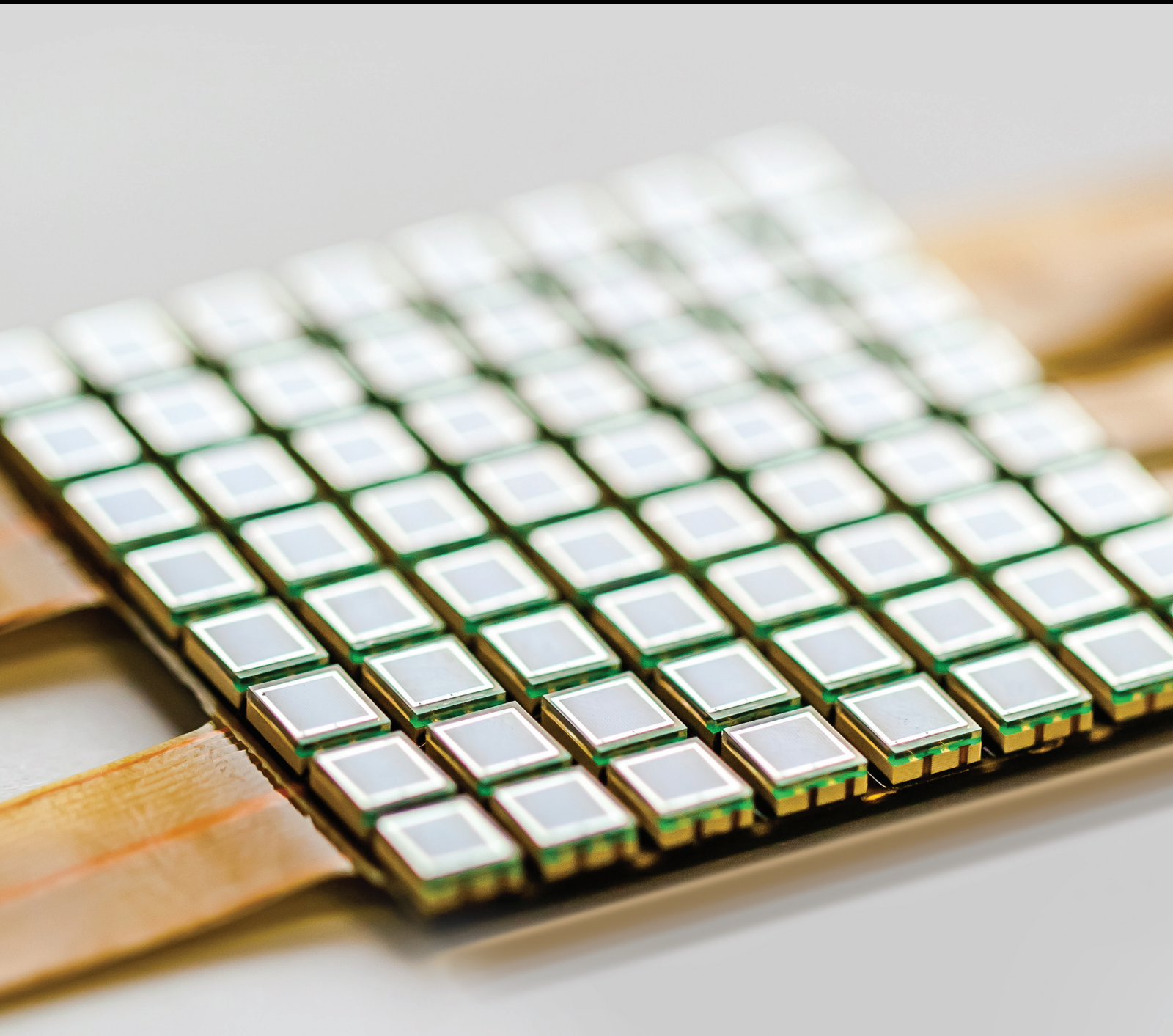
Journal of Sensors

Machine Vision Sensors

Lead Guest Editor: Oleg Sergiyenko
Guest Editors: Vera Tyrsa, Wendy Flores-Fuentes, Julio Rodriguez-Quiñonez, and Paolo Mercorelli



