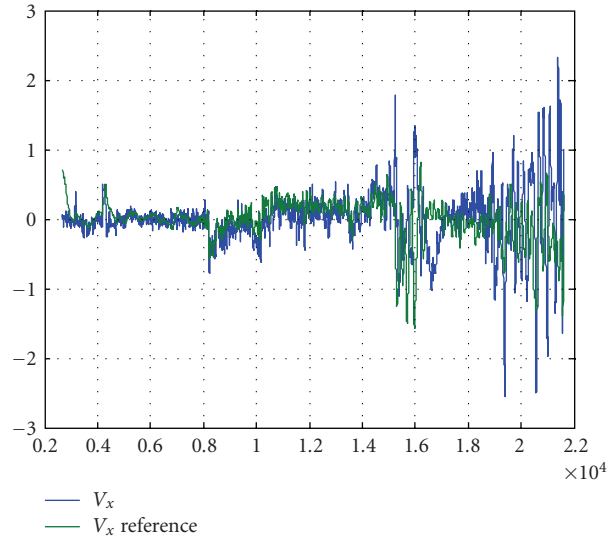


Figure 7: Velocities before and after filtering.

Hemisphere GPS, which is shown in Figure 4, is a space-based satellite radio navigation system developed by a Canada company. GPS provides three-dimensional position and time with the deduced estimates of velocity and heading. The GPS provides position estimates at up to 10 Hz. For operation, the GPS and the antenna are installed on the host aerial vehicle.

HMR3000 digital compass, which is presented in Figure 5, is an electronic compass module that provides yaw, pitch, and roll output for navigation and guidance systems. This compass provides fast refrash rate of up to 20 Hz and a high accuracy of 0.5 degree with 0.1-degree resolution.

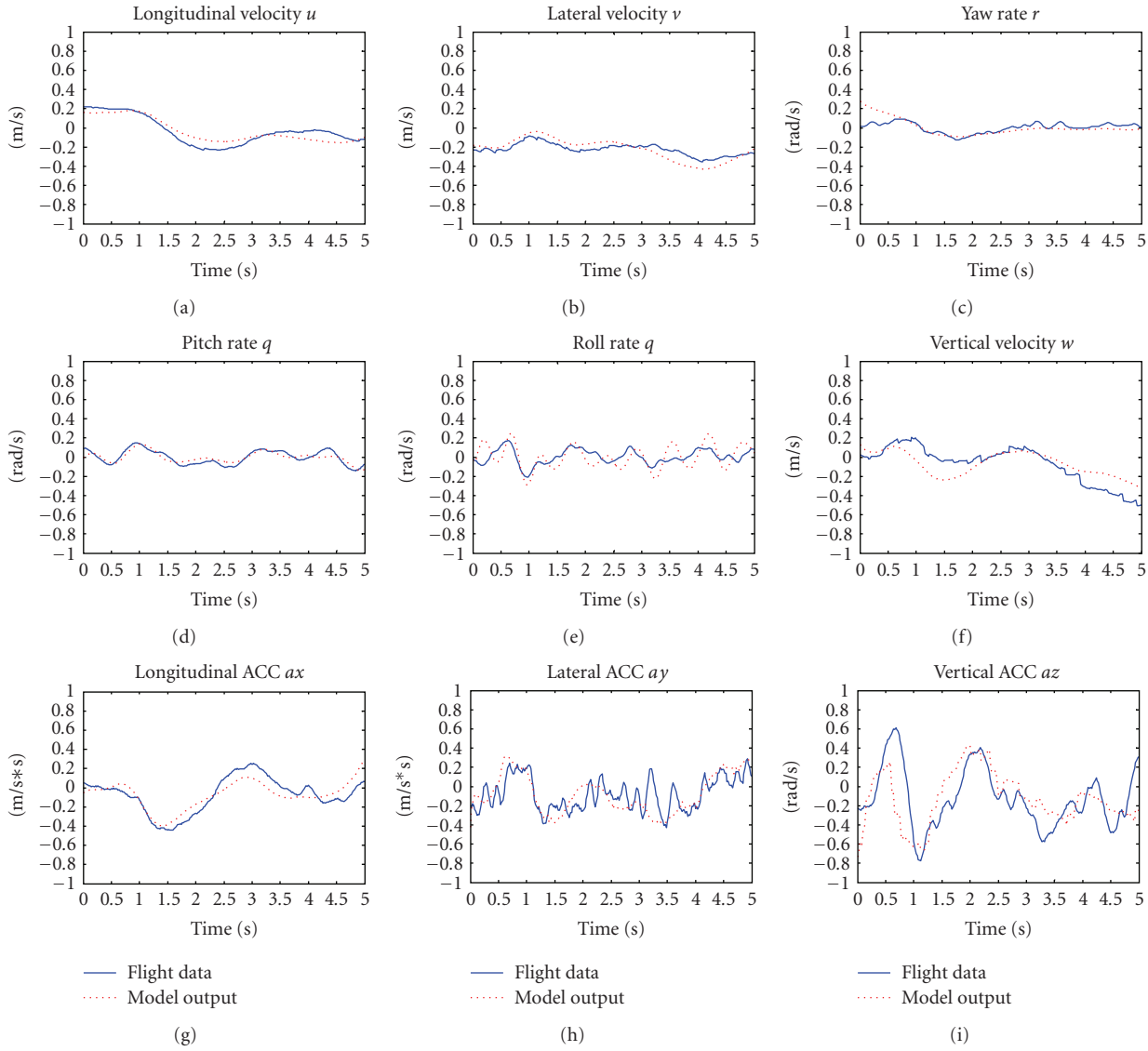


Figure 8: Rotorcraft UAV modeling verification.

In order to get the accurate altitude information of the vehicle, an air-pressure altimeter that is collecting data higher than 5 meters as well as an ultrasonic sensor that is getting the information on other situations is equipped under the avionics box.

The update rate of all sensors is ranging from 10 to 100 Hz, which is enough for implementation for advanced control algorithms.

3.2. EKF Based Navigator Design. In our avionics box, we use rate gyros and accelerometers to measure rates about three axes and accelerations along 3 axes; an independent ARM processor is used to extract absolute roll and pitch. However, in the real flight environment, the sensors will be subjected to rotor frequency vibrations; both the rate and acceleration readings are grossly inaccurate and consequential; so it is the attitude estimation.

In order to isolate the unit from these frequencies, we use EKF method to build an effective navigator for flight information estimation. The proposed GPS_INS Navigator is in Figure 6, and a typical plot of the forward velocities before and after filtering is given in Figure 7.

4. Rotorcraft UAV Modeling

For effective hovering identification, the original model in [8] is decomposed into three groups (longitudinal, lateral, and yaw-heave coupling), and a semidecoupled model is obtained. Each group has a decoupled system matrix, and the coupling characteristics are presented only in the control matrix. Thus, the number of unknown parameters and control inputs is reduced and the control loops are semidecoupled. Then, to identify the unknown parameters in the MIMO semidecoupled model, a new cost function is

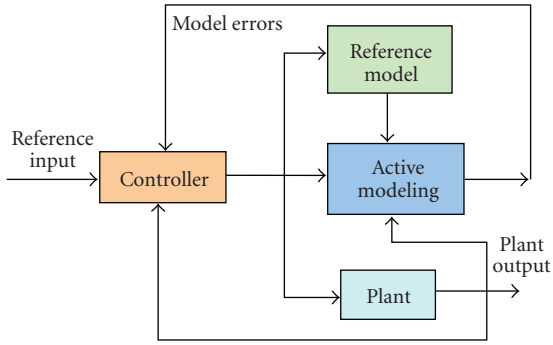


Figure 9: The scheme of active model-based control.

Figure 10: ServoHeli-40 small-size helicopter platform for experiment use.

proposed to make the traditional method of SISO system frequency estimation [9] applicable to the MIMO state-space models. The proposed cost function is presented in the addition form of the frequency error of every input-output pair for transfer matrix, and the parameters are identified by minimizing the cost function. The simplified model and proposed identification method free the selection of initial estimation and constraint is not required.

We have got the numerical model, and then the other serial of input data was tested using the proposed model. The blue line is the measurement of the yaw rate from the real flight as well as the red line is calculated by the numerical model and the real flight input. As is shown in Figure 8, the estimation output is similar to the real flight data and we can conclude that the proposed modeling method is useful to the rotorcraft UAV.

5. Active Modeling Control Scheme

Figure 9 illustrates the active model-based control scheme. The error between the reference model and the actual dynamics of the controlled plant is estimated by an online modeling strategy. The control, which is designed according to the reference model, should be able to compensate the estimated model error and fix it in real time. In the followings of this paper, we use the Adaptive Set Membership Filter (ASMF) [10] as the active modeling algorithm and the modified GPC as the control.

For normal missions of an unmanned helicopter, the flight modes include hovering (velocity under 5 m/s), cruising (velocity above 5 m/s), taking off and landing (distance to the ground is below 3 m while significant ground effect exists), and the transitions among these modes. A reference model is typically obtained by linearizing the nonlinear dynamics of a helicopter at one flying mode. The model errors from linearization, external disturbance, simplification, and unmodeled dynamics can be considered as additional process noise. Thus, a linearized state-space model for helicopter dynamics in full flight envelope can be formulated as

$$\begin{aligned} \dot{X} &= A_0 X + B_0 U + B_f f(X, \dot{X}, W), \\ Y &= CX, \end{aligned} \tag{1}$$

where $X \in R^{13}$ is the state, including 3-axis velocity, pitch and roll angle, 3-axis gyro values, flapping angles of main rotor and stabilizer bar, and the feedback of yaw gyro. $Y \in R^8$ is the output, including 3-axis velocity, pitch and roll angle, and 3-axis gyro values, A_0 and B_0 contain parameters that can be identified in different flight modes, and we use them to describe the parameters in hovering mode. $U \in R^4$ is the control input vector. The detail of building the nominal model and physical meaning of parameters is explanted in [11].

To describe the dynamics change, in (1), here, we introduce $f(X, \dot{X}, W, t) \in R^{13}$ to represent the time varying model error in full flight envelope, and $W \in R^{13}$ is the process noise.

5.1. Velocity Tracking Control. All flight tests are performed on the ServoHeli-40 setup (Figure 10), which was developed in the State Key Laboratory, SIACAS. It is equipped with a 3-axis gyro, a 3-axis accelerometer, a compass, and a GPS. The sensory data can be sampled and stored into an SD card through an onboard DSP.

Generalized predictive control (GPC), stationary increment predictive control (SIPC), and active model-based stationary increment predictive control (AMSIPC), which are proposed by us in [11], were all tested in the same flight conditions, and the comparison results are shown in Figures 11–13. The identified parameters were in [12] as nominal model.

It can be seen that when the helicopter increases its longitudinal velocity and changes flight mode from hovering to cruising, GPC (brown line) has a steady velocity error and increasing position error because of the model errors. SIPC (blue line) has a smaller velocity error because it uses increment model to reject the influence of the changing operation point and dynamics' slow change during the flight. However, the increment model may enlarge the model errors due to the uncertain parameters and sensor/process noises, resulting in the oscillations in the constant velocity period (clearly seen in Figures 11 and 12). While for AMSIPC (green line), because the model error has been online estimated

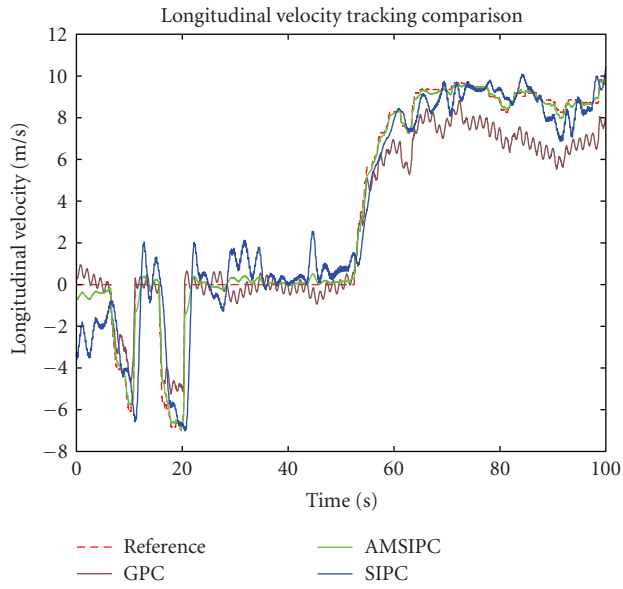


Figure 11: Longitudinal tracking results.

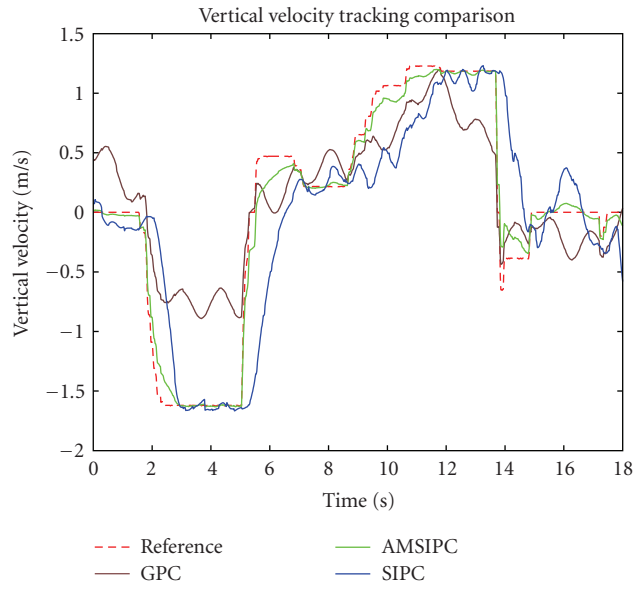


Figure 13: Vertical tracking results.

