International Scholarly Research Network
ISRN Signal Processing
Volume 2011, Article ID 425621, 12 pages
doi:10.5402/2011/425621

Research Article

Camera-Based Motion Recognition for Mobile Interaction

Jari Hannuksela,[1] Mark Barnard,[2] Pekka Sangi,[1] and Janne Heikkilä[1]

[1] *Machine Vision Group, Department of Electrical and Information Engineering, University of Oulu, 90570 Oulu, Finland*
[2] *Centre for Vision, Speech and Signal Processing, Faculty of Engineering and Physical Sciences, University of Surrey, Surrey GU27XH, UK*

Correspondence should be addressed to Jari Hannuksela, jari.hannuksela@ee.oulu.fi

Received 22 February 2011; Accepted 18 March 2011

Academic Editors: L.-L. Wang, N. Younan, and L. Zhang

Multiple built-in cameras and the small size of mobile phones are underexploited assets for creating novel applications that are ideal for pocket size devices, but may not make much sense with laptops. In this paper we present two vision-based methods for the control of mobile user interfaces based on motion tracking and recognition. In the first case the motion is extracted by estimating the movement of the device held in the user's hand. In the second it is produced from tracking the motion of the user's finger in front of the device. In both alternatives sequences of motion are classified using Hidden Markov Models. The results of the classification are filtered using a likelihood ratio and the velocity entropy to reject possibly incorrect sequences. Our hypothesis here is that incorrect measurements are characterised by a higher entropy value for their velocity histogram denoting more random movements by the user. We also show that using the same filtering criteria we can control unsupervised Maximum A Posteriori adaptation. Experiments conducted on a recognition task involving simple control gestures for mobile phones clearly demonstrate the potential usage of our approaches and may provide for ingredients for new user interface designs.

1. Introduction

Designing comfortable user interfaces for mobile phones is a challenging problem, given the limited amount of interaction hardware and the small size of the device. Touch sensitive technology has already enabled new ways for users to interact with handheld devices. Recent touch screens provide an intuitive interface for navigating content but this equipment still imposes some limitations: the user's fingertip size can decrease pointing accuracy, the area of interest can be occluded by fingers, and most importantly the operation area is restricted. Moreover, the amount of functionalities in mobile devices is likely to keep increasing due to the forthcoming 3D user interfaces and applications. Going forward we will see also multiple sensors in portable devices that can enrich the mobile user experience by allowing control through gestures and other types of movement. Studies into alternatives to mobile user interaction have, therefore, become a very active research area in recent years.

Much of the work in mobile interaction has been in direct manipulation interfaces, such as screen navigation by scrolling or pointing and clicking. In particular, it has been shown that different sensors provide viable alternatives to conventional user interaction. For example, tilting interfaces can be implemented with gyroscopes [1] and accelerometers [2]. Using both tilt and buttons, the device itself is used as input for navigating menus and maps. During the operation, only one hand is required for manipulation. Several devices employ a detachable stylus in which interaction is done by tapping the touch screen to activate buttons or menu choices. Interestingly, Apple's products make use of the same technology in a different way. In the iPhone, users are allowed to zoom in and out by performing multiple fingers gestures on the touch screen. In addition, a proximity sensor shuts off the display in certain situations to save battery power, and an accelerometer senses the orientation of the phone and changes the screen accordingly.

On the other hand, many of the current mobile phones have also two cameras built-in, one for capturing high-resolution photography and the other for lower-resolution video telephony. Even the most recent devices have not yet utilised these unique input capabilities enabled by cameras for

purposes other than just photographing. With appropriate computer vision methods, information provided by images allow us to create new self-intuitive user interface concepts. In our work we have focused on what could be described as indirect interfaces, where an abstract shape is recognised, and this is then interpreted as a command by the mobile device.

In this paper we investigate two specific approaches for creating patterns of motion: firstly the estimation of the egomotion of the device itself using the inbuilt camera now available on most mobile devices and also the use of this camera for tracking the motion of an external object, in our case the user's finger. These motion trajectory sequences are then modelled using *Hidden Markov Models* (HMMs). In order to improve the initial, we propose to automatically filter incorrectly classified sequences from the final result. This filtering is based on two criteria: entropy and likelihood ratio. The first, entropy, is a measure of the data itself, whilst the second, likelihood ratio, is a measure of the confidence in the classification result. In our case the entropy measure is used to characterise the randomness of the velocity of the motion sequence. Our hypothesis is that sequences with more random velocity are more likely to be incorrectly classified, as opposed to a sequence with a more constant velocity. The likelihood ratio is the ratio between the most likely sequence and the second most likely sequence. This ratio can be seen as a confidence measure of the classification result.

In the following section, Section 2, we look at previous approaches to vision-based control of mobile user interfaces. In Section 3, we present two methods used for producing motion information from image sequences. In Section 4, we describe the HMMs used for sequence classification and the use of *Maximum A Posteriori* (MAP) adaptation in adapting these models. In Section 5, we demonstrate in two sets of experiments how the criteria of entropy and likelihood ratio can be used to filter the results of a recognition task and also how the same criteria can be used to select data for performing unsupervised adaptation of HMMs using MAP adaptation. Finally we present our conclusion in Section 6.

2. Related Work

Much of the previous work on vision-based user interaction with mobile phones has utilised measured motion information directly for controlling purposes. In these systems the user can operate the phone through a series of hand movements whilst holding the phone to perform actions on the screen of the device such as scrolling or pointing and clicking [3]. For example, Siemens introduced an augmented reality game called Mozzies developed for their SX1 cell phone in 2003. This was the first mobile phone application utilizing the camera as a sensor. The goal of the game was to shoot down the synthetic flying mosquitoes projected onto a real-time background image by moving the phone around and clicking at the right moment. During user movements, the motion of the phone is recorded using a simple optical flow technique.

After this work, we have seen many other image motion based approaches. Möhring et al. [4] presented a tracking system for augmented reality on a mobile phone to estimate 3D camera pose using special colour coded markers. Other marker-based methods use a printed or hand-drawn circle [5], a hand-held target [6], and a set of squares [7] to facilitate the control task. One new solution was presented by Pears [8]. The idea of this approach was to use a camera on the mobile device to track markers on the computer display. This technique can compute which part of the display is viewed and the 6-DOF position of the camera with respect to the display.

An alternative to markers is to estimate motion between successive image frames with similar methods to those commonly used in video coding. Rohs [9] divided incoming frames into the fixed number of blocks and then determined the relative x, y, and rotational motion using a simple block matching technique in order to to interact with an RFID tag. Another possibility is to extract distinctive features such as edges and corners from images which exist naturally in the scene. Haro et al. [10] have proposed a feature-based method to estimate movement direction and magnitude, so the user can navigate the device screen in 2D. Instead of using local features, some approaches extract global features such as integral projections from the image [11].

Some recent and generally interesting directions for mobile interaction are to combine information from several different sensors. In their feasibility study, Hwang et al. [12] combined forward and backward movement and rotation around the y-axis from camera-based motion tracking, and tilts about the x- and z-axis from the 3-axis accelerometer. Also, a technique to couple wide area, absolute, and low resolution global data from a GPS receiver with local tracking using feature-based motion estimation was presented by DiVerdi and Höllerer [13].