Machine Vision Sensors

Journal of Sensors

Machine Vision Sensors

Lead Guest Editor: Oleg Sergiyenko
Guest Editors: Vera Tyrsa, Wendy Flores-Fuentes, Julio Rodriguez-Quiñonez, and Paolo Mercorelli

Editorial Board

Contents

Hindawi
Journal of Sensors
Volume 2018, Article ID 3202761, 2 pages
https://doi.org/10.1155/2018/3202761

Editorial

Machine Vision Sensors

Oleg Sergiyenko[1], **Vera Tyrsa**[2], **Wendy Flores-Fuentes**[1], **Julio Rodriguez-Quiñonez**[1] and **Paolo Mercorelli**[3]

[1]*Autonomous University of Baja California, Mexicali, BC, Mexico*
[2]*Automobile and Highway University, Kharkiv, Ukraine*
[3]*Leuphana University, Lueneburg, Germany*

Correspondence should be addressed to Oleg Sergiyenko; srgnk@uabc.edu.mx

Received 8 January 2018; Accepted 9 January 2018; Published 27 February 2018

The advances in mechatronics and robotics in the last 50 years have been improved by the sensory inputs to systems, providing them with the sight capacity, when the machine visual interpretation is carried out, it increases systems intelligence. While visual perception to machines promises the greatest improvement, at the same time, it presents the greatest challenge, particularly when they are in the real-world situations.

This special issue is intended to present and discuss breakthrough technological developments which are expected to revolutionize contributions to sensors, signal processing, and algorithms in machine vision, control, and navigation. It provides a reference on machine vision supporting techniques and 3D reconstruction for researchers and engineers. These contributions are focused on sensors for vision systems, intelligent navigation algorithms, and machine motion controllers, particularly on applications of unmanned aerial vehicle, drones, and autonomous and mobile humanoid robots. We received several submissions, and after two rounds of rigorous review, 10 papers were accepted.

In the first paper "A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles," A. Carrio et al. perform a thorough review on recent reported uses and applications of deep learning for UAVs, including the most relevant developments as well as their performances and limitations. Also, a detailed explanation of the main deep learning techniques is provided.

In the paper "An Efficient Calibration Method for a Stereo Camera System with Heterogeneous Lenses Using an Embedded Checkerboard Pattern," P. Rathnayaka et al. present two approaches to calibrate a stereo camera setup with heterogeneous lenses: a wide-angle fish-eye lens and a narrow-angle lens in left and right sides, respectively. Instead of using a conventional black-white checkerboard pattern, they design an embedded checkerboard pattern by combining two differently colored patterns.

In the paper "Automatic Detection Technology of Sonar Image Target Based on the Three-Dimensional Imaging," W. Kong et al. present a set of sonar image automatic detection technologies based on 3D imaging. The process consists of two steps, approximate position of the object by calculating the signal-to-noise ratio of each target and then the separation of water bodies and strata by maximum variance between clusters (OTSU).

In the paper "Vision System of Mobile Robot Combining Binocular and Depth Cameras," Y. Yang et al. propose a 3D reconstruction system combining binocular and depth cameras to improve the precision of the 3D reconstruction. The whole system consists of two identical color cameras, a TOF depth camera, an image processing host, a mobile robot control host, and a mobile robot; finally, double thread processing method is applied to improve the efficiency of the system.

In the paper "Aphid Identification and Counting Based on Smartphone and Machine Vision," S. Xuesong et al. present a design of counting software that can be run on smartphones for real-time enumeration of aphids, the counting accuracy in the greenhouse is above 95%, while it can reach 92.5% outside.