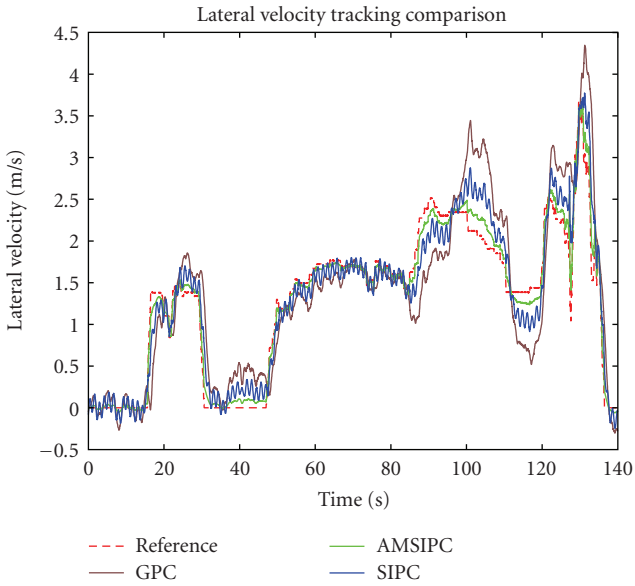


Figure 12: Lateral tracking results.

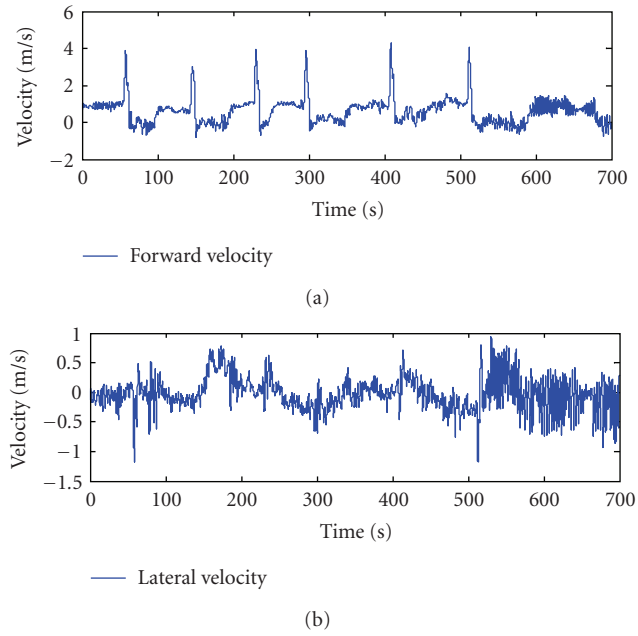


Figure 14: Forward and lateral velocity during the flight.

by the ASMF, the proposed AMSIPC successfully reduces velocity oscillations and tracking errors.

5.2. Preliminary Autonomous Flight Result. A two-loop control scheme for the rotorcraft UAV system was designed and tested using the ServoHeli-40 platform, in [7]. Some specified trajectories were designed to be flown. These trajectories were selected in order to evaluate the inner loop and outer loop response over several different sequences of inputs. We selected a tunnel way to be followed, as is shown in Figure 16. The proposed controller handled this flight trajectory with minimal error; see Figure 17. Figures 14 and 15 show that the angles and velocities which are controlled by inner loops also get a stable response.

5.3. New Flight Result. In this experiment, the identified hovering model was selected as nominal model for PID controller parameters calculation. The helicopter flies between two selected points, point A and point B. In the flight, the helicopter turns head 180° to point B at point A first; then, the helicopter increases the longitudinal velocity from 0 m/s to 10 m/s. When the helicopter arrives at point B, it decreases the longitudinal velocity from 10 m/s to 0 m/s and flies back to point A in the same way. To verify the strategy for estimate

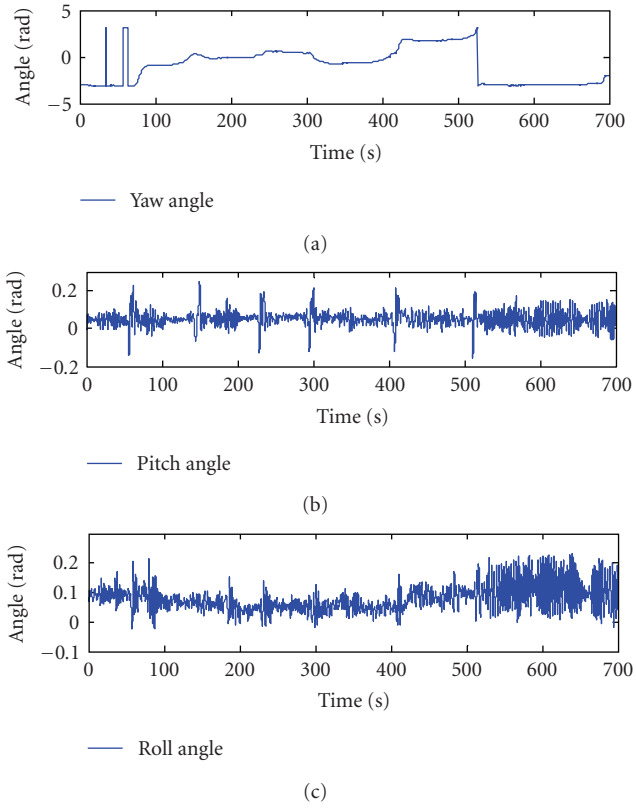


Figure 15: 3-axes angles during the flight.

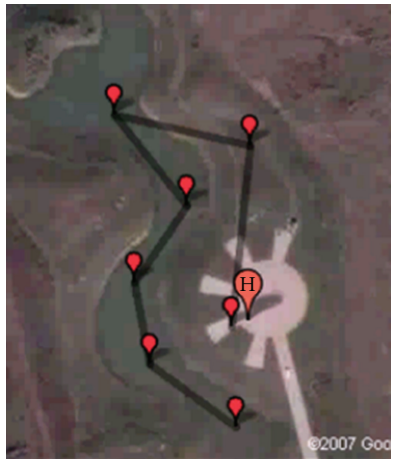


Figure 16: Trajectory in the Google Map.

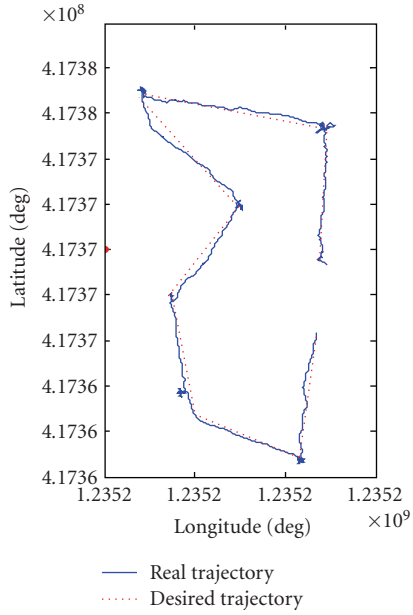


Figure 17: Desired and real trajectory.

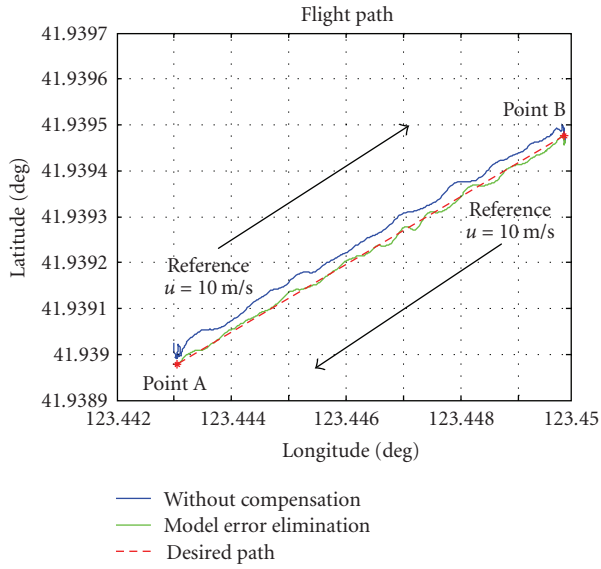


Figure 18: Flight path for test.

and elimination of the online model error, during the flight, we only use the PI controller without the strategy of model error elimination before time 150 seconds and compensate for the model error after that time in order to show the necessity of the strategy for model-error estimation and elimination. The flight path and position tracking errors are shown in Figures 18-19, and Figure 20 shows longitudinal velocity in the flight experiment. The wind in the flight environment is 3–6 m/s from the southeast.

It is clearly seen in Figures 18–20 that when the helicopter increases the longitudinal velocity and changes flight mode from hovering to cruise, the adaptive set-membership filter estimates the nonzero model errors and the estimated model error boundaries converge into the constant ellipse intersection, which means that the filter is stable. The effect of the strategy can be clearly seen in Figures 18–20, the lateral and vertical position errors decrease into 0.5 m, and the accuracy of longitudinal velocity tracking increases after the compensation, while nominal PID controller cannot get rid of the wind disturbance and changing flight mode that cause position error.

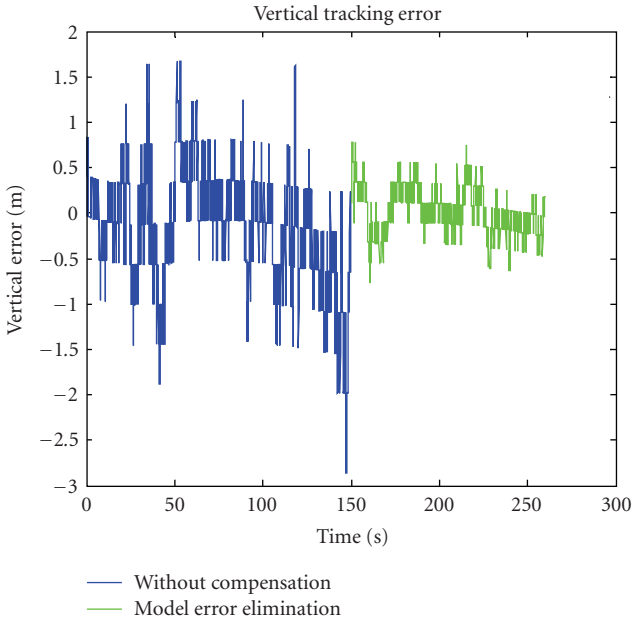


Figure 19: Vertical tracking error in real flight.

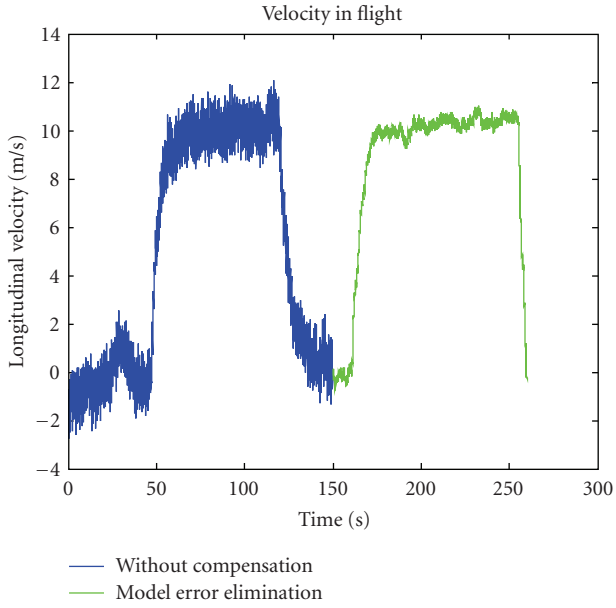


Figure 20: Longitudinal velocity in the flight.

6. Conclusions

This paper describes the current status of the ServoHeli-40 autonomous helicopter. We have introduced the system implementation of the rotorcraft UAV and control scheme for model-scaled helicopter. A reliable helicopter is built as the basic helicopter, which is changed to adapt to a heavier load in the future. We also introduce the sensors and algorithm for attitude and position estimation. The active modeling control and navigator works better than the two loop linear control scheme, which is applied by us in [7] last year.

The rotorcraft UAV system has been tested successfully for full autonomous flight including autonomous taking off and landing. The next step is to integrate the visual and IMU estimation into a unified sensor suite and to develop advantage autonomous flight control algorithm for maneuverable fight.

Acknowledgments

This work was supported by a Grant from the National High Technology Research and Development Program of China (863 Program) (no. 2007AA041503). The authors gratefully acknowledge the contribution of Shenyang Institute of Automation, Chinese Academy of Sciences, and reviewers' comments.

References

[1] C. P. Sanders, P. A. DeBitetto, and E. Feron, "Hierarchical control of small autonomous helicopters," in *Proceedings of the 37th IEEE Conference on Decision & Control*, vol. 4, pp. 3629–3634, Tampa, Fla, USA, December, 1998.