Recently, the motion input was also applied for more advanced indirect interaction such as recognising signs. This increases the flexibility of the control system as the abstract signs can be used to represent any command, such as controls for a music player. A number of authors have examined the possibility of using phone motion to draw alphanumeric characters. Liu et al. [14] show examples of Latin and Chinese characters drawn using the ego-motion of a mobile device, although these characters are not recognised or used for control. Kratz and Ballagas [15] propose using a simple set of motions to interact with the external environment through the mobile device. In their case there are four symbols, consisting of a three-sided square in four different orientations, and due to the small size of the symbol set they report good performance with no user training.

The other solution studied in this paper, vision-based finger tracking, is well studied problem on desktop computers with numerous applications [16, 17]. On mobile devices, Henrysson et al. [18] considered how a front-facing camera on the phone can be used for 3-D augmented reality interaction. They compared finger gesture input to tangible input, keypad interaction, and phone tilting in user interface tasks. However, in their work finger tracking was performed by using simple frame differencing method. Similar system called Finteraction was introduced by Jenabi and Reiterer [19] but they do not provide much detail of

the tracking method. Davis et al. [20] presented a real-time algorithm for finger pointing. The method is based on skin detection which makes it susceptible to illumination changes and noise. In experiments, they evaluate the method in a picture browsing task achieving promising results. Recently, Terajima et al. [21] presented another template-based finger tracking system for recognizing motion made by the user. They achieve real-time performance but they do not provide any quantitative analysis of the algorithm.

3. Motion Feature Extraction

In our contribution, we propose two alternative solutions to extract motion information from successive images which can be used as a feature for classification. In the first approach, the ego-motion of the device is estimated while the user operates the phone through a series of hand movements. The second technique is to move an object such as a finger in front of the camera and simultaneously track the object during gestures. Both these approaches utilise the feature-based motion analysis as a subtask where a sparse set of image features are first selected from one image and then their displacements are determined. In order to improve accuracy of the motion information, an uncertainty of these features is also analysed.

3.1. Feature Motion Analysis. Feature motion analysis begins with the selection of image features from the first frame. The goal is to ensure that the features are distributed over the image so that the probability of sufficient presentation of overall image motion is high. We use a computationally straightforward way where the image area is split to nonoverlapping regions and one feature is selected from each region [22].

Another goal is to select some distinctive features which guarantee high precision in the estimation of the displacement vectors. Various criteria for selecting such features typically analyse the richness of texture within an image area [23]. One approach is to consider first-order image derivatives in the horizontal and vertical directions. The sum of squared derivatives provides a computationally simple criterion. An alternative approach we have used is eigenvalue analysis of 2×2 normal matrice which can give better features, but has slightly higher computational complexity.

To estimate the displacement of the features i, a block matching measure is evaluated exhaustively for a suitable range of integer displacements in both x- and y-directions. As a matching measure, we use either the sum of squared differences (SSDs) measure or its variant, zero-mean sum of squared differences (ZSSDs). The latter measure is more robust to lighting changes which can be crucial in some applications. Exhaustive evaluation of either of these measures gives a motion profile. The displacement that minimizes the criterion provides a feature motion estimate $\mathbf{d}_i$ which can be refined to subpixel accuracy via quadratic interpolation of the motion profile values.

Uncertainty of the obtained estimate is analysed by detecting those displacements that may be close to the true displacement according to the matching measure value. Selection of the set of those displacements is based on gradient-based thresholding of the motion profile. The result of this analysis is summarized as a covariance matrix $\mathbf{C}_i$.

As a result of these computational steps, we obtain a set of *motion features*. A motion feature i consists of the feature centroid location in the first image, $\mathbf{p}_i$, its displacement estimate $\mathbf{d}_i$, and the result of uncertainty analysis ($\mathbf{C}_i$). Device motion estimation and object tracking use this information as an input.

3.2. Device Motion Estimation. A mobile user interface system controlled through a series of hand movements requires a method for estimating the ego-motion of the device's camera [22]. Camera ego-motion is often estimated from 2-D image motion measured between two successive frames. As the observed motion in an image sequence may consist of multiple motions due to moving objects in a scene and motion parallax, one must consider solutions that estimate the dominant motion.

The ego-motion estimation generally refers to the computation of 6-DOF motion. However, the choice of a model and the number of parameters for the computation are application dependent. For simplicity we use a four-parameter similarity motion model which represents the displacement $\mathbf{d}$ of a feature located at $\mathbf{p} = [x, y]^T$ using

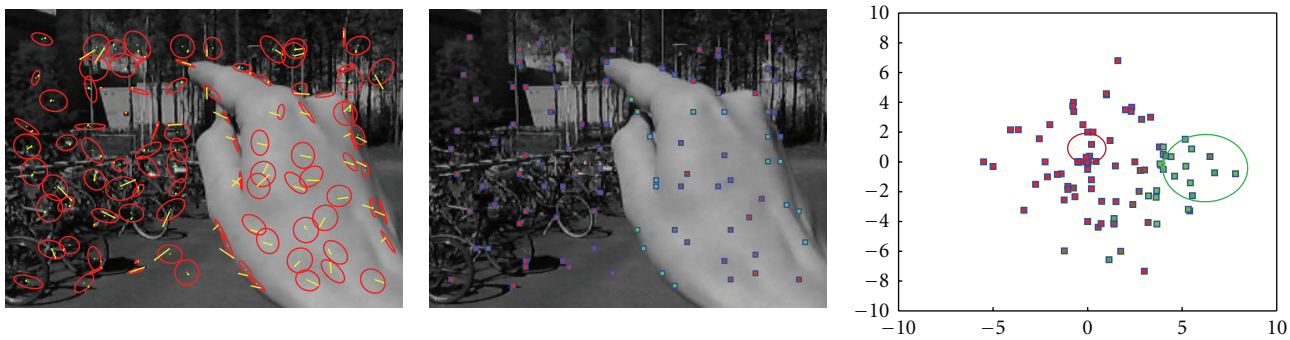
$$\mathbf{d} = \mathbf{d}(\boldsymbol{\theta}, \mathbf{p}) = \mathbf{H}[\mathbf{p}]\boldsymbol{\theta} = \begin{bmatrix} 1 & 0 & x & y \\ 0 & 1 & y & -x \end{bmatrix} \boldsymbol{\theta}, \tag{1}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3, \theta_4]^T$ is a vector of model parameters and $\mathbf{H}[\mathbf{p}]$ is an observation matrix. Here, θ_1 and θ_2 are related to common translational motion, and θ_3 and θ_4 encode information about 2-D rotation ϕ and scaling s, $\theta_3 = s\cos\phi - 1$ and $\theta_4 = s\sin\phi$.

The global motion describing the device motion is estimated using those motion features which pass an outlier analysis stage. Such analysis is necessary as feature displacement estimates can be erroneous due to image noise, or there may be several independent motions in the scene. It is assumed that the majority of motion features are associated with the global motion we want to estimate. To select those inlier features, we use an RANSAC-based scheme where pairs of motion features are used to instantiate motion model hypotheses, which are then voted for by other features.