In the paper "Multisource Data Fusion Framework for Land Use/Land Cover Classification Using Machine Vision," S. Qadri et al. present the design of a novel framework for multispectral and texture feature-based data fusion to identify the land use/land cover data types correctly. The study describes the data fusion of five land use/cover types, that is, bare land, fertile cultivated land, desert rangeland, green pasture, and Sutlej basin river land derived from remote sensing. The obtained accuracy acquired using multilayer perceptron for texture data, multispectral data, and fused data was 96.67%, 97.60%, and 99.60%, respectively.

In the paper "A Nonlocal Method with Modified Initial Cost and Multiple Weight for Stereo Matching," S. Gao et al. present a new nonlocal cost aggregation method for stereo matching, a modified initial cost is used, and a robust and reasonable tree structure is developed. The proposed method was tested on Middlebury datasets and, experimental results show that the proposed method surpasses the classical nonlocal methods.

In the paper "Global Measurement Method for Large-Scale Components Based on a Multiple Field of View Combination," Y. Zhang et al. propose a noncontact and flexible global measurement method combining a multiple field of view (FOV); the measurement system consists of two theodolites and a binocular vision system with a transfer mark. The authors also propose a new global calibration method to solve the coordinate system unification issue of different instruments in the measurement system.

In the paper "Automated Recognition of a Wall between Windows from a Single Image," Y. Zhang et al. propose a method to recognize the wall between windows from a single image automatically. The method starts from detection of line segments with further selection. The color features of the two sides are employed to detect line segments as candidate window edges. Finally, the images are segmented into several subimages, window regions are located, and then the wall between the windows is located.

In the paper "Visual Vehicle Tracking Based on Deep Representation and Semisupervised Learning," Y. Cai et al. propose a semisupervised tracking algorithm that uses deep representation and transfer learning. A 2D multilayer deep belief network is trained with a large number of unlabeled samples. The nonlinear mapping point at the top of this network is subtracted as the feature dictionary. Then, the feature dictionary is utilized to transfer train and update a deep tracker. The positive samples for training are the tracked vehicles. Finally, a particle filter is used to estimate vehicle position.

Acknowledgments

The guest editorial team would like to thank the authors of all the papers submitted to this special issue. We also want to thank the institutions to which we are affiliated, specially the Autonomous University of Baja California. Finally, the editors also wish to thank the anonymous reviewers, some of whom helped with multiple review assignments. We hope that you will enjoy reading this special issue focused on machine vision.

Oleg Sergiyenko
Vera Tyrsa
Wendy Flores-Fuentes
Julio Rodriguez-Quiñonez
Paolo Mercorelli

Hindawi
Journal of Sensors
Volume 2018, Article ID 5813514, 8 pages
https://doi.org/10.1155/2018/5813514

Research Article

Face Recognition and Drunk Classification Using Infrared Face Images

Gabriel Hermosilla,[1] **José Luis Verdugo**,[2] **Gonzalo Farias**,[1] **Esteban Vera**,[1] **Francisco Pizarro**,[1] **and Margarita Machuca**[3]

[1]*Escuela de Ingeniería Eléctrica, Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile*
[2]*Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile*
[3]*Universidad de Santiago de Chile, Santiago, Chile*

Correspondence should be addressed to Gonzalo Farias; gonzalo.farias@ucv.cl

Received 25 April 2017; Revised 10 July 2017; Accepted 3 December 2017; Published 30 January 2018

Academic Editor: Paolo Mercorelli

The aim of this study is to propose a system that is capable of recognising the identity of a person, indicating whether the person is drunk using only information extracted from thermal face images. The proposed system is divided into two stages, face recognition and classification. In the face recognition stage, test images are recognised using robust face recognition algorithms: Weber local descriptor (WLD) and local binary pattern (LBP). The classification stage uses Fisher linear discriminant to reduce the dimensionality of the features, and those features are classified using a classifier based on a Gaussian mixture model, creating a classification space for each person, extending the state-of-the-art concept of a "DrunkSpace Classifier." The system was validated using a new drunk person database, which was specially designed for this work. The main results show that the performance of the face recognition stage was 100% with both algorithms, while the drunk identification saw a performance of 86.96%, which is a very promising result considering 46 individuals for our database in comparison with others that can be found in the literature.