[2] M. A. Garratt and J. S. Chahl, "Visual control of an autonomous helicopter," in *Proceedings of the 41st Aerospace Sciences Meeting and Exhibit*, Reno, Nev, USA, January 2003.

[3] R. Enns and J. Si, "Helicopter ight control design using a learning control approach," in *Proceedings of the 39th IEEE Conference on Decision and Control*, pp. 1754–1759, Sydney, Australia, 2000.

[4] B. Bijnens, Q. P. Chu, G. M. Voorsluijs, and J. A. Mulder, "Adaptive feedback linearization flight control for a helicopter UAV," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, vol. 6, pp. 4463–4472, San Francisco, Calif, USA, August 2005.

[5] T. J. Koo, D. H. Shim, O. Shakernia, et al., "Hierarchical hybrid system design on Berkeley UAV," in *Proceedings of the International Aerial Robotics Competition (IARC '98)*, Richland, Wash, USA, August 1998.

[6] Z. Jiang, J. Han, Y. Wang, and Q. Song, "Enhanced LQR control for unmanned helicopter in hover," in *Proceedings of the 1st International Symposium on Systems and Control in Aerospace and Astronautics*, pp. 1438–1443, Harbin, China, January 2006.

[7] J. Qi, D. Song, L. Dai, and J. Han, "The ServoHeli-20 rotorcraft UAV project," in *Proceedings of the 15th International Conference on Mechatronics and Machine Vision in Practice (M2VIP '08)*, pp. 92–96, Auckland, New Zealand, December 2008.

[8] B. Mettler, C. Dever, and E. Feron, "Scaling effects and dynamic characteristics of miniature rotorcraft," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 3, pp. 466–478, 2004.

[9] J. S. Bendat and A. G. Piersol, *Engineering Application of Correlation and Spectral Analysis*, John Wiley & Sons, New York, NY, USA, 1993.

[10] B. Zhou, J. Han, and G. Liu, "A UD factorization-based nonlinear adaptive set-membership filter for ellipsoidal estimation," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 16, pp. 1513–1531, 2008.

[11] D. Song, J. Qi, and J. Han, "Active modeling based model predictive control for yaw-heave coupling dynamics of unmanned helicopters in full flight envelope," in *Proceedings of the AUVSI's Unmanned Systems North America*, vol. 1, pp. 108–121, 2009.

[12] D. L. Song, J. T. Qi, L. Dai, J. D. Han, and G. L. Liu, "Modeling a small-size unmanned helicopter using optimal estimation in the frequency domain," *International Journal of Intelligent Systems Technologies and Applications*, vol. 8, pp. 70–85, 2010.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 870923, 9 pages
doi:10.1155/2010/870923

Research Article

Solution for Ill-Posed Inverse Kinematics of Robot Arm by Network Inversion

Takehiko Ogawa and Hajime Kanada

Department of Electronics and Computer Systems, Takushoku University, 815-1 Tatemachi, Hachioji, Tokyo 193-0985, Japan

Correspondence should be addressed to Takehiko Ogawa, togawa@es.takushoku-u.ac.jp

Received 1 December 2009; Revised 9 April 2010; Accepted 7 July 2010

Academic Editor: Noriyasu Homma

In the context of controlling a robot arm with multiple joints, the method of estimating the joint angles from the given end-effector coordinates is called inverse kinematics, which is a type of inverse problems. Network inversion has been proposed as a method for solving inverse problems by using a multilayer neural network. In this paper, network inversion is introduced as a method to solve the inverse kinematics problem of a robot arm with multiple joints, where the joint angles are estimated from the given end-effector coordinates. In general, inverse problems are affected by ill-posedness, which implies that the existence, uniqueness, and stability of their solutions are not guaranteed. In this paper, we show the effectiveness of applying network inversion with regularization, by which ill-posedness can be reduced, to the ill-posed inverse kinematics of an actual robot arm with multiple joints.

1. Introduction

In the context of controlling a robot arm with multiple joints, the problem of estimating the joint angles from the given end-effector coordinates is called the inverse kinematics problem, which is a type of inverse problems [1]. Inverse problems that estimate the cause from the given results are studied in various engineering fields [2]. There are a number of methods for solving inverse kinematics problems: analytical method, iterative calculation by using an algorithm, and so forth [1]. In addition, neural-network-based inverse modeling has been proposed [3], and we can use it as a solution of inverse kinematics [4, 5]. The network inversion method has been proposed for solving inverse problems by using a multilayer neural network [6]. In this method, inverse problems are solved by using a trained multilayer neural network inversely to estimate the corresponding input from the given output [7, 8]. The advantages of this method are easiness of the direct modeling by learning and adaptive estimation of the inverse solution. It has been applied to actual problems [9, 10]. In addition, it was introduced as a method to solve the inverse kinematics problem of estimating the multiple joint angles of a robot arm from the given end-effector coordinates [11–13].

In general, inverse problems are affected by ill-posedness, which implies that the existence, uniqueness, and stability of their solutions are not guaranteed [2]. Ill-posedness also affects the solution when a problem is solved using network inversion. The regularization method to decrease ill-posedness by limiting the solution space of the inverse problem has been proposed for the ill-posed inverse problems [14, 15]. It has also been examined for network inversion [16, 17]. The inverse-modeling approach with a multilayer neural network and the approach that uses the network inversion method have been examined as neural-network-based methods. The problem of ill-posedness is an important aspect of inverse problems, and it has often been studied in the field of mathematical engineering [2, 15]. However, only a few studies have examined the possibility of solving the problem of ill-posedness by using a neural network [18]. In particular, a few systematic investigations of ill-posedness have been carried out by applying a neural network to the actual problem, but these investigations considered only the uniqueness of the solution [19].

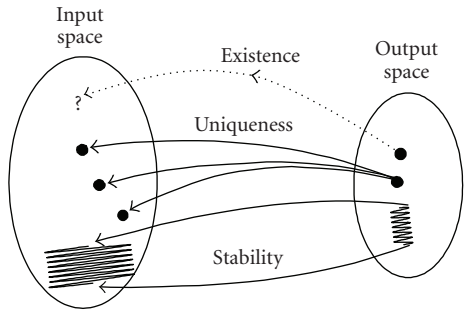


Figure 1: Concept of inverse problem and ill-posedness.

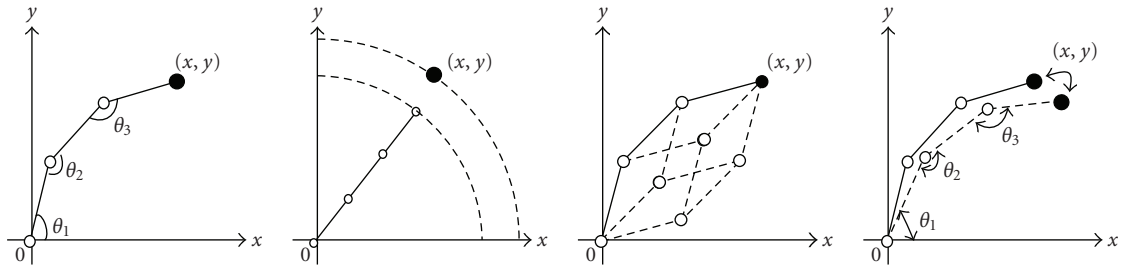


Figure 2: Inverse kinematics problem of 3-DOF robot arm in 2D plane. (a) Three joints and an end-effector coordinate. Ill-posed solutions in terms of (b) existence, (c) uniqueness, and (d) stability.

In this study, we aim at examining the applicability of the network inversion method to the inverse kinematics of a robot arm with multiple joints. That is, we aim to demonstrate the applicability of this method to the ill-posed inverse kinematics in terms of the existence, uniqueness, or stability of the solutions provided by the method. In addition, we aim to examine the applicability of the network inversion method to all problems that involve ill-posedness. We consider a three-degree-of-freedom (DOF) robot arm that moves in two-dimensional space and set the ill-posed problem as one in which the existence, uniqueness, or stability of the solution is not satisfied. We then examine the problem and solution by using the network inversion method so as to demonstrate the effectiveness of this method in addressing the inverse kinematics of a robot arm with multiple joints. We consider that the results suggest the effectiveness of the method when applied to inverse not only kinematics but also a general ill-posed inverse problem.

2. Inverse Estimation of Joint Angles of Robot Arm

An inverse problem determines the inner mechanism or cause from observed phenomena. In the case of a direct problem, the result is derived from a given cause by using a fixed mathematical model, whereas in the case of an inverse problem, the cause is estimated from the fixed model and given results. In the case of a direct problem, the existence, uniqueness, and stability of the solution are guaranteed. Problems in which these three conditions are satisfied are called as well-posed problems. In the case of an inverse problem, it may be difficult to obtain a solution because the well-posedness of the problem is not guaranteed [2]. An example of ill-posedness is shown in Figure 1.

Direct kinematics implies a problem of calculating the end-effector coordinates from the joint angles of robot arms. On the other hand, inverse kinematics inversely estimates the joint angles from the provided end-effector coordinates of robot arms. A number of analytical methods, algorithms of iterative calculation, and so forth exist for solving inverse kinematics problems. Analytical methods are effective for a robot arm of a low degree-of-freedom. They include the method of solving kinematics equations as nonlinear simultaneous algebraic equations, the method of solving an equation as a problem in planar geometry, and so on. Because an analytical method is not a general method, the iterative approximation algorithm that is based on the Newton method is generally used [1]. In addition, there is a method of solving inverse kinematics problems by using the adaptively estimated inverse model of kinematics [5]. The use of a neural network is proposed as a method of this inverse modeling [11]. Motion control of a robot arm is often achieved by the point-to-point or continuous-path methods, and it is required to solve the inverse kinematics problem.

In this paper, we consider a 3-DOF robot arm that moves in two-dimensional space, which is shown in Figure 2(a). We examine the problem of estimating the joint angles for the specified end-effector coordinate of the 3-DOF robot arm. In an inverse kinematic problem, the problem of ill-posedness in terms of the existence, uniqueness, and stability of the solution also arises. For instance, the combination of joint

angles for the specified end-effector coordinate may not exist, as shown in Figure 2(b). This is an ill-posed problem where a solution does not exist. Figure 2(c) shows an example where there are a number of combinations for the joint angles of an end-effector coordinate; here, the problem is ill-posed in terms of its uniqueness. In addition, a minor measurement error for the end-effector coordinate may cause a large estimation error of the joint angles according to scaling of the joint angles and end-effector coordinate, as shown in Figure 2(d); the problem is ill-posed because the solution is unstable [2].

In this paper, we introduce the network inversion method in order to solve the inverse estimation problem of joint angles for a 3-DOF robot arm. We examine the solutions to the above-mentioned ill-posed problems.

3. Network Inversion

An ordinary multilayer neural network is used to solve a direct problem. In a usual multilayer network in which the training is complete, the input-output relation is given by $y = f(w, x)$, where x, y, and f represent the input vector, output vector, and function defined by the interlayer weights w of the network, respectively. If the input vector x is provided, the network calculates the output vector y.

Linden and Kindermann proposed the network inversion method [6]. In this method, the observed output data y can be applied with f fixed after finding the forward relation f by training. The input x can then be updated according to the calculated input correction signal based on the duality of the weights and input. The input is estimated from the output by iterative updating of the input based on the output error, as shown in Figure 3. In this manner, the inverse problem for estimating the input x from the output y is solved with the multilayer neural network by inversely using the forward relation.

We use a network in two phases so as to solve the inverse problem by network inversion: forward training and inverse estimation. The procedure is shown in Figure 4. In the training phase, we provide the training input x and training output y to calculate the output error E. The weight w is then updated by

$$w(n+1) = w(n) - \varepsilon_t \frac{\partial E}{\partial w}, \tag{1}$$

where ε_t denotes the training gain, because the output error is due to maladjustments of the weights. By repeating this update procedure, the forward relation is obtained. This procedure is based on the usual back propagation method.

In the inverse estimation phase, we fix the relation obtained in the training, provide the random input x and test output y, and calculate the output error E. The input x is then updated by

$$x(n+1) = x(n) - \varepsilon_e \frac{\partial E}{\partial x}, \tag{2}$$

where ε_e denotes the input update gain, because the output error is due to the error in the input. By repeating this update procedure, the input is estimated from the output.

4. Ill-Posedness and Regularization

In general, inverse problems are ill-posed in that the existence, uniqueness, and stability of their solutions are not guaranteed [2]. Regularization, which limits the solution space of an inverse problem, is a method that reduces ill-posedness. In this method, a functional intended to provide some constraints is added to the error functional of the iterative method. For example, when the transform from the element x in space X to the element y in space Y is expressed as $Kx = y$ by mapping K, we search the element x to minimize the functional

$$E = ||y - Kx||^2 + \lambda F(x), \tag{3}$$

where $F(x)$ and λ represent the regularization functional and regularization coefficient, respectively.