A feature votes for a hypothesis if the displacement instantiated from the hypothesis is close to the estimated displacement. The covariance matrix $\mathbf{C}_i$ provides information about the feature motion uncertainty in different directions, and the calculation of votes uses $\mathbf{C}_i$-based Mahalanobis distance measure. Once inlier features have been selected, a weighted least squares approach is used to compute the estimate of the device motion. Primarily, weighting is based on measured uncertainties.

3.3. Object Tracking. The goal of object tracking is to estimate the motion of an object such as a finger which can then be used as a feature for recognising gestures [24].



(a) (b) (c)

Figure 1: (a) Motion features. Estimates of feature displacements (lines) and associated error covariances (ellipses). (b) Assignment of motion measurements to two components. Weightings are illustrated using colors (red : background ($w_{i,1}$ large), blue : foreground ($w_{i,2}$ large)). (c) Values of displacement estimates $\mathbf{d}_i = [u_i, v_i]^T$. Ellipses visualize the covariances of the two distributions.

With hand-held devices the camera also moves slightly when the user is operating the device. The problem is therefore formulated as a task of estimating two distinct motion components, the camera motion and the object motion. However, we are not so interested in segmenting the observed displacements into coherent regions in an image.

One way to track multiple object motions and cope with multimodal distribution is combinatorial data association methods [25]. In many tracking problems there is more than one measurement at the same time step available. Data association is a process to assign each of measurements to the appropriate objects or motion. Assigning measurements can be effective in the case of incoherent motion. Methods of this kind often perform data association and estimation separately by first assigning the measurements and then estimating the state. In the following, we review our method that is able to track multiple motions using a sparse set of motion features. One benefit compared to previous approaches is that no iterations are needed, making the algorithm computationally efficient.

In our model, we assume that the background and foreground motions are constant but subject to random perturbations. Translational models are considered as sufficient approximations, and then the state-space model of the camera ($j = 1$) and object + camera ($j = 2$) motions is

$$\mathbf{s}_j(k) = \mathbf{s}_j(k-1) + \varepsilon_j(k), \tag{2}$$

where the state $\mathbf{s}_j(k) = [u_j(k), v_j(k)]^T$ denotes the motion between the frames $k-1$ and k. $u_j(k)$ is the motion in x-direction and $v_j(k)$ is the motion in y-direction. $\varepsilon_j(k)$ is the process noise term, which is assumed to be zero-mean white Gaussian noise with covariance matrix $\mathbf{Q}_j = \sigma_j^2\mathbf{I}$. As foreground motion contains both camera and object motion, it is reasonable to assume that $\sigma_2^2 > \sigma_1^2$.

Object tracking uses motion features described in Section 3.1 and illustrated in Figure 1(a) as an input. Observed displacements of those features, $\mathbf{d}_i(k)$, are modelled as

$$\mathbf{d}_i(k) = \lambda_i\mathbf{s}_1(k) + (1-\lambda_i)\mathbf{s}_2(k) + \boldsymbol{\eta}_i(k), \tag{3}$$

where $\boldsymbol{\eta}_i(k)$ is the observation noise, which is assumed to obey zero-mean Gaussian distribution with covariance $\mathbf{R}_i$, and λ_i is a hidden binary assignment variable which indicates the object that generates the measurement.

To estimate the motions we use a technique where the Kalman filter [26] and the EM algorithm [27] are combined. The basic assumption is that the motion measurements $\mathbf{d}_i$ are drawn from either of two distributions corresponding to the background or foreground. Having some estimate of distribution parameters, we can assign measurements to the appropriate motion. Note that the Kalman filter could be used to directly estimate $\mathbf{s}_j(k)$ if the assignments λ_i were known. As these assignments are unknown, the predicted estimates of $\mathbf{s}_j(k)$ and *a priori* probabilities of associating features to motion components are used to compute soft assignments $w_{i,j}$ using a Bayesian formulation. An example of assignments is shown in Figures 1(b) and 1(c). The assignments step corresponds to the E step of the EM algorithm.

Soft assignments are then used in the computation of the Kalman gains which are needed to get the filtered estimates of $\mathbf{s}_j(k)$. The principle is that the lower the value of $w_{i,j}$ is the higher the observation noise $\mathbf{R}_{i,j}$ becomes. This weighting of the measurements corresponds to the M step of the EM algorithm.

To describe the algorithm in more detail, we denote the estimate of the state $\mathbf{s}_j(k)$ with $\hat{\mathbf{s}}_j^+(k)$ and associated error covariance matrix with $\mathbf{P}_j^+(k)$. The steps used to obtain the state estimate at time instant $k+1$ are as follows.

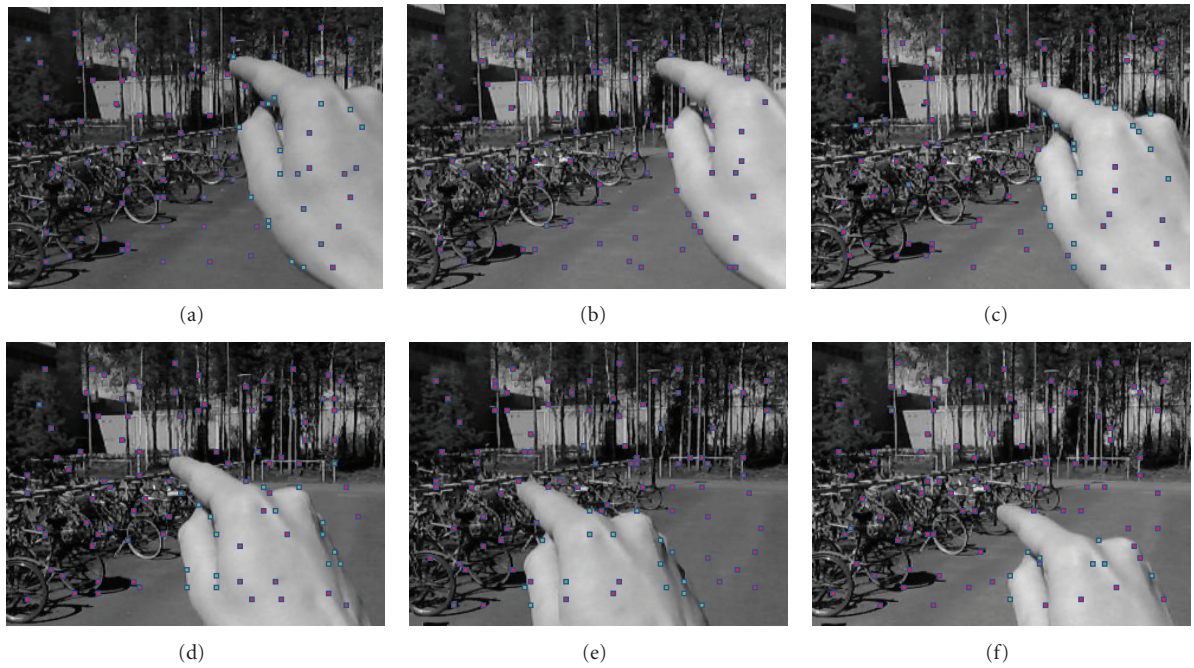
(1) Predict estimate $\hat{\mathbf{s}}_j^-(k+1)$ by applying dynamics (2)

$$\hat{\mathbf{s}}_j^-(k+1) = \hat{\mathbf{s}}_j^+(k), \tag{4}$$

and predict error covariance $\mathbf{P}_j^-(k+1)$

$$\mathbf{P}_j^-(k+1) = \mathbf{P}_j^+(k) + \mathbf{Q}_j. \tag{5}$$

(2) Compute the weights $w_{i,j}$ for each motion feature $\mathcal{F}_i(k+1) = (\mathbf{d}_i(k+1), \mathbf{C}_i(k+1))$ using a Bayesian formulation. Let $\pi_j(k) > 0$ be the *a priori* probability



(a) (b) (c)

(d) (e) (f)

Figure 2: Sample frames from the sequence 1. (a) Frame 40, (b) frame 50, (c) frame 60, (d) frame 70, (e) frame 80, and (f) frame 110.

of associating a feature with the motion j ($\sum_j \pi_j(k) = 1$). The weight $w_{i,j}$ is the *a posteriori* probability given by ($\sum_j w_{i,j} = 1$)

$$w_{i,j} \propto p\left(\mathbf{d}_i \mid \hat{\mathbf{s}}_j^-(k+1), \mathbf{P}_j^-(k+1) + \mathbf{C}_i(k+1)\right)\pi_j(k), \quad (6)$$

where the likelihood function $p(\cdot)$ is a Gaussian pdf, with mean $\hat{\mathbf{s}}_j^-(k+1)$ and covariance $\mathbf{P}_j^-(k+1)$.

(3) Use the weights $w_{i,j}$ to set the observation noise covariance matrices in (3) according to

$$\mathbf{R}_{i,j} = \mathbf{C}_i\left(w_{i,j} + \epsilon\right)^{-1}, \quad (7)$$

where ϵ is a small positive constant. Compute the Kalman gain

$$\mathbf{K}_j(k+1) = \mathbf{P}_j^-(k+1)\mathbf{H}^T\left(\mathbf{H}\mathbf{P}_j^-(k+1)\mathbf{H}^T + \mathbf{R}_j\right)^{-1}, \quad (8)$$

where $\mathbf{R}_j$ is a block diagonal matrix composed of $\mathbf{R}_{i,j}$ and $\mathbf{H} = [\mathbf{I}_2 \ldots \mathbf{I}_2]^T$ is the corresponding $2N \times 2$ observation matrix. Note that if $w_{i,j}$ has a small value, corresponding measurement is effectively discarded by this formulation.

(4) Compute filtered estimates of the state

$$\hat{\mathbf{s}}_j^+(k+1) = \hat{\mathbf{s}}_j^-(k+1) + \mathbf{K}_j(k+1)\left(\mathbf{z}(k+1) - \mathbf{H}\mathbf{s}_j^-(k+1)\right) \quad (9)$$

and compute the associated error covariance matrix

$$\mathbf{P}_j^+(k+1) = \left(\mathbf{I} - \mathbf{K}_j(k+1)\mathbf{H}\right)\mathbf{P}_j^-(k+1), \quad (10)$$

where $\mathbf{z}(k+1) = [\mathbf{d}_1(k+1)^T, \mathbf{d}_2(k+1)^T, \ldots, \mathbf{d}_N(k+1)^T]^T$.

(5) Update *a priori* probabilities for assignments with a recursive filter

$$\pi_j(k+1) = a\pi_j(k) + (1-a)\frac{1}{N}\sum_{i=1}^{N} w_{i,j}, \quad (11)$$

where $a < 1$ is a constant learning rate.

Figure 2 shows some frames of the sequence 1 with motion features observed during tracking. The weightings for each feature are illustrated using colors. The red and blue colors depict the background motion and the finger motion, respectively. It can be seen that most of the features are correctly associated. In Figure 2(b) all features are associated to the background because the finger motion is negligible. In our experimental tracker, 100 motion features are used, the image feature size is 5 by 5 pixels, and the maximum displacement is 12 pixels. We assume that the majority of features are extracted from the background. Therefore, the initial probabilities π_1 and π_2 (see (6)) for the background and the finger motion were set to 0.7 and 0.3, respectively. The learning rate a in (11) was set to 0.95 that guarantees a decent change in the proportion of mixture components.

4. Motion Sequence Recognition

In order to perform classification we must select an appropriate method of modelling the motion sequences produced by the feature extraction methods described in Section 3. The most common method currently used to model sequences of data are HMMs [28]. An HMM is a statistical model capable of representing temporal relations in these sequences. The data is characterised as a parametric stochastic process, and

the parameters of this process are automatically estimated. The data sequence is factorised over time by a series of hidden states and emissions from these states. The transition between states is probabilistic and depends only on the previous state. In our case [22, 24] the continuous emission probability from each state is modelled using *Gaussian Mixture Models* (GMMs) [28]. HMM training can be carried out using the *Expectation-Maximisation* (EM) algorithm [27] and sequence decoding using the Viterbi algorithm [29].

4.1. Maximum a Posteriori Adaptation. Due to the difficultly in tracking and the noisy nature of the measurements, in some applications, it may be difficult to create general models for each class that will perform well for many different users. In order to improve the models performance we propose using unsupervised *Maximum A Posteriori* (MAP) adaptation to tailor the general models for a specific user. We address the problem of controlling unsupervised learning by proposing a method of selecting adaptation data using a combination of entropy and likelihood ratio. We demonstrate how this approach can significantly improve the performance in the task of finger gesture recognition.

When using statistical models for pattern recognition we must train the models based on a training set. If this training set is labelled, then the *Maximum Likelihood* (ML) principle is used to update the model parameters during training. The likelihood of a training set $\mathcal{X}_{\text{train}}$ is maximised with respect to the parameters of the model θ. So we select the parameters θ^{ML} such that

$$\theta^{\text{ML}} = \arg\max_{\theta} p(\mathcal{X}_{\text{train}} \mid \theta). \tag{12}$$

The EM algorithm can be used to estimate (12). If, however, we are presented with unlabelled data to train the model, then there is the possibility that some of the training data will not correspond to the class we wish to recognise. Therefore we need some way of constraining the estimation of the model parameters to limit the effects of incorrect data. In MAP adaptation [30] a prior distribution over the parameters θ, $P(\theta)$, is used to constrain the updated parameters. The formulation for MAP estimation is similar to the formulation for ML estimation given in (12). However, in MAP estimation it is assumed that there is a prior distribution on the parameters to be estimated. The estimation of the parameters θ according to the MAP principle is given by

$$\begin{aligned}\theta_{\text{MAP}} &= \arg\max_{\theta} P\left(\theta \mid \mathcal{X}_{\text{adapt}}\right) \\ &= \arg\max_{\theta} p\left(\mathcal{X}_{\text{adapt}} \mid \theta\right) P(\theta),\end{aligned} \tag{13}$$

where $\mathcal{X}_{\text{adapt}}$ is the data selected for adaptation. Again the EM algorithm can be used for MAP estimation. The next section will look at the use of MAP learning for adapting the parameters of HMMs.