We can also use regularization for network inversion [17]. In order to provide the constraint condition, we minimize the regularization functional in accordance with the output error in the inverse estimation phase. The energy function $E(x)$ with the regularization functional $F(x)$ is defined as

$$E(x) = ||\tilde{y} - Kx||^2 + \lambda F(x), \tag{4}$$

where $\tilde{y}$, K, and x indicate the network output, transform, and input, respectively. In (4), the first and second terms represent the output error of the network and the regularization functional, respectively. The parameter λ is the regularization coefficient. The objective of the regularization term $F(x)$ is to express the restraint condition that the input x should satisfy and to be minimized simultaneously with the error term. By using the regularization term, we aim to obtain a feasible solution that minimizes the error and satisfies the restraint condition. Since $F(x)$ is a restraint condition concerning the input, we can use the minimum norm to select a specific solution, the minimum difference to smooth the solution, and so forth. Moreover, we can use the minimum and maximum norms of the vector of the joint angles, the minimum and maximum inclinations of the arm, and so forth by expressing them as the regularization term $F(x)$ for the robot arm considered in this paper. The coefficient λ is usually fixed and determined empirically or by trial and error. However, the balance between the output error minimization and regularization is often uncertain. The regularization coefficient may actually be reduced with a decrease in the output error. We refer to this as dynamic regularization. For example, the regularization coefficient $\lambda(t)$ is reduced for each input correction as

$$\lambda(t) = \lambda(0) \exp(-mt), \tag{5}$$

where $\lambda(0)$, t, and m denote the initial value of λ, current epoch number, and decay coefficient, respectively. The parameter $\lambda(t)$ decays from $\lambda(0)$ to zero with the epoch number t.

Table 1: Network parameters.

Input neurons		3
Hidden neurons		10
Output neurons		2
Training rate ε_t		0.01
Inverse estimation rate ε_e		0.01
Training error to be attained		<0.001
Estimation error to be attained		<0.0001
Joint angles for training data (°)	θ_1	0, 15,. . ., 90
	θ_2	45, 60,. . ., 315
	θ_3	45, 60,. . ., 315
Arm length l (cm)		30

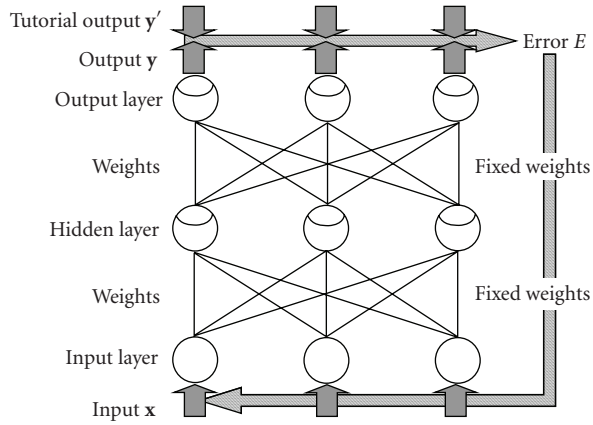


Figure 3: Iterative update of input from provided output by network inversion.

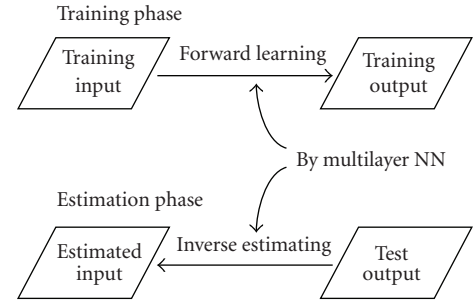


Figure 4: Two-step procedure to solve inverse problem by using network inversion.

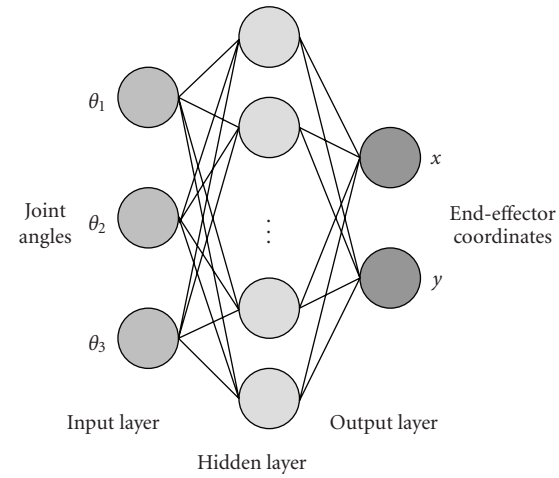


Figure 5: Network architecture.

In this paper, we introduce the regularization method to reduce the ill-posedness of the problems in terms of the uniqueness and stability of their solutions. For the uniqueness problem, we examine the addition of a regularization term for conditioning and attempt to selectively estimate a specific solution. We select a solution that meets the requirement by specifying conditions for the joint angles. This method is shown to be effective when the solution for a specific shape of the arm is required to avoid some obstacle. Moreover, we attempt to stabilize the solutions by decreasing the difference between multiple solutions. We consider a regularization term to minimize the difference in solutions to successive data. This method is shown to be effective for estimating joint angles when the robotic arm is successively operated.

The principle of network inversion is the iterative correction of the input so as to minimize the error for a given output. That is, the output error converges to almost zero when it reaches an appropriate input for the given output. Therefore, the output error remains when the input to a given output does not exist. Thus, we can estimate that a solution may not exist by observing decreases in the output error in response to the corrections made to the input by network inversion. As a result, we consider it possible to deal with ill-posedness in terms of the existence of a solution.

5. Simulation

In order to demonstrate the application of the network inversion method to an inverse kinematics problem and its effect on the ill-posedness, we carry out four simulations considering an ill-posed problem with no solution, with a nonunique solution, and with an unstable solution and a well-posed problem. These four simulations aim to cover all situations concerning ill-posedness when the network inversion method is applied to actual inverse kinematics problems.

We examine the inverse estimation of joint angles from the end-effector coordinates of the 3-DOF robot arm, shown in Figure 2(a), using network inversion. First, we train the network by providing the joint angles and end-effector coordinate corresponding to the joint angles as the tutorial input and output, respectively. The training of the network is conducted by the usual error back-propagation method. The inputs are then iteratively updated due to the error between the network outputs and the end-effector coordinate value. Initially, the inputs are set to random values, and the trained weights are assigned to the values obtained during learning. In this manner, the joint angles are estimated from the provided end-effector coordinates.

The network architecture used in the simulation is shown in Figure 5, and the network parameters are listed in Table 1.

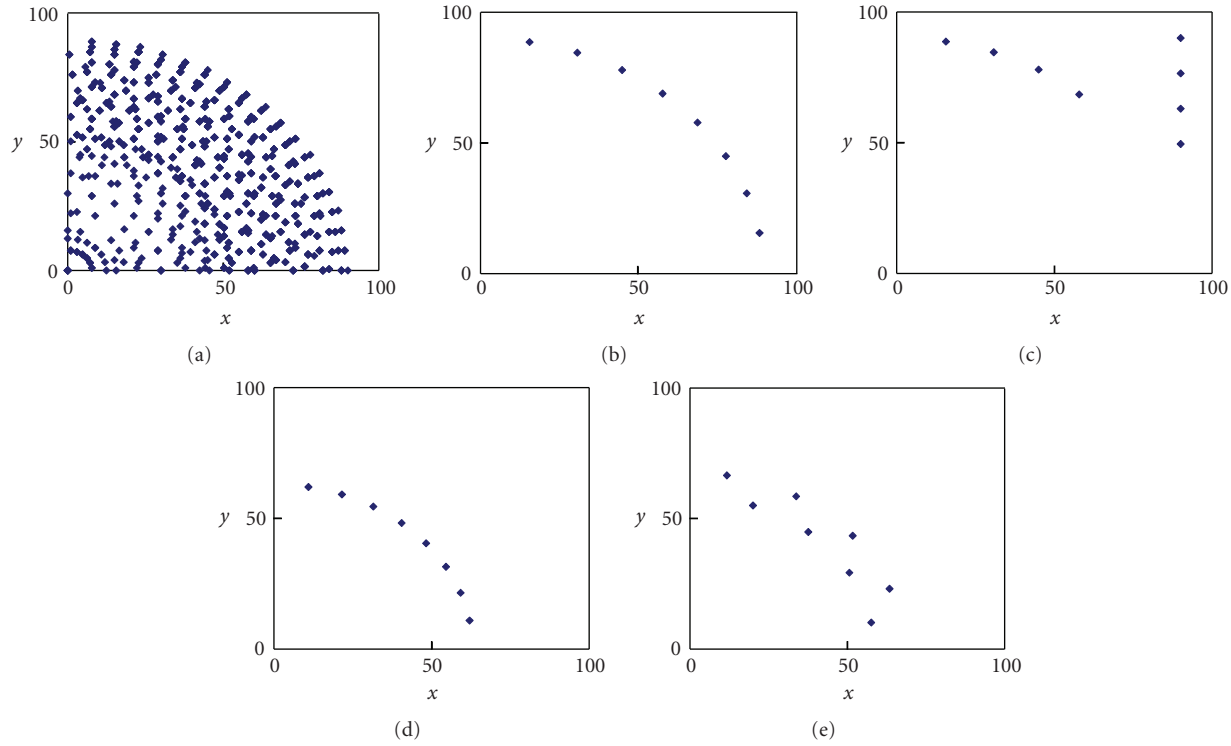


Figure 6: Distribution of end-effector coordinates (a) for training data and for test data of (b) well-posed problem and a problem with (c) nonunique solution, (d) no solution, and (e) unstable solution.

Because there are three joint angles (θ_1, θ_2, and θ_3) and two end-effector coordinates (x and y) for the arm, we use a network with three inputs and two outputs. The hidden neurons are decided to be 10 units by trial and error. The input and output values of the network are normalized to the range [0.0, 1.0] by the maximum and minimum values of the joint angles and coordinates.

First, we prepare the training data by calculating the end-effector coordinates for the three joint angles. We obtain 1136 data points in the tutorial training dataset by varying each of the joint angles θ_1, θ_2, and θ_3 to 7, 19, and 19 different values, respectively, and by selecting only positive coordinates (x, y). The length of each arm is 30 cm. The distribution of the end-effector coordinates in the training data is shown in Figure 6(a). The training is continued until the mean square error decreases to less than 0.001. The training rate ε_t is set to 0.01. The actual training is completed after approximately 8000 epochs.

In the inverse estimation, the joint angles are estimated from the given end-effector coordinates by means of network inversion. In order to confirm the effectiveness of the network inversion to the ill-posed inverse kinematics, we set each of the end-effector coordinates for well-posed, nonexistent, nonunique, and unstable solutions. The end-effector coordinates shown in Figures 6(b), 6(c), 6(d), and 6(e) correspond to well-posed, nonexistent, nonunique, and unstable solution data, respectively. The inverse estimation is continued until the mean square error decreases to less than 0.0001. The inverse estimation rate ε_e is set to 0.001. In order to confirm the effect of the regularization method, the initial input values are set to the constant value of 0.5 in this simulation, although they are generally set to random values for network inversion.

Table 2: Estimated joint angles, calculated end-effecter coordinates, and mean square errors for well-posed problem.

	Joint angles (°) (θ_1, θ_2, θ_3)	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(15.56, 163.07, 189.15)	(88.61, 11.40)	4.23
2	(38.55, 153.58, 175.06)	(82.56, 28.75)	2.86
3	(42.80, 166.39, 173.72)	(75.84, 46.69)	2.70
4	(44.49, 176.89, 178.63)	(66.89, 60.14)	3.07
5	(45.67, 184.19, 181.96)	(58.85, 67.98)	1.39
6	(47.29, 194.13, 186.55)	(45.96, 76.19)	2.00
7	(64.68, 190.92, 168.88)	(33.22, 83.25)	2.77
8	(73.66, 194.68, 169.71)	(15.52, 88.13)	0.51

5.1. Results for Well-Posed Problem. In order to demonstrate the fundamental ability of inverse estimation by network inversion, we simulate a well-posed problem. We use the abovementioned training data. The eight points of the end-effector coordinates that have unique solutions and are shown in Figure 6(b) are used as data for inverse estimation. Regularization is not used for this simulation. The estimated results for the joint angles are shown in Figure 7 and Table 2. The end-effector coordinates calculated from the estimated

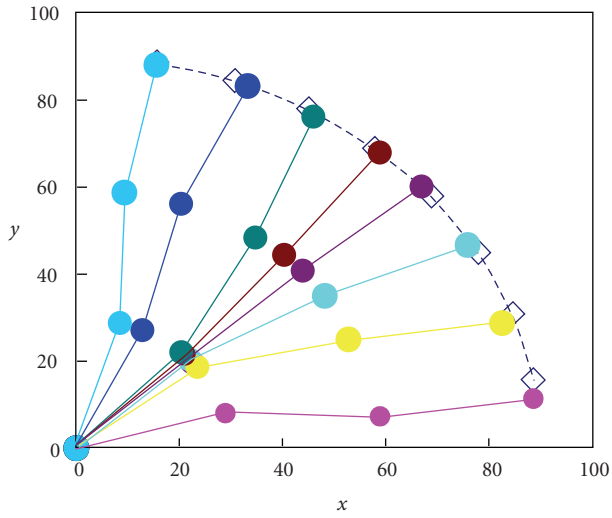


Figure 7: Graph showing the coordinates of robot arm whose joint angles are estimated by network inversion for well-posed problem.

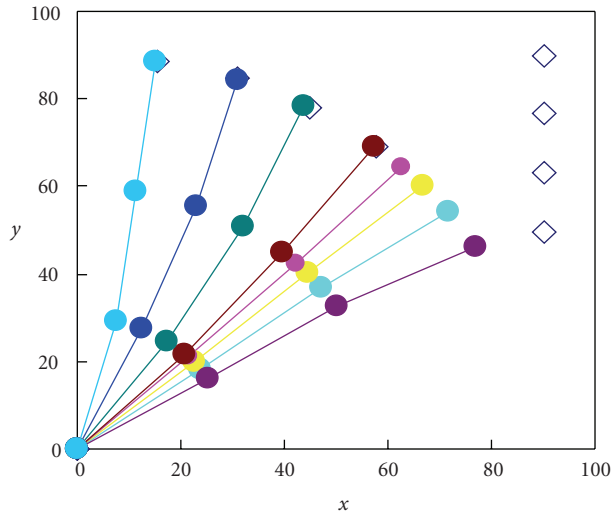


Figure 8: Graph showing the coordinates of robot arm whose joint angles are estimated by network inversion for ill-posed problem with no solution.

joint angles closely corresponded to the given end-effector coordinates shown in Figure 7. Table 2 shows that the error between the estimated and given coordinates is small. According to these results, the joint angles that realized the given end-effector coordinates are accurately estimated. We confirm that the joint angles are correctly estimated from the given end-effector coordinates by network inversion in the case of the well-posed problem.

5.2. Results for Ill-Posed Problem with No Solution. We consider measures to reduce the ill-posedness when no solution exists. When an impossible end-effector coordinate is given, the system is required to estimate the joint angles that realize the end-effector coordinate as much as possible for an approximate or provisional solution. Further, it is necessary to specify that the estimated solution is an approximate one and that a rigorous solution does not exist.

Here, we present the inverse estimation of an approximate solution by means of network inversion when a rigorous solution does not exist. The abovementioned training data are used for training. For inverse estimation, we use data that consist of four possible and four impossible end-effector coordinates, as shown in Figure 6(c). We do not use regularization in this simulation. We consider the transition and the converged values of the output error in the inverse estimation as criteria specifying the nonexistence of a solution.

The estimated joint angles are shown in Figure 8 and Table 3. According to the results, the joint angles are approximately estimated for the impossible end-effector coordinates, while they are correctly estimated for the possible end-effector coordinates. In other words, Figure 8 shows the results of the estimated end-effector coordinates in the possible range for impossible coordinates. Figure 9 shows the transition of the output error in the inverse estimation. We found that the transition and converged value of the output error were obviously different for the impossible

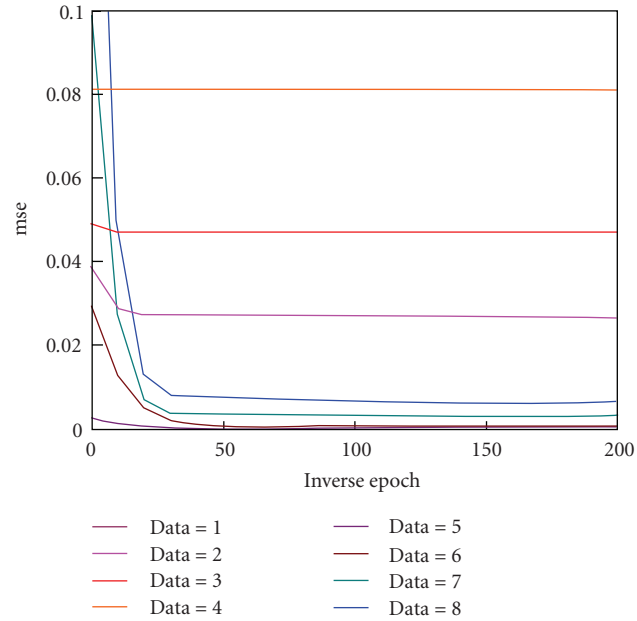


Figure 9: Error curves of inverse estimation for each estimation data.

data points no.1 to no.4 and the possible data points no.5 to no.8. We confirmed that the approximated joint angles for a given end-effector coordinate can be estimated by network inversion when a solution does not exist. Moreover, we confirmed that the nonexistence of a solution can be determined according to the output error of the inverse estimation.

5.3. Results for Ill-Posed Problem with Nonunique Solution. We consider the effect of using regularization as a selection method when a solution is not unique. Here, we examine the maximization and minimization of the joint angles.

Table 3: Estimated joint angles, calculated end-effector coordinates, and errors for ill-posed problem with no solution.

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(32.20, 181.72, 172.81)	(77.07, 46.23)	13.34
2	(37.13, 181.99, 175.97)	(71.74, 54.29)	20.23
3	(41.25, 181.77, 178.91)	(66.81, 60.29)	28.29
4	(44.54, 181.36, 181.13)	(62.71, 64.53)	37.33
5	(45.88, 185.27, 182.78)	(57.37, 69.15)	0.52
6	(57.85, 185.84, 186.30)	(43.69, 78.30)	1.36
7	(65.69, 183.78, 185.06)	(30.87, 84.35)	0.24
8	(75.49, 187.07, 180.73)	(14.91, 88.59)	0.72

For example, for the arm shape, we assumed the restraint condition as a case where an obstacle exists. The training dataset is the same, as described above. For inverse estimation, we use data consisting of eight end-effector coordinates, as shown in Figure 6(d).

We examine two types of regularization terms, where the regularization term $F(x)$ of (4) is set as

$$F_1(x) = x_1^2 + (x_2 - 0.5)^2,$$
$$F_2(x) = (1 - x_1)^2 + (x_2 - 0.5)^2, \tag{6}$$

where $\mathbf{x}$ represents a normalized joint angle. $F_1(x)$ forces the angles θ_1 and θ_2 to 90° and 180°, respectively, while $F_2(x)$ forces the angles θ_1 and θ_2 to 0° and 180°, respectively. Moreover, we use a dynamic regularization method that reduces the regularization coefficient according to the updated input

$$\lambda(t) = \lambda_0\left(1 - \frac{t}{T}\right), \tag{7}$$

where the number of input updates, maximum inverse estimation epoch, and initial regularization coefficient are expressed as t, $T = 10000$, and $\lambda_0 = 0.001$, respectively. The inverse estimation is carried out for two regularization methods by using $F_1(x)$ and $F_2(x)$ of (6).

The estimated joint angles are shown in Figure 10 and Table 4. According to the results shown in Figure 10(a), we found that the estimated joint angle θ_1 became maximum for all end-effector coordinates when the regularization term $F_1(x)$ was added. Because of the ratio between the regularization coefficient and the inverse estimation rate, we found that the regularization term only had a slight effect on the estimated joint angle θ_2. Similarly, we found that the estimated joint angle θ_1 was minimized to all end-effector coordinates when the regularization term $F_2(x)$ was added, as shown in Figure 10(b).

From the above-mentioned results, we confirmed that the joint angles for the shape of the robot arm could be estimated on the basis of specific conditions from the given end-effector coordinates by using network inversion with regularization. In other words, an arbitrary shape can be estimated using network inversion with regularization by providing conditions for the joint angles when the joint angles of a 3-DOF robot arm are inversely estimated.

Table 4: Estimated joint angles, calculated end-effector coordinates, and errors with (a) regularization $F_1(x)$ and (b) regularization $F_2(x)$, for ill-posed problem of nonuniqueness.

(a)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(0.00, 145.90, 284.08)	(65.11, 11.37)	3.10
2	(0.00, 170.03, 285.01)	(56.91, 24.69)	3.89
3	(0.00, 182.40, 282.07)	(52.48, 30.30)	2.40
4	(0.00, 199.61, 284.23)	(41.55, 34.99)	8.68
5	(0.00, 217.91, 264.82)	(37.45, 43.67)	5.51
6	(0.00, 234.76, 245.68)	(32.11, 50.37)	4.23
7	(0.00, 252.69, 225.30)	(24.85, 55.13)	5.24
8	(0.00, 281.08, 189.01)	(13.93, 57.62)	5.34

(b)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(90.00, 85.51, 152.49)	(55.35, 11.76)	6.74
2	(90.00, 106.39, 132.56)	(54.48, 23.00)	4.94
3	(90.00, 123.27, 117.75)	(51.33, 31.93)	3.26
4	(90.00, 139.97, 103.46)	(46.13, 39.56	2.33
5	(90.00, 158.16, 89.18)	(38.85, 46.29)	2.57
6	(90.00, 177.82, 87.58)	(31.04, 57.57)	3.04
7	(90.00, 184.35, 90.19)	(27.63, 62.29)	6.82
8	(90.00, 121.83, 290.82)	(1.64, 64.02)	9.51

5.4. Results for Ill-Posed Problem of Instability. We consider the effect of using regularization as a stabilization method when a solution is instable. Here, we attempt to stabilize the solution by assuming that the data for estimation is successively provided and impose the restraint condition between data. This method effectively estimates the joint angles for a given orbit of the end-effector coordinates by the point-to-point method. The above-mentioned training data are used in this simulation. We use the data shown in Figure 6(e) for the inverse estimation; the data are created by adding a disturbance to the amplitude of 10% for every eight points shown in Figure 6(d).

The regularization term $F_3(x)$ of (4) isset as

$$F_3(x) = \|x(n) - x(n-1)\|^2. \tag{8}$$

This regularization term minimizes the difference between the estimation data. We also use dynamic regularization as expressed in (6) and use the same parameters as those mentioned in the previous subsection.

The estimated results of the joint angles with and without the regularization term are shown in Figures 11(a) and 11(b), respectively. The estimated values are shown in Table 5. In Table 5, the mean square error of the end-effector coordinates represents the difference between the

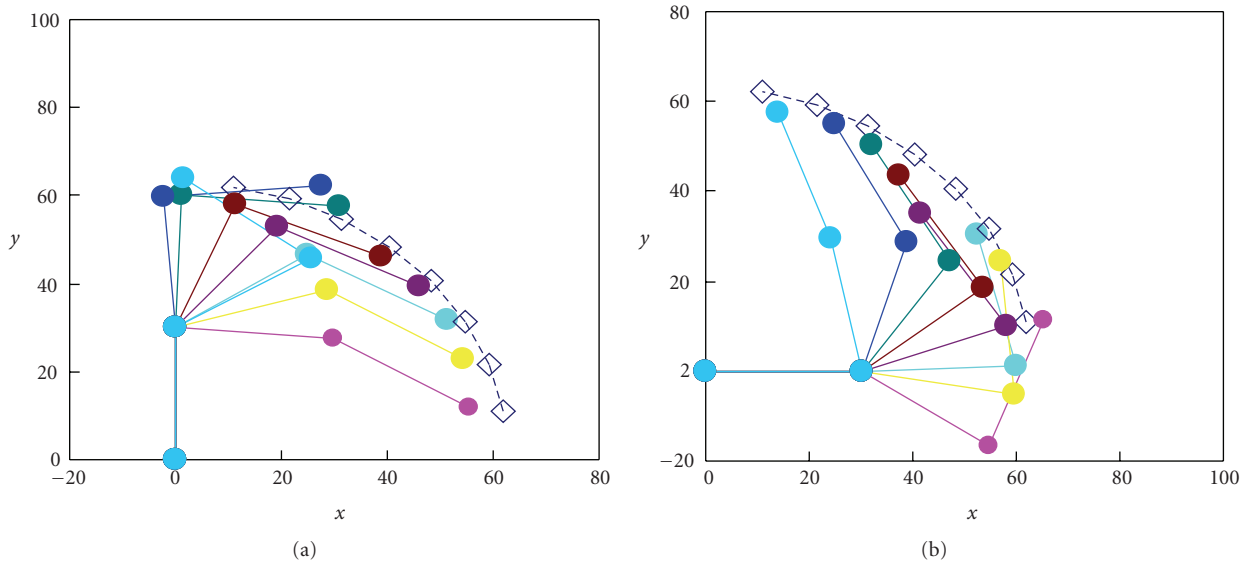


FIGURE 10: Graphs showing the coordinates of robot arm whose joint angles are estimated by network inversion with (a) regularization $F_1(\mathbf{x})$ and (b) regularization $F_1(\mathbf{x})$, for ill-posed problem with nonunique solution.