4.2. MAP Adaptation for Hidden Markov Models. In a GMM the set of parameters θ for each mixture m is given by

$$\theta = \{W, \mu, \Sigma\}, \tag{14}$$

where $W = \{\omega_m\}$ is the set of scalar mixture weights, $\mu = \{\mu_m\}$ is the set of vector means, and $\Sigma = \{\Sigma_m\}$ is the set of covariance matrices of the Gaussian mixture. This HMM is trained by updating the parameters of each Gaussian and also the transitions between each state. In MAP adaptation the estimation of the model parameters for each state is constrained by a prior distribution for these parameters, $\theta^{\text{prior}} = \{W^{\text{prior}}, \mu^{\text{prior}}, \Sigma^{\text{prior}}\}$. The updated parameters of a particular GMM mixture m, θ_m^{MAP}, can be estimated according to the following update equations:

$$\begin{aligned}w_m^{\text{MAP}} &= \alpha \cdot w_m^{\text{prior}} + (1-\alpha) \cdot w_m^{\text{ML}}, \\ \mu_m^{\text{MAP}} &= \alpha \cdot \mu_m^{\text{prior}} + (1-\alpha) \cdot \mu_m^{\text{ML}}, \\ h\Sigma_m^{\text{MAP}} &= \alpha \cdot \left[\sum_m^{\text{prior}} + \left(\mu_m^{\text{prior}} - \mu_m^{\text{MAP}}\right)\left(\mu_m^{\text{prior}} - \mu_m^{\text{MAP}}\right)^T\right] \\ &\quad + (1-\alpha) \cdot \left[\sum_m^{\text{ML}} + \left(\mu_m^{ML} - \mu_m^{\text{MAP}}\right)\left(\mu_m^{\text{ML}} - \mu_m^{\text{MAP}}\right)^T\right],\end{aligned} \tag{15}$$

where α is a weighting factor on the contributions of the prior parameters, θ^{prior}, and the current estimated parameters using ML, θ^{ML}.

4.3. Filtering Recognition Results. A key point in unsupervised learning is controling of either the learning process or the data used for adaptation. One way to control unsupervised adaptation is to filter out any incorrectly classified sequences before they are used for adapting the model. In this case we propose the use of entropy and the log likelihood ratio as criteria for selecting sequences for adaptation. This is based on our previous work [22] where we demonstrated that a combination of log likelihood ratio and entropy can be used as a measure of the confidence in the recognition result for a sequence.

To adapt the general model to a user specific model, we first classify the sequences produced by the user using the general model. We then use the entropy and log-likelihood ratio as the criteria for filter incorrect sequences from these results. Finally the sequences that pass the selection criteria are used to update the model for that class using MAP adaptation.

4.3.1. Likelihood Ratio. If we have a set of classes $\{Cl_1, Cl_2, \ldots, Cl_N\}$ and a sequence of data $\mathcal{X}$, then the class with the highest likelihood given $\mathcal{X}$ is denoted by Cl_a and the class with the second highest likelihood given $\mathcal{X}$ is denoted by Cl_b. The log-likelihood ratio δ for a particular sequence is given by

$$\delta = \log(p(Cl_a \mid \mathcal{X})) - \log(p(Cl_b \mid \mathcal{X})), \tag{16}$$

where $p(Cl \mid \mathcal{X})$ is the likelihood of the class Cl given the data sequence $\mathcal{X}$. In our experiments any sequence that produces a value of δ below a certain threshold is not used for adaptation.

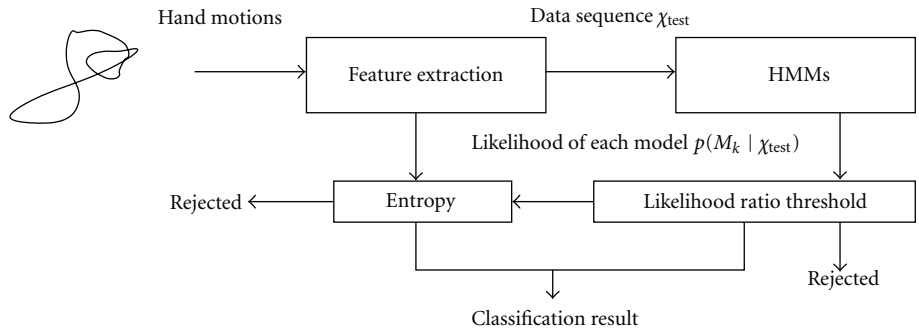


Figure 3: An overview of the proposed system for the recognition of device ego-motion, with result filtering based on log likelihood ratio and entropy.

4.3.2. Entropy. Information Entropy is a measure of the randomness of a probability distribution of a random variable $\mathcal{Y}$, and is given by [31]

$$\text{Ent}(\mathcal{Y}) = -L\sum_{n=1}^{N} P(y_n)\log_2 P(y_n), \tag{17}$$

where $P(y_n)$ is the probability of y, N is the number of samples, and L is a constant. In our case we take the first derivative of the motion trajectory $\mathcal{X}$, and this gives us the velocity of the motion. This continuous velocity sequence is quantised into a histogram, and we calculate the entropy of the entries in this histogram.

So a sequence with a higher entropy will have a more random velocity, while a sequence with lower entropy would have a more constant velocity. Our hypothesis is that well-formed signs will have a more constant velocity, and so a lower entropy, as opposed to more random or poorly formed signs. In this paper we demonstrate that sequences with higher entropy are more likely to be incorrectly classified and that by setting a threshold on the entropy we can filter these potentially incorrect sequences from the data used for adaptation.

5. Experiments

5.1. Device Motion Recognition. The system we propose here uses HMMs, described in Section 4, to model the device motion features described in Section 3.2. These models are then used to classify the device motion sequences that are input from the user. In order to ensure a minimum number of incorrect classifications this initial result is then filtered in order to reject any possibly incorrect commands before they can be executed. The methods used for filtering the results, likelihood ratio and entropy, are described in Section 4.3. An overview of the system is shown in Figure 3.

We use a two-level filtering of the result. The first level of filtering is based on the likelihood ratio between the most likely command and the second most likely command. This ratio can be seen as a confidence measure of the classification result. If this ratio is below a certain predefined threshold, then the confidence in the result is low and the sequence is rejected. A second level of filtering is employed to reject unintentional or accidental sequences, such as when the input system is activated without the user's knowledge or the user loses control of the phone for some reason. It is important that these unintended commands are not recognised and executed as real commands.

In our experiments we use two threshold values for δ. The first, δ_{hard}, is a hard decision, and any sequence with a log-likelihood ratio below this threshold is rejected. The second, δ_{soft}, is a soft decision, for any sequence where $\delta_{\text{hard}} < \delta < \delta_{\text{soft}}$ the entropy of the sequence is used as an additional indicator of the quality of the classification. Those sequences with a log likelihood ratio, δ, satisfying $\delta_{\text{hard}} < \delta < \delta_{\text{soft}}$ are classified according to their entropy. Any sequences with an entropy higher than a predetermined threshold are rejected as potentially incorrect. This entropy threshold Ent_{th} is set on the validation set as described in the next section.

Entropy is used for filtering the final classification result. In our experiments we have included a number of sequences where the user has either deliberately made a bad sign or has just moved the phone at random. These sequences are used to test the case where the system may be unintentionally turned on by the user or the user loses control of the phone whilst making a sign. The mean of the entropy of "bad" sequences in the validation was found to be significantly higher than the mean of the "good" sequences. This initial result indicates the potential of using the velocity entropy as a measure of the quality of the sign.