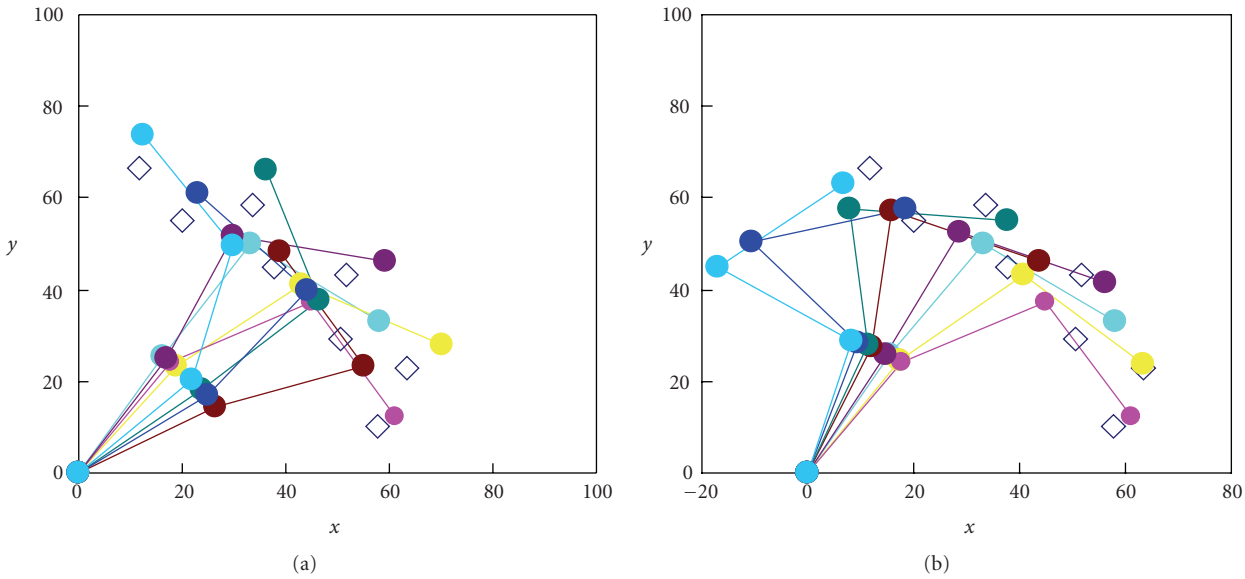


FIGURE 11: Graphs showing the coordinates of robot arm whose joint angles are estimated by network inversion (a) with regularization $F_3(x)$ and (b) without regularization for ill-posed problem of instability.

estimated and correct values without disturbance. According to the results obtained without regularization (Figure 11(a)), the difference between the estimated and correct joint angles were large and unstable. In contrast, with regularization, we found that the fluctuation of the joint angles became small and that they could be estimated with stability (Figure 11(b)). The large variance in the mean square error is clearly shown in Table 5(a), while the stable mean square error is shown in Table 5(b). These results confirm that we could estimate stable joint angles from given uneven end-effector coordinates through network inversion with regularization.

6. Conclusion

In this paper, we applied network inversion to the inverse kinematics problem of estimating the joint angles of a robot arm from the given end-effector coordinates. We then examined different aspects of ill-posedness in the inverse kinematic problem for a robot arm. We showed that network inversion could detect the nonexistence of a solution and estimate an approximate or provisional solution. We also confirmed that network inversion with regularization provided a unique or stable solution to the ill-posed problems. Through our results, we demonstrated

Table 5: Estimated joint angles, calculated end-effector coordinates, and errors(a) with regularization $F_3(x)$ and (b) without regularization for ill-posed problem of instability.

(a)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(53.67, 152.29, 96.80)	(60.98, 12.08)	1.56
2	(50.75, 165.75, 117.70)	(70.11, 28.02)	12.69
3	(57.45, 178.11, 90.36)	(57.96, 33.22)	3.81
4	(55.85, 188.36, 104.63)	(59.33, 46.03)	12.38
5	(28.37, 168.91, 285.48)	(38.81, 48.39)	1.69
6	(37.31, 183.61, 248.99)	(36.31, 66.04)	12.45
7	(34.15, 195.49, 265.50)	(22.99, 60.86)	2.20
8	(42.89, 211.92, 230.95)	(12.31, 73.71)	11.75

(b)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(53.67, 152.29, 96.80)	(60.98, 12.08)	1.56
2	(55.03, 163.23, 101.18)	(63.54, 23.65)	4.82
3	(59.11, 174.72, 92.30)	(58.02, 33.24)	3.87
4	(60.25, 182.58, 94.85)	(56.34, 41.34)	8.12
5	(66.21, 196.54, 75.65)	(43.78, 46.17)	3.89
6	(67.99, 208.19, 78.62)	(37.89, 54.92)	6.40
7	(71.44, 241.13, 61.29)	(18.38, 57.72)	3.50
8	(74.08, 253.47, 69.53)	(6.84, 63.03)	4.22

the effectiveness of applying network inversion to ill-posed inverse kinematic problems. We consider that the results suggest the applicability of network inversion to general ill-posed inverse problems.

In the future, we intend to search for a more effective regularization method for actual inverse kinematics problems by examining different types of regularization methods. In addition, we intend to examine the inverse estimation of joint angles for a multi-DOF robot arm.

References

[1] J. J. Craig, *Introduction to Robotics Mechanics and Control*, Addison-Wesley, Reading, Mass, USA, 1989.

[2] C. W. Groetsch, *Inverse Problems in the Mathematical Sciences*, Friedrich Vieweg & Sohn, 1993.

[3] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biological Cybernetics*, vol. 57, no. 3, pp. 169–185, 1987.

[4] W. T. Miller, R. S. Sutton, and P. J. Werbos, *Neural Networks for Control*, MIT Press, Cambridge, Mass, USA, 1990.

[5] E. Oyama and S. Tachi, "A study of human hand position control learning–output feedback inverse model," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 1434–1443, 1991.

[6] A. Linden and J. Kindermann, "Inversion of multilayer nets," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '89)*, pp. 425–430, June 1989.

[7] B.-L. Lu, H. Kita, and Y. Nishikawa, "Inverting feedforward neural networks using linear and nonlinear programming," *IEEE Transactions on Neural Networks*, vol. 10, no. 6, pp. 1271–1290, 1999.

[8] C. A. Jensen, R. D. Reed, R. J. Marks II et al., "Inversion of feedforward neural networks: algorithms and applications," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1536–1549, 1999.

[9] W. R. Murray, C. T. Heg, and C. M. Pohlhammer, "Iterative inversion of a neural network for estimating the location of a planar object," in *Proceedings of the World Congress on Neural Networks*, vol. 3, pp. 188–193, 1993.

[10] I. Valova, K. Kameyama, and Y. Kosugi, "Image decomposition by answer-in-weights neural network," *IEICE Transactions on Information and Systems*, vol. E78-D, no. 9, pp. 1221–1224, 1995.

[11] J. Wang, Q. Hu, and D. Jiang, "A Lagrangian network for kinematic control of redundant robot manipulators," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1123–1132, 1999.

[12] T. Ogawa, H. Matsuura, and H. Kanada, "A solution of inverse kinematics of robot arm using network inversion," in *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, pp. 858–862, November 2005.

[13] N. Sekiguchi, T. Ogawa, and H. Kanada, "Inverse estimation of joint angles of robot arm by network inversion," in *Proceedings of the Society of Instrument and Control Engineers Annual Conference (SICE '07)*, pp. 991–995, September 2007.

[14] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*, Winstion and Sons, 1977.

[15] Y. P. Petrov and V. S. Sizikov, *Well-Posed, Ill-Posed, and Intermediate Problems with Application*, Koninklijke Brill NV, 2005.

[16] Y. Kosugi and K. Kameyama, "Inverse use of BP net in answer-in-weights scheme for arithmetic calculations," in *Proceedings of the World Congress on Neural Networks*, vol. 3, pp. 462–465, 1993.

[17] T. Ogawa, Y. Kosugi, and H. Kanada, "Neural network based solution to inverse problems," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 2471–2476, May 1998.

[18] T. Poggio and F. Girosi, "Regularization algorithms for learning that are equivalent to multilayer networks," *Science*, vol. 247, no. 4945, pp. 978–982, 1990.

[19] B. Lu and K. Ito, "Regularization of inverse kinematics for redundant manipulators using neural network inversions," in *Proceedings of IEEE International Conference on Neural Networks*, pp. 2726–2731, December 1995.

Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 523757, 8 pages
doi:10.1155/2010/523757

Review Article

Deep Blue Cannot Play Checkers: The Need for Generalized Intelligence for Mobile Robots

Troy D. Kelley[1] and Lyle N. Long[2]

[1] *U.S. Army Research Laboratory, AMSRD-ARL-HR-SE, Aberdeen Proving Ground, MD 21005-5425, USA*
[2] *Department of Engineering and Mathematics, The Pennsylvania State University, University Park, PA 16802, USA*

Correspondence should be addressed to Lyle N. Long, lnl@psu.edu

Received 7 November 2009; Accepted 20 March 2010

Academic Editor: Noriyasu Homma

Generalized intelligence is much more difficult than originally anticipated when Artificial Intelligence (AI) was first introduced in the early 1960s. Deep Blue, the chess playing supercomputer, was developed to defeat the top rated human chess player and successfully did so by defeating Gary Kasporov in 1997. However, Deep Blue only played chess; it did not play checkers, or any other games. Other examples of AI programs which learned and played games were successful at specific tasks, but generalizing the learned behavior to other domains was not attempted. So the question remains: Why is generalized intelligence so difficult? If complex tasks require a significant amount of development, time and task generalization is not easily accomplished, then a significant amount of effort is going to be required to develop an intelligent system. This approach will require a system of systems approach that uses many AI techniques: neural networks, fuzzy logic, and cognitive architectures.

1. Introduction

The problem of generalized intelligence has been plaguing the field of Artificial Intelligence (AI) since its inception in the early 1960s. Researchers in the area of AI had once hoped that a generalized intelligent system would be able to "grease a car or read Shakespeare; tell a joke or play office politics" [1]. Instead, AI researchers found that the learned rules and hard coded knowledge developed in order to solve a specific task was not likely to be transferable to other tasks. The International Business Machines (IBM) Corporation was eventually able to develop a computer program which defeated the most highly ranked chess player in the world [2]. However, IBM's computer only accomplished one task—playing chess. It did not play backgammon or checkers. It did not read Shakespeare or grease a car. The real problem for AI turned out to be the brittleness of behavior, or the lack of generalization across behaviors. In other words—how does playing chess help with reading Shakespeare? Or how does learning to tell a joke help with learning office politics? It could be that very little knowledge can be transferred from playing chess to reading Shakespeare. If this is the case then how can generalized intelligence ever be realized? This paper will examine the different aspects of generalization and whether it can be performed successfully by future computer programs or robots.

Generalized intelligence is especially important for mobile robots, whether they are in the air, on the ground, in space, or in water. There is a need for these robots to be both intelligent and autonomous (and they may become conscious too [3]). Current mobile robots have very little intelligence or autonomy. Intelligence would allow the robots to perceive their environment, plan their activities, solve problems, and learn from experience [4]. Autonomy would allow them to survive in a real-world environment without external control. Intelligence and autonomy will require a complex system of systems approach that is highly interconnected, much like the human brain and central nervous system. The sensor input data will need to be processed in a hierarchical manner from low-level features to complex objects. In addition, learning will be crucial. It will not be possible to program these systems for all possible circumstances that they will encounter. It will also not be physically possible to write all the needed software and logic rules. They

will need to be trained and nurtured as are human infants and children, and they need to learn from experience.

2. Symbolic and Subsymbolic Generalization

The chess playing computer program developed by IBM grew out of the symbolic tradition of AI. The symbolic tradition emphasized symbolic manipulations of information for problem solving, for example, the blocks world problem, the water jug problem, and so forth. A symbolic system uses localized representations of knowledge for problem solving (i.e., concepts are represented in one place). Mathematics and language are symbolic systems of knowledge representation (this assumption has been challenged by neural network researchers [5]). Symbolic systems of knowledge representation are in contrast to subsymbolic or distributed representations of knowledge. Subsymbolic representations of knowledge are not localized (i.e., concepts can be represented across a collection of nodes as weights). Neural networks and computer vision algorithms are subsymbolic representations of knowledge.

Subsymbolic systems were once hoped to be a solution to complex recognition tasks [5] and indeed they are capable of recognizing a wide variety of functions. Neural networks are capable of perceiving and classifying the noisy outside world given the correct topology and enough training examples. However, neural networks still have a problem with generalization. Within neural network research the generalization problem is defined as the over-fitting or overtraining problem. Neural networks can be trained to recognize training data; however, if the training is conducted with too many iterations, then the network will *only* be able to recognize the training data. Specifically, the network will not be able to generalize to data from outside the training set—it will be "over trained" or "over fit" to the original training set. And while recognition of previously trained data is an important component of an AI system, generalization is more important for a robust intelligence system. A neural network which has succumbed to over fitting will not be able to generalize outside the original training set. This can be especially problematic in real-world dynamic environments where the outside world is less predictable.

This issue is also a problem for the neural network developer since a network's architecture can be "under fit" as well as "over fit." In the "under fit" condition, the network will not be able to recognize training data without a sufficient number of training iterations. Moreover, the problem of over fitting a neural network is related to a number of structural factors in the network (i.e., network size, number of hidden layers, number of training examples) in addition to the point of learning convergence. These factors can make the autonomous selection of a cut-off point for training difficult to determine since the determination is related to many other architectural and data factors. The problem has become harder than expected within the neural network community [6] and this will continue to limit the subsymbolic generalization of traditional neural network architectures.

Once a network has been trained, it could be easily assumed that new training examples could be added to the original training set and this would overcome the generalization problem. However, this leads to the catastrophic forgetting problem [7]. Once a neural network has been trained on a specific set of training examples, training the net on new examples without including the older examples leads to the loss of the older examples. The network "forgets" the previously learned material. For example, one could easily train a neural network to recognize the alphabet, but once it is trained to do that, it would be difficult to then have it also learn to recognize numbers without human intervention. This greatly affects the generalization of the network since it becomes difficult to add new information to the network. Or, a neural network for a mobile robot that is trained to recognize doors would be hard to expand to include windows.

Newer neural network architectures [8], however, have overcome some of these problems and there are solutions to the problem catastrophic forgetting with traditional neural networks [9]. A newer promising architecture for overcoming problems of catastrophic forgetting is the Adaptive Resonance Theory (ART) [10]. ART was specifically developed to overcome problems of associated with catastrophic forgetting by adding additional functionality to a typical neural network. Specifically, ART uses a feedback mechanism between different layers of the network which allows the network to automatically switch between stable and flexible modes of operation. Additionally, ART has a competitive network, which, based on some criteria, allows nodes in the network to compete and select a winner in classification tasks. These additional functional mechanisms allow for online continual learning without the destruction of previous learned information. Additionally, they allow for learning based on a few number of instances (so called one-trial learning) which is one of the defining hallmarks of human learning.

Another type of relatively new neural networks is spiking neural networks [11]. These are biologically plausible time-dependent networks that are especially good for forming episodic memories and for sensor processing. These more closely model human neurons and synapses using nonlinear differential equations for membrane voltages and time-dependent Hebbian learning for synapses. These have been implemented to learn object recognition tasks. As in the human brain, one could also implement neurogenesis and synaptogenesis to allow the network to keep learning without forgetting the existing information.

It would appear then that newer types of subsymbolic architectures will be needed to allow for increased generalization over and above traditional neural networks. These newer architectures appear to be the only solutions to the problems of subsymbolic generalization (see Table 1).

3. Subsumptive Architectures

Subsumptive architectures [12] were developed as an answer to the brittleness and lack of generalization found in traditional symbolic AI systems. In a jab at symbolic systems, Brooks [13] titled his seminal work on subsumptive architectures, "Elephants do not play chess" implying that

Table 1: Symbolic and subsymbolic distinctions.

	Symbolic	Subsymbolic
Data	Discrete Symbolic	Distributed Metric
Strengths	Readable Logical	Fault tolerant Adaptive
Weakness	Frame problem	Catastrophic Forgetting

elephants don't need symbolic manipulation. In a subsumptive architecture, explicit representations of the world or the problem space are intentionally avoided to allow for relatively simple generalizations across environments. Rules are used to represent a problem space, much like a symbolic architecture, but these rules tend to be extremely simple (i.e., move forward, turn left, move to light). These rules are embodied within agents and the rules compete for behavioral priority. For example, "turn left" would compete with "turn right," and would only execute if the threshold for the execution of "turn right" were, for some reason, lower than the threshold for the execution of "turn left." Subsumptive architectures do not represent the world as a model, because, as the subsumptive researchers explain, "the world is its own best model." The logic here is—why go about memorizing previous experiences and creating and updating a world model when one only has to look at the world to determine how to behave?

The answer to this question is that the world does not hold all the answers to all the questions and some questions are too abstract to represent with simple if-then statements. Learning and memory are very important to generalized intelligence. The problem with subsumptive architectures is that they work well for some simple behaviors, but more complex behavior is more difficult to represent. A subsumptive architecture would have difficulty playing a good game of chess. It might also produce mobile robots that get stuck in environments such cul-de-sacs. It is possible that a subsumptive architecture could play a game of chess, and perhaps get better at playing chess, but not reach a level of a grand master without formal symbolic representations. It is this very lack of symbolic complexity that leads to problems with subsumptive architectures. A reactive system might be suitable for lower level reactive behaviors of the type exhibited by an elephant, but what if the task domain required the symbolic manipulations necessary to play chess?

4. Cognitive Architectures

Cognitive architectures are well-known symbolic AI approaches that attempt to mimic human cognitive abilities via rule-based processing. Examples of these are Soar [14], ACT-R [15], and EPIC [16]. Some of these have been implemented on mobile robots [4, 17].

Some generalization can be accomplished using simple, rule-based, symbolic representations of knowledge. They can use general rule sets in order to encompass a wide variety of structural mappings and thus allow for more robust decision making. For example, a door can be defined as an opening that leads into, or out of, a predefined space. Note that this definition says nothing about the specific properties of the door or even the specific properties of the space. Having such a broad, or generalized, definition of a "door" allows for a significant amount of generalization that is not specific to the properties of a door. For instance, using our previous definition, if a garage can be defined as a pre defined space, then any door leading into or out of the garage could be defined as also being a "door." So the definition applies to a garage door—even though a typical garage door has structural properties which are very different from a typical household door (i.e., a garage door is very large, opens horizontally, and typically does not include a doorknob while a household door is smaller, opens vertically, and usually includes a doorknob). The specification of an abstract or general rule can enable a system to perform symbolic generalizations across the underlying structural similarities of a problem space.

However, one problem with symbolic generalization techniques, especially those which generate a large number of possible actions, is the frame problem [18]. The frame problem can generally be thought of as having to represent every possibility symbolically. The discovery of the frame problem was one of the essential motivating factors in the development of the previously mentioned subsumptive architectures [12]. Interestingly, the frame problem goes away if development concentrates on task-specific behavior, which may be one reason why task-specific development has dominated AI research. It was an effort to ignore or overcome the frame problem. Task specific behavior, by very definition, is constrained, so developers did not have to worry about a wide variety of possible behaviors.

While the cognitive architectures are powerful and useful, they too have their limitations. A purely symbolic approach is difficult to use for processing detailed sensor data, for example. It would also be difficult to use it for sensitive motor output tasks. In addition, it is quite difficult to add learning to cognitive architectures, which is essential for future mobile robots. And unless the systems can develop their own rules, they are susceptible to the frame problem as well.

5. Hybrid Approach to Complex Cognition

Both symbolic and subsymbolic data and data processing have their advantages and disadvantages for mobile robots, and a hybrid approach using both are really required.

Human cognitive systems, and some intelligent animals, exhibit within their neurological functionality indications of *both* symbolic and subsymbolic representations of knowledge [19]. This is a *functional* distinction and not a cellular or neurological distinction. Specifically, the cerebral cortex and frontal lobes of the human cognitive system are capable of accomplishing symbolic manipulations of information while other parts of the system appear to operate—functionally—in a more distributed manner. However, the cerebral cortex and frontal lobes are probably neurologically distributed

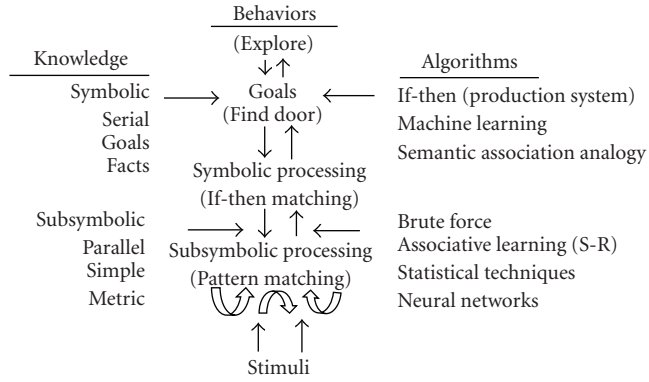


Figure 1: SS-RICS notional data flow with the symbolic and subsymbolic distinctions [3].

systems—at least in terms of memory components [20], but in this paper, we are making a distinction between the functional aspects not the structural aspects. We are making this distinction between different types of knowledge representation (i.e., symbolic versus subsymbolic) because the type of knowledge representation used within an intelligent system will affect the ways in which the information can be manipulated.

There are several examples of hybrid systems: SS-RICS [17], CRS [4], and CLARION [21].

SS-RICS is largely based on the Adaptive Control of Thought-Rational (ACT-R) (see Figure 1). It implements a production system architecture similar to ACT-R with decay algorithms that affect memories. Like ACT-R, it is production system architecture at the highest level, which we are using to mimic the functionality of human working memory. SS-RICS also includes a subsymbolic system at the lower levels, which mimic iconic short-term memory and perception. The production system within SS-RICS is composed of rules and goals. SS-RICS can also access "concepts" which are long term facts; similar to declarative memories within ACT-R, but within SS-RICS they are considered long term memories (i.e., memories which do not decay). SS-RICS can also generate productions automatically in order to generalize [17]. SS-RICS has two types of processes for the symbolic generation of new productions, top-down learning and bottom-up learning. It is envisioned that SS-RICS will have other types of symbolic generation mechanisms primarily because there seem to be a number of other techniques already available [22].

CRS uses Soar combined with subsymbolic processing (e.g., computer vision systems). In this case, Soar is coupled to Java software for input and output processing. The sensor inputs and motor control outputs are controlled in Java while the symbolic data is stored in Soar. CRS has been used on both wheeled and legged robots [4]. It has also been used with several types of sensor input systems (vision, sonar, GPS, compass, touch, etc.). The approach has been very effective, and it is fairly easy to add additional sensors, rules, or output devices.

The Connectionist Learning with Adaptive Rule Induction ON-line, or CLARION cognitive architecture is designed to capture the interaction between subsymbolic and symbolic processes, or as the CLARION developers say, the distinction between implicit and explicit knowledge. CLARION uses Q learning, or reinforcement learning, at subsymbolic levels. Additionally, the architecture uses rule extraction algorithms at the symbolic level to develop links between subsymbolic and symbolic information.

Generalized symbolic representations work well with real-world data as compared to subsymbolic and statistical representations. In a recent exercise at the National Institutes of Standards and Technology (NIST) using SS-RICS (February, 2009) we found that a symbolic representation of the intersections in a maze was just as useful as neural network representations or statistical representations. We essentially used "primitives" to define a small core set of examples which were predefined by the programmer. The use of primitives seems to work well as an overall strategy for symbolic generalization. This is essentially the same strategy as the use of "scripts" [23] or "frames" [24] where abstract symbolic representations are used to represent a variety of problem situations (e.g., restaurants). This approach seems to work well for symbolic representations of knowledge.

6. Human Generalization

In order for humans to successfully accomplish a generalized learning procedure, a number of complex operations or components are required. Each one of the components within the generalization process can be difficult for humans to accomplish effectively. Researchers [25] have identified four major components of the analogy or generalization processes.

(1) The retrieval or selection of a plausibly useful source analog.

(2) Mapping.

(3) Analogical inference or transfer.

(4) Subsequent learning.

Gick and Holyoak [26, 27] found that in the absence of clear relationships between two problem spaces useful analogies are often not discovered by problem solvers. Additionally, Holyoak and Koh [25] found that subjects are

frequently fixated on the salient surface regularities of a problem space and not the underlying structural features, making the selection of a useful source analog even more problematic. Moreover, mapping can be a difficult process as well, because it is not only the selection of a source analog but also selecting which aspects of the source are important [28]. Additionally, Novick [29] found instances of negative transfer—where misleading surface similarities were used to create the analogy. So it would appear from the human behavioral data that human generalization is not easily accomplished by human problem solvers.

To complicate matters, it would also appear that mastering a task to the level of an expert is also quite difficult. Ericsson [30] has found that in order to develop expertise in a complex task, a human subject needs to have a considerable amount of time doing the task. This amount of time, according to Ericsson, is roughly estimated to be 10,000 hours (roughly 5 years of full-time effort). Even human experts who were once considered to be child prodigies, like Wolfgang Mozart or Bobby Fisher, still required 10,000 hours before they were able to perform at a level that was considered a "master" level. Additionally, not only does task-specific behavior require a large amount of experience but it also requires "deliberate practice" in order for an expert to fully master a complex task (like playing chess). Ericsson has shown that human masters in a specific task require a large amount of practice before they become proficient at the task and that the practice has to be efficient and useful for a full development of skills. The human data from Ericsson's studies seem to be concurrent with computer software data: that is, in order to develop a system with expertise in one area a significant amount of development time is required. This would imply that task-specific behavior requires a large amount of learning or development time before an autonomous system would be capable of mastering a task.

If generalization is difficult for human subjects and task-specific behavior is time consuming for human subjects then one must only assume that generalization for robotics and autonomous systems will also be difficult. AI researchers have found that the development of complex task-specific behavior requires an enormous amount of hand tuning in order to achieve the desired results and this would seem to apply to human learning as well. Generalization has proved difficult for human subjects and this seems to also be the case for AI systems as well. The two problems are related. The more task-specific behavior that is developed, the more likely a system can generalize to a new environment. Thrun [31] has noted that learning becomes easier when embedded in a life-long context. So, it could be then that the development of task-specific behaviors will help with generalization—but only if the bulk of the behaviors can be applied to a new situation.

7. Hybrid Intelligent Systems for Mobile Robots

Until we can reverse engineer human or animal brains, we will need to use a clever assortment of algorithms in order to build intelligent, autonomous, and possibly conscious mobile robots. We will need to take a system of systems approach. The various sensor inputs (vision, touch, smell, sound, etc.) will each need to be highly refined systems, and these will most likely be subsymbolic systems as they are in humans. We will also need symbolic systems that use fuzzy logic, rule-based approaches, and other AI techniques. And we will also need motor control output systems. There are roughly 600 muscles in the human body, each controlled by the central nervous system.

It is also important to recognize that whether an approach is considered subsymbolic or symbolic really depends on the granularity of the view taken. In the human brain, all the processes are cell or neuron-based. So at that level everything is subsymbolic. Groups of neurons working together, however, can perform tasks that appear symbolic. For example, in the human vision system there are face recognition subsystems. The human brain also has place cell subsystems, and possibly "grandmother cells" subsystems. Another example is the use of fuzzy logic. It has been shown that neural networks can be replaced by fuzzy logic systems, while the neural network would appear to be a subsymbolic system the ultimate function of the system could be replaced by a symbolic system. Since, at the current time, we cannot model all the roughly 10^{11} neurons or 10^{14} synapses (or the wiring diagram that connects them all), we will need to model some of these systems using symbolic approaches. Intelligence is defined as: "a very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly, and learn from experience" [32].