5.1.1. Experimental Procedure. In order to validate the technique described here a hypothetical control system of mobile phone functions was devised. In this system a series of control commands was proposed. These commands are composed of seven simple elements based on seven different motions. These seven elements are shown in Table 1. Using these basic elements alone the system would be limited to seven different commands, so in order to provide a greater number of commands more complex commands are constructed from these motion elements. These complex commands are used for our recognition experiments and are shown in Table 2. Although we have used 11 complex commands this could easily be extended to a larger number of commands.

Table 1: Seven basic motion elements.

Element	Type of motion
←	Left horizontal
→	Right horizontal
↑	Up vertical
↓	Down vertical
↙	Left diagonal
↘	Right diagonal
↺	Anticlockwise circle

Table 2: Eleven complex commands constructed from the seven basic motions.

Name	Command
Com1	↙↘↺
Com2	↙↘
Com3	↺
Com4	↑→
Com5	↑←
Com6	↓→
Com7	↓←
Com8	↑→↺
Com9	↑←↺
Com10	↓→↺
Com11	↓←↺

The experimental data was collected from 35 subjects. Each subject was asked to draw each of the commands in Table 2 five times using a standard camera equipped mobile phone, a Nokia N90. The majority of subjects had no previous experience in performing this task. There was considerable variability of the sequences both between subjects and also between different attempts from the same subject; this can clearly be seen in Figure 4.

The subjects were randomly divided into training, validation, and test sets. There were 20 subjects in the training set, 5 subjects in the validation set, and 10 subjects in the test set. Additionally 10 "bad" sequences were added to the validation set and 20 to the test set, giving a total of 1100, 285, and 570 sequences in the training, validation, and testing sets, respectively.

In addition to these subjects 30 random sequences were collected. These sequences were produced by moving the camera in a random way. These random or "bad" sequences were included in the data to test the system's performance with input caused by accidental activation of the camera or the user losing control of the phone whilst making a sign. The mean of the entropy of "bad" sequences in the validation set is 0.88 with standard deviation of 0.11, while the mean and standard deviation of the "good" sequences is 0.58 and 0.13, respectively.

It must be emphasised that there was no overlap of subjects between these three sets. The training set was used to train the parameters of the HMMs. The validation set was used to set the hyperparameters of the individual models, such as the number of Gaussians in the GMMs, that model the state distributions of the HMMs, and the number of states in the HMMs. The validation set was also used to set the hard threshold, δ_{hard}, and the soft threshold, δ_{soft}. In addition the entropy threshold Ent_{th} was set by using the validation set. The values of these parameters were; number of states = 11, number of Gaussians = 10, $\delta_{\text{hard}} = 5$, $\delta_{\text{soft}} = 30$ and $\text{Ent}_{\text{th}} = 0.72$.

5.1.2. Results and Discussion. The results of running the system on the 570 test sequences are shown in Table 3. It can be seen from these results that only 5 sequences are incorrectly classified, while 27 sequences that would have produced an incorrect result were rejected by the system. These rejected sequences included all of the 20 deliberately bad sequences. It is particularly interesting that 13 of these bad sequences were rejected using the entropy criteria. This result confirms that the higher entropy of the bad sequences observed in the validation set can be generalised to the bad sequences in the test set.

5.2. Object Motion Recognition. In this section we propose a system for control of a mobile device by again recognising motion sequences. In this instance the sequences are generated from tracking the users finger motion in front of the mobile device camera, as described in Section 3.3. In this system a series of simple control commands were proposed; these eight commands are shown in Figure 5. These command gestures are formed by the user drawing the sign in the air with an extended index finger in front of the mobile phone camera. So there are two challenges to overcome, first the tracking of the finger and secondly the classification of the sequences produced by this tracking.

In order to recognise the motion trajectories produced by finger tracking we are again using HMMs. However, due to the diversity in how people make the gestures it may be difficult to create general models for each class that will perform well for many different users. In order to improve the model performance we propose using unsupervised MAP adaptation to tailor the general models for a specific user. We address the problem of controlling unsupervised learning by proposing a method of selecting adaptation data using a combination of entropy and likelihood ratio. We demonstrate how this approach can significantly improve the performance in the task of finger gesture recognition.

5.2.1. Experimental Procedure. The experimental data was collected from 10 subjects. Each subject was asked to draw each of the commands in Figure 5 four times using a standard camera equipped mobile phone, a Nokia N73. This data formed the initial training and validation sets and also the baseline test set to measure the models performance before any adaptation. The collected data was divided into a training set of four subjects (128 sequences), a validation set of two subjects (64 sequences), and a test set of four subjects (128 sequences). To form the training and test sets for adaptation two subjects from the test set were asked to draw each sign an additional 11 times. These sequences

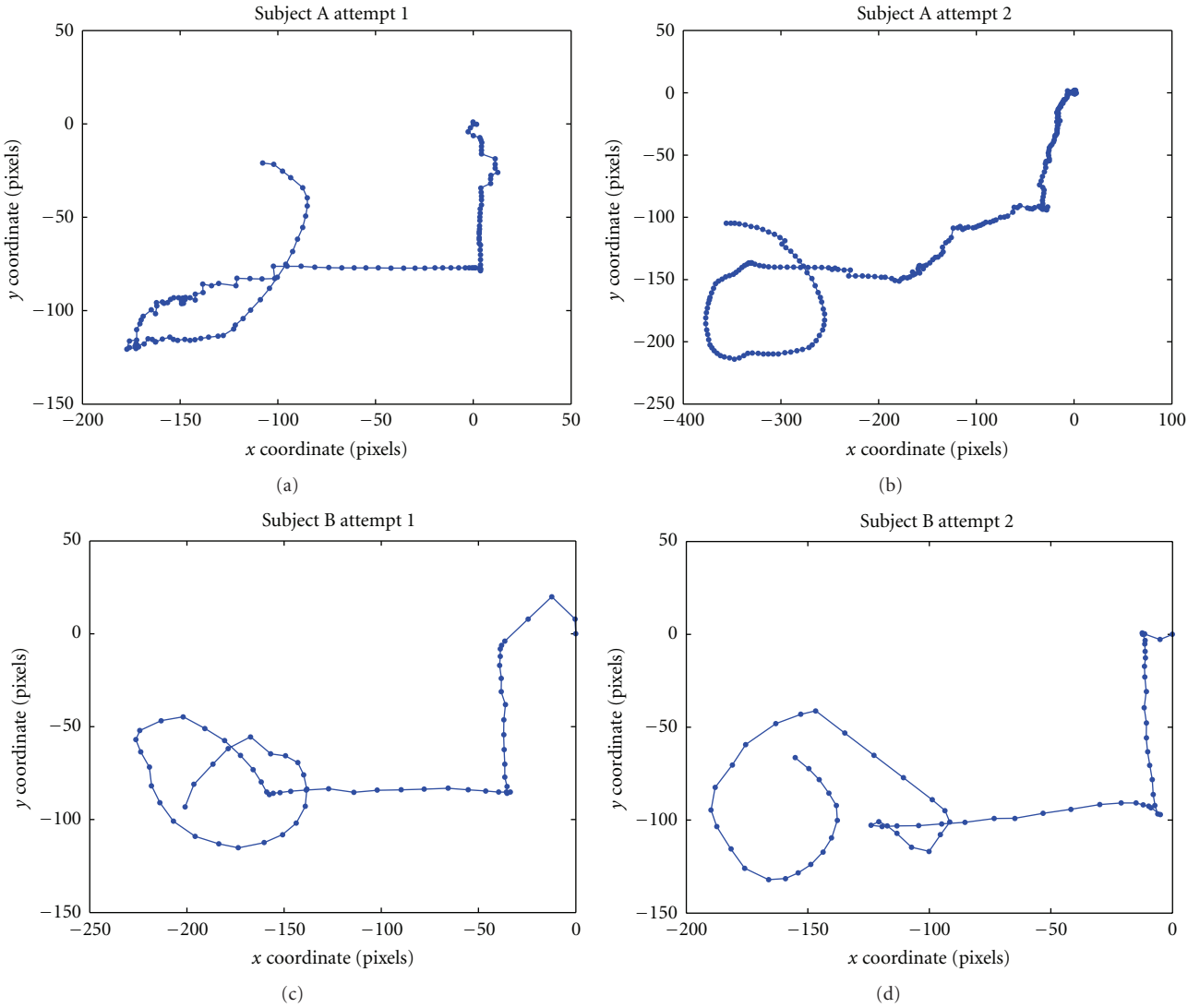


Figure 4: The sequence ↓ ← ↺ performed by two different users. Each row shows two attempts of the same sign from a single user. This shows both the intrauser and interuser variability of the data.

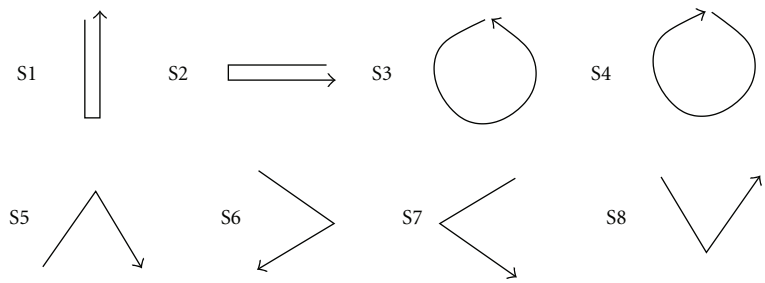


Figure 5: The eight signs chosen to represent mobile phone commands. In the experiments each user was asked to draw the sign in the air in front of the mobile phone camera.

form an adaptation set and test set for each of the subjects. Seven sequences from each subject are used to create a model adapted to that specific subject, and four sequences from each subject are used to test the performance of the adapted models. During MAP adaptation experiments the initial prior model is the baseline model, after this the prior model, θ^{prior}, is the ML estimated model, θ^{ML}, from the previous iteration. The adaption weighting, α, used in the testing was 0.8; this was found to be a reasonable value from previous work and was also tested using the validation set.

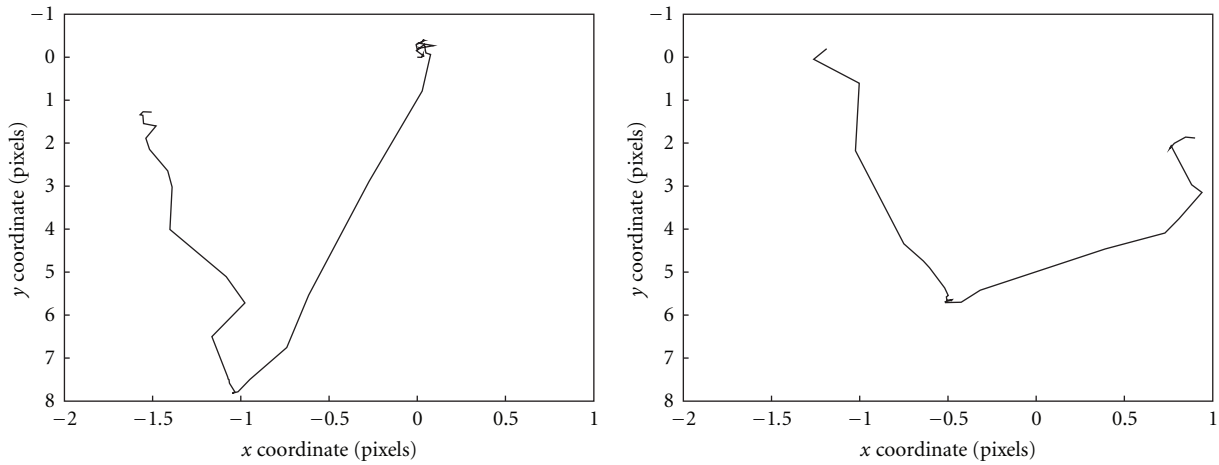


Figure 6: The signs S1 (a) and S8 (b) performed by a single user.

Table 3: Results of testing on 570 sequence of which 20 were intentionally bad. It should be noted that of 570 sequences only 5 were finally incorrectly classified.

Command	Correct	Correct rejected	Incorrect rejected	Incorrect
Com1	50	0	0	0
Com2	47	2	0	1
Com3	46	4	0	0
Com4	49	0	1	0
Com5	50	0	0	0
Com6	48	0	2	0
Com7	50	0	0	0
Com8	47	0	1	2
Com9	46	1	1	2
Com10	48	0	2	0
Com11	50	0	0	0
Bad seq	0	20	0	0
Total	531	27	7	5

5.2.2. Results and Discussion. We first ran the baseline experiments using the training set of four subjects and the test set of four different subjects. This produced a sequence recognition rate of 82% on the test set. It can be seen from the confusion matrix shown in Table 4 that the errors show a distinct pattern of confusion between sign S1 and sign S8 and also between sign S2 and sign S7. These signs are quite similar to each other with the only difference being a horizontal or vertical separation of the strokes in signs S8 and S7. This may be due to the variability between different subjects when making the signs, so if a user in the training set does a particularly narrow S8 or S7 this may cause the model to incorrectly classify S1 and S2 in the test set. This problem can clearly be seen in Figure 6.

In the next set of experiments we tailor the general model to an individual user using unsupervised MAP adaptation.

Table 4: Confusion matrix for the baseline recognition experiment using unadapted HMMs. The rows are the recognition result, and the columns are the labelling.

	S1	S2	S3	S4	S5	S6	S7	S8
S1	6	0	0	0	0	0	0	0
S2	0	8	0	0	1	0	0	0
S3	2	2	15	0	0	1	1	0
S4	2	0	0	16	0	0	0	0
S5	0	0	0	0	15	0	0	0
S6	0	0	0	0	0	15	1	0
S7	0	6	0	0	0	0	14	0
S8	6	0	1	0	0	0	0	16

Table 5: Confusion matrix for recognition experiment using HMMs adapted to the two subjects. The rows are the recognition result, and the columns are the labelling.

	S1	S2	S3	S4	S5	S6	S7	S8
S1	7	0	0	0	0	0	0	0
S2	0	8	0	0	0	0	0	0
S3	0	0	8	0	0	0	0	0
S4	1	0	0	8	0	2	0	0
S5	0	0	0	0	8	0	0	0
S6	0	0	0	0	0	4	0	0
S7	0	0	0	0	0	0	8	0
S8	0	0	0	0	0	2	0	8

The results of these experiments can be seen in Table 6; this shows that adapting with no constraints on the data used for adaptation can produce an increased recognition rate. If, however, we filter the sequences used for adaptation by applying the likelihood ratio and entropy criteria we can significantly improve this result. This improvement can also been seen in the confusion matrix shown in Table 5.