Current cognitive architectures do have the ability to perform some of these tasks, but in order to accomplish many of these the robot will need the abilities discussed earlier regarding analogies and inference. Autonomy is defined as: The ability to operate "in the real-world environment without any form of external control for extended periods of time" [33].

Current mobile robots are usually designed for very specific environments, and even indoor and outdoor robots are designed very differently. The ultimate goal should be to build robots that can adapt to new environments or can be taught and nurtured by humans for the various environments.

It will not be possible to completely wire or program these robots, however. They will need to learn on their own, through traditional machine learning approaches, traditional neural network learning, or neuro/synaptogenesis. Learning will be required at both the symbolic and subsymbolic levels. For the sensor input or motor-control output systems, we will most likely need subsymbolic learning, while at the symbolic level we will need more traditional machine learning approaches. The human brain is possibly the most complex system in the known universe, we must also remember that it takes roughly 20 years to train this system, and roughly 5 years to become an expert on some topic. For robots to reach human levels of intelligence and autonomy the training and learning will need to be extensive. Once it has been trained though, it can then be easily replicated!

We have done some initial work with conceptual generalization and we have found that instance-based recognition of

objects is relatively easy, but that generalized recognition of objects is much more difficult. For example, during robotic navigation, a robot needs to be able to recognize doors as landmarks. Using simple image correlation algorithms (i.e., template matching), we were able to train SS-RICS to recognize specific instances of doors (i.e., the front door, the back door, Sue's door). However, the image correlation algorithm was easily fooled by lighting variations or occlusion. Additionally, the algorithm would sometimes not work with objects that had not been seen before, especially if the new image was significantly different from the template. In addition to using image correlation, we also have used other algorithms for shape definitions. These algorithms extracted the shape of the object being viewed. For instance, a door shape is typically rectangular. However, these algorithms can be problematic if the door is partially occluded. Also, the shape algorithm worked best with a extremely clean data, which is not typical in real world conditions.

We have hypothesized that developing generalized recognition emerges from previous exposures to numerous instance based recognitions. The essential component to this hypothesis is a prior expectation of the input data. In order to account for changes across time due to lighting variations an algorithm is being developed that checks to see if the image being viewed is somehow different from what is expected (correlation). This obviously requires an expectation metric for each object being viewed. If the input data is somehow lower than what is expected, the algorithm first needs to assume that what is being viewed is the previously viewed object but it has somehow changed, and not a new object. This assumption is not always correct, but for the initial implementation of the algorithm, this was the default assumption. Next, the algorithm checks to see what has remained the same across the two images. The features that have remained the same across the two images would represent the essential features of the object. The essential features are what need to be learned to allow for generalization.

8. Conclusions

We have discussed the idea that both symbolic and subsymbolic generalization are difficult for computational systems. We have also discussed the research that shows that generalization is difficult for people as well as computer systems. However, while subsymbolic generalization is difficult for computer systems it is relatively *easy* for people. Human subjects can learn from one example and do not seem to suffer from catastrophic forgetting in the same way that traditional subsymbolic systems do. People can be exposed to just one example of a stimulus and be able to incorporate this new example into a generalized understanding of the stimulus (a.k.a. one-trial learning). One would assume then that the mechanisms of subsymbolic generalization used by humans must somehow be different from the traditional generalizations in subsymbolic systems. However, perhaps the newer subsymbolic algorithms [8–10, 34] will lead to more robust subsymbolic generalizations.

Task specific behaviors, which have been extensively developed by AI systems over the last 40 years, require a large amount of development time and programmer expertise. The continuation of task-specific development must be approached with longer term goals of modularity and reuse in order to facilitate broader generalization. A long development time for task-specific behavior is also reflected in the human data associated with acquisition of expert performance [28]. If task generalization becomes easier when learning is done within the context of extensive previous knowledge [29], then generalization can only proceed once an extensive amount of task-specific behavior has already been developed. This would argue for the continuation of task-specific approaches, but with the need for generalization to occur following the development of task-specific behaviors. This seems to have been ignored by much of the AI research. If we assume the need for an extensive knowledge base, and this is represented symbolically—primitives, scripts, or frames can be used to generalize across problem spaces. But this only applies to symbolic representations of knowledge not subsymbolic representations of knowledge.

Subsumptive architectures are capable of task generalization across simple tasks. However, the very mechanisms that make subsumptive architectures powerful generalization systems within a simple task domain are the same mechanisms that exclude them from being able to generalize across more complex task domains (i.e., lack of a world model). This would argue for a combination of approaches, with a subsumptive architecture being able to generalize over simple reactive tasks coupled with a symbolic system for generalization across more complex tasks. Some researchers have called for the combinations of Bayesian/probabilistic and connectionist approaches [34]. This would facilitate further understanding of the applicability of subsymbolic knowledge structures. In truth, there is probably a continuum of approaches that span the range between symbolic and subsymbolic approaches to include statistical, Bayesian, and chaotic knowledge structures.

Perhaps another approach to accomplishing a generalized intelligent system capable of performing a wide variety of tasks would be something similar to the DARPA Grand Challenge [35] except instead of researching mobility across an open road, the task would be researching generalization across tasks. This would help to emphasize the architectural constraints associated with generalization within each architecture and would force researchers to address how their respective architectures would generalize to different environments. Perhaps the "winning" architecture could then be accepted by researchers and then allowed to learn other tasks on the way to a generalized learning system.

The prospects for a generalized intelligence system are daunting. We have argued in previous papers [3, 36] that complex cognition will require a complex approach Both symbolic and subsymbolic systems appear to have some limitations with regard to generalization, however, newer subsymbolic are capable of addressing some of the limitations associated with subsymbolic learning. Symbolic systems show some promise with generalization given enough prior information and the use of frames or scripts. The careful

combination of symbolic systems with newer subsymbolic structures, along with an extensive experiential knowledge base, all appear to be necessary to solving the generalization challenge. Only then will intelligent and autonomous mobile robots be possible.

References

[1] B. Darrach, "Meet Shaky: the first electronic person," *Life Magazine*, vol. 69, no. 21, pp. 58B–68B, 1970.

[2] D. Goodman and R. Keene, *Man Versus Machine, Kasparov Versus Deep Blue*, H3 Publications, Cambridge, Mass, USA, 1997.

[3] L. N. Long and T. D. Kelley, "Review of consciousness and the possibility of conscious robots," *Journal of Aerospace Computing, Information and Communication*, vol. 7, no. 2, pp. 68–84, 2010.

[4] S. D. Hanford, O. Janrathitikarn, and L. N. Long, "Control of mobile robots using the soar cognitive architecture," *Journal of Aerospace Computing, Information and Communication*, vol. 6, no. 2, pp. 69–71, 2009.

[5] D. E. Rumelhart and J. L. McClelland, "On learning the past tenses of English verbs," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, J. L. McClelland and D. E. Rumelhart, Eds., vol. 11, pp. 216–270, MIT Press, Cambridge, Mass, USA, 1986.

[6] S. Lawrence, C. L. E. E. Giles, and A. H. C. Tsoi, "Lessons in neural network training: overfitting may be harder than expected," in *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI '97)*, pp. 540–545, Menlo Park, Calif, USA, August 1997.

[7] M. McCloskey and N. Cohen, "Catastrophic interference in connectionist networks: the sequential learning problem," in *The Psychology of Learning and Motivation*, G. H. Bower, Ed., vol. 24, pp. 109–164, Academic Press, New York, NY, USA, 1989.

[8] S. E. Fahlman and C. Lebiere, "The cascade-correlation architecture," in *Advances in Neural Information Processing Structures*, D. S. Touretsky, Ed., vol. 2, pp. 524–532, Morgan Kaufmann, San Mateo, Calif, USA, 1990.

[9] A. V. Robins and S. J. R. McCallum, "A robust method for distinguishing between learned and spurious attractors," *Neural Networks*, vol. 17, no. 3, pp. 313–326, 2004.

[10] G. A. Carpenter and S. Grossberg, "Adaptive resonance theory (ART)," *The Handbook of Brain Theory and Neural Networks*, MIT Press, Cambridge, Mass, USA, 2nd edition, 2002.

[11] W. Maass and C. M. Bishop, *Pulsed Neural Networks*, MIT Press, Cambridge, Mass, USA, 1999.

[12] R. Pfeifer and C. Scheier, *Understanding Intelligence*, MIT Press, Cambridge, Mass, USA, 2000.

[13] R. A. Brooks, "Elephants don't play chess," *Robotics and Autonomous Systems*, vol. 6, no. 1-2, pp. 3–15, 1990.

[14] A. Newell, *Soar: A Cognitive Architecture in Perspective*, Harvard University Press, Cambridge, Mass, USA, 1990.

[15] J. R. Anderson and C. Lebiere, *The Atomic Components of Thought*, Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1998.

[16] D. E. Meyer and D. E. Kieras, "A computational theory of executive cognitive processes and multiple-task performance: part 1. Basic mechanisms," *Psychological Review*, vol. 104, no. 1, pp. 3–65, 1997.

[17] T. D. Kelley, E. Avery, L. N. Long, and E. Dimperio, "A hybrid symbolic and sub-symbolic intelligent system for mobile robots," in *InfoTech@Aerospace Conference*, Seattle, Wash, USA, 2009, AIAA 2009-1976.

[18] J. McCarthy and P. J. Hayes, "Some philosophical problems from the standpoint of artificial intelligence," *Machine Intelligence*, vol. 4, pp. 463–502, 1969.

[19] T. D. Kelley, "Symbolic and sub-symbolic representations in computational models of human cognition: what can be learned from biology?" *Theory and Psychology*, vol. 13, no. 6, pp. 847–860, 2003.

[20] P. Dean, "Recapitulation of a theme by Lashley? Comment on Wood's simulated lesion experiment," *Psychological Review*, vol. 87, no. 5, pp. 470–473, 1980.

[21] R. Sun and T. Peterson, "Hybrid learning incorporating neural and symbolic processes," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 727–732, Anchorage, Alaska, USA, June 1998.

[22] T. A. Mitchell, *Machine Learning*, WCB/McGraw Hill, Boston, Mass, USA, 1997.

[23] R. C. Schank, "The structure of episodes in memory," in *Representation and Understanding: Studies in Cognitive Science*, D. Bobrow and A. Collins, Eds., Academic Press, New York, NY, USA, 1975.

[24] M. Minsky, "A framework for representing knowledge," in *Mind Design: Philosophy, Psychology, Artificial Intelligence*, J. Haugeland, Ed., MIT Press, Cambridge, Mass, USA, 1975.

[25] K. J. Holyoak and K. Koh, "Surface and structural similarity in analogical transfer," *Memory and Cognition*, vol. 15, no. 4, pp. 332–340, 1987.

[26] M. L. Gick and K. J. Holyoak, "Analogical problem solving," *Cognitive Psychology*, vol. 12, pp. 306–355, 1980.

[27] M. L. Gick and K. J. Holyoak, "Schema induction and analogical transfer," *Cognitive Psychology*, vol. 15, no. 1, pp. 1–38, 1983.

[28] K. J. Holyoak and P. Thagard, "Analogical mapping by constraint satisfaction," *Cognitive Science*, vol. 13, no. 3, pp. 295–355, 1989.

[29] L. R. Novick, "Analogical transfer, problem similarity, and expertise," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 14, no. 3, pp. 510–520, 1988.

[30] K. A. Ericsson, "The influence of experience and deliberate practice on the development of superior expert performance," in *Cambridge Handbook of Expertise and Expert Performance*, K. A. Ericsson, N. Charness, P. Feltovich, and R. R. Hoffman, Eds., Cambridge University Press, Cambridge, UK, 2006.

[31] S. Thrun, "Lifelong learning algorithms," in *Learning to Learn*, S. Thrun and L. Pratt, Eds., Kluwer Academic Publishers, Boston, Mass, USA, 1998.

[32] L. S. Gottfredson, "Mainstream science on intelligence: an editorial with 52 signatories, history, and bibliography," *Intelligence*, vol. 24, no. 1, pp. 13–23, 1997.

[33] G. A. Bekey, *Autonomous Robots: From Biological Inspiration to Implementation and Control*, MIT Press, Cambridge, Mass, USA, 2005.

[34] A. Gupta and L. N. Long, "Hebbian learning with winner take all for spiking neural networks," in *Proceedings of International Joint Conference on Neural Networks (IJCNN '09)*, pp. 1054–1060, Atlanta, Ga, USA, June 2009.

[35] J. L. McClelland, "The place of modeling in cognitive science," *Topics in Cognitive Science*, vol. 1, no. 1, pp. 11–38, 2009.

[36] T. D. Kelley, "Simulating intelligent behavior requires a complex approach," in *Proceedings of the AAAI Spring Symposium on Between a Rock and a Hard Place: Cognitive Science Principles Meet AI-Hard Problems*, C. Lebiere and R. Wray, Eds., pp. 70–73, AAAI press, Stanford, Calif, USA, March 2006.