Table 6: Results for the adaptation experiments. This shows the baseline percentage recognition rate, the recognition rate when adapting with no constraints, and the recognition rate after adapting with entropy and likelihood ratio constraints.

	Baseline	Adapting with no constraints	Adapting with constraints
Subject 1	81.2	84.4	90.6
Subject 2	81.2	87.5	93.7

6. Conclusions

We have presented here two camera-based user interaction techniques for mobile devices that combine motion features and statistical sequence modelling to classify the hand movements of a user: the first by recognising the motion of the device held in the user's hand and second by recognising the motion of the user's finger. In order to improve the results produced by these systems we have introduced two methods of filtering the result of this classification, likelihood ratio and entropy. In the first application these criteria were used to filter incorrect or random sequences from the final result, while in the second the criteria are used for selecting data for unsupervised adaptation. It is clear from the results shown in Section 5.1.2 that our proposed method of using entropy as an indicator of badly formed sequences is able to filter out all such sequences from the final result. Additionally the results in Section 5.2.2 demonstrate that the same criteria can be used to improve the performance of the models using MAP adaptation.

We conclude that the computer vision-based motion estimation and recognition techniques presented in this paper have clear potential to become practical means for interacting with mobile devices. They can possibly also augment the information provided by other sensors, such as accelerometers and touch screens, in a complementary manner. In fact, the cameras in future mobile devices may, for most of time, be used as sensors for self-intuitive user interfaces rather than using them for digital photography.

References

[1] J. Rekimoto, "Tilting operations for small screen interfaces," in *Proceedings of the 9th Annual ACM symposium on User Interface Software and Technology*, pp. 167–168, ACM Press, 1996.

[2] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz, "Sensing techniques for mobile interaction," in *Proceedings of the 13th Annual ACM symposium on User Interface Software and Technology*, pp. 91–100, 2000.

[3] T. Capin, K. Pulli, and T. Akenine-Möller, "The state of the art in mobile graphics research," *IEEE Computer Graphics and Applications*, vol. 28, no. 4, pp. 74–84, 2008.

[4] M. Möhring, C. Lessig, and O. Bimber, "Optical tracking and video seethrough ar on consumer cell phones," in *Proceedings of the Workshop on Virtual and Augmented Reality of the GI-Fachgruppe AR/VR*, pp. 193–204, 2004.

[5] T. R. Hansen, E. Eriksson, and A. Lykke-Olesen, "Mixed interaction space—designing for camera based interaction with mobile devices," in *Proceedings of the Extended Abstracts on Human Factors in Computing Systems (CHI '05)*, pp. 1933–1936, 2005.

[6] M. Hachet, J. Pouderoux, and P. Guitton, "A camera-based interface for interaction with mobile handheld computers," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pp. 65–71, 2005.

[7] S. Winkler, K. Rangaswamy, and Z. Zhou, "Intuitive map navigation on mobile devices," in *Proceedings of the 4th International Conference on Universal Access in Human-Computer Interaction*, vol. 4555 of *Lecture Notes in Computer Science*, pp. 605–614, 2007.

[8] N. Pears, P. Olivier, and D. Jackson, "Display registration for device interaction a proof of principle prototype," in *Proceedings of the 3rd International Conference on Computer Vision Theory and Applications (VISAPP '08)*, pp. 446–451, January 2008.

[9] M. Rohs, "Real-world interaction with camera-phones," in *Proceedings of the 2nd International Symposium on Ubiquitous Computing Systems*, pp. 39–48, 2004.

[10] A. Haro, K. Mori, T. Capin, and S. Wilkinson, "Mobile camera-based user interaction," in *Proceedings of the IEEE International Conference on Computer Vision, Workshop on Human-Computer Interaction*, pp. 79–89, Beijing, China, 2005.

[11] A. Adams, N. Gelfand, and K. Pulli, "Viewfinder alignment," in *Proceedings of the Eurographics*, pp. 597–606, 2008.

[12] J. Hwang, J. Jung, and G. J. Kim, "Hand-held virtual reality: a feasibility study," in *Proceedings of the 13th ACM Symposium Virtual Reality Software and Technology (VRST '06)*, pp. 356–363, November 2006.

[13] S. DiVerdi and T. Höllerer, "GroundCam: a tracking modality for mobile mixed reality," in *Proceedings of the IEEE Virtual Reality Conference (VR '07)*, pp. 75–82, March 2007.

[14] X. Liu, D. Doermann, and H. Li, "Fast camera motion estimation for handheld devices and applications," in *Proceedings of the 4th International Conference on Mobile and Ubiquitous Multimedia*, pp. 103–108, 2005.

[15] S. Kratz and R. Ballagas, "Gesture recognition using motion estimation on mobile phones," in *Proceedings of the 3rd International Workshop on Pervasive Mobile Interaction Devices at Pervasive*, 2007.

[16] L. Jin, D. Yang, L. X. Zhen, and J. C. Huang, "A novel vision based finger-writing character recognition system," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, pp. 1104–1107, August 2006.

[17] S. M. Dominguez, T. Keaton, and A. H. Sayed, "A robust finger tracking method for multimodal wearable computer interfacing," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 956–972, 2006.

[18] A. Henrysson, J. Marshall, and M. Billinghurst, "Experiments in 3d interaction for mobile phone AR," in *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia*, pp. 187–194, 2007.

[19] M. Jenabi and H. Reiterer, "Finteraction—finger interaction with mobile phone," in *Proceedings of the Future Mobile Experiences (NordiCHI '08)*, 2008.

[20] J. E. Davis, O. Gallo, and S. M. Arteaga, "A camera-based pointing interface for mobile devices," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '08)*, pp. 1420–1423, October 2008.

[21] K. Terajima, T. Komuro, and M. Ishikawa, "Fast finger tracking system for in-air typing interface," in *Proceedings of the 27th Extended Abstracts on Human Factors in Computing Systems (CHI '09)*, pp. 3739–3744, 2009.

[22] M. Barnard, J. Hannuksela, P. Sangi, and J. Heikkilä, "A vision based motion interface for mobile phones," in *Proceedings of the 5th International Conference on Computer Vision Systems*, 2007.

[23] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 593–600, June 1994.

[24] J. Hannuksela, M. Barnard, P. Sangi, and J. Heikkilä, "Adaptive motionbased gesture recognition interface for mobile phones," in *Proceedings of the 6th International Conference on Computer Vision Systems*, vol. 5008 of *Lecture Notes in Computer Science*, pp. 271–280, 2008.

[25] B. Rao, *Data Association Methods for Tracking Systems*, MIT Press, Cambridge, Mass, USA, 1993.

[26] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME-Journal of Basic Engineering, Series D*, vol. 82, pp. 35–45, 1960.

[27] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[28] L. R. Rabiner, "Tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[29] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE Transactions Information Theory*, vol. 13, pp. 260–269, 1967.

[30] J. L. Gauvain and C. H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[31] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.