1. Introduction

Thermoregulation is a process in which a biological organism modifies its internal temperature within certain limits and is commanded by the hypothalamus. For humans, the temperature reached in a normal state is approximately 36.7° (internal) and 33.5° (skin). In the case that the internal temperature is above 36.7°, thermoregulation generates two processes for heat loss, sweating and vasodilation. If the temperature is less than 36.7°, the thermoregulatory system generates thermogenesis processes (vasoconstriction and piloerection) to increase the temperature.

Some research studies have shown that the thermoregulatory system can be altered depending on mood or the consumption of certain foods [1]. In a series of publications, it is concluded that alcohol alters the correct operation of the system responsible for thermoregulation [2, 3], generating an induced vasodilation in the skin, which increases heat loss through convection, leading to a decrease in body temperature that is directly related to the amount of alcohol consumed.

The identification of drunk people has its basis in biology, medicine, and toxicology. Alcohol causes motor disturbances and disturbances in the psychic system, resulting in abnormal behaviour on a biological level, such as dilation of blood vessels [2–6] and increased blood pressure. In the case of the human face, there is a temperature increase in capillary density, such as around the nose, forehead, and eyes.

Despite the large number of applications in machine learning, such as face recognition, facial expressions, and personal identification, computer systems applied to the classification of drunk people have not been widely studied. The most notable work done is that of researchers at the University of Patras, Greece [7–11], who have tried to

distinguish a sober or drunk person based on the variations in facial features. In simple words, in [7, 10] and [11], it is shown that the frontal region and nose are the most appropriate for acquiring information to classify people as drunk or sober, using a neural network to perform the classification task. In [9], it is concluded that alcohol causes an increase in the temperature of the eyes, which could be useful for classification. In [8], the extraction of the vascular network is proposed based on the works of Buddharaju et al. [3–6]. When analysing the total area in pixels of the vascular network of a sober subject compared to a drunk subject, it may be possible to obtain an indicator (feature) to identify whether the subject has consumed alcohol. In [10], the aim is to demonstrate that it is possible to differentiate between sober and drunk persons using the intensities of pixels located in certain regions or areas of the face (forehead, nose, and mouth). A space of separable features can be generated using these intensities; however, this study used only a small number of subjects (8 individuals) and the individuals met certain characteristics (similar weight and height), and as such, it is not possible to ensure generalisation of the classifier.

Again in [10], a method is proposed to find regions with higher thermal variation in the face by comparing intensities of a person while sober and while in a drunken state. It is concluded that the forehead region shows an increase in temperature relative to the region of the nose. The feature extraction used in [10] is mainly based on analysing a number of pixels around interesting areas of the face, to which methods to reduce the dimensionality of the features are applied, such as linear discriminant analysis (LDA) or principal component analysis (PCA). The space generated by LDA, called "DrunkSpace," is used to distinguish between the states of sober and drunk.

In this context, the purpose of this study is to generate a classification system, based in the "DrunkSpace" proposed in [10], to identify whether an individual is drunk, using approaches of computer vision and pattern recognition. The primary objective is to extract features (patterns) of the face of thermal images obtained from a drunk person to build a Bayesian classifier based on a Gaussian mixture model (GMM) [12, 13]. The importance of thermal imaging is that it can be used to obtain patterns based on the thermal information of the face, which is linked to the processes of thermoregulation of the human face and the amount of alcohol consumed. Besides, since there are few databases available with a reduced number of thermal images of drunk people, it is proposed to create a public drunk thermal database to study the drunk person classification (available in https://goo.gl/7Gxs18).

2. Thermal Face Database

This section describes in detail how the Pontificia Universidad Católica de Valparaíso-Drunk Thermal Face database (PUCV-DTF) was acquired.

2.1. Recruitment. An open call was made through posters at the school of Electrical Engineering at the Pontificia Universidad Católica de Valparaíso. People who attended the call were informed of the research protocol and signed an informed consent form which was previously approved by the Ethics Committee of the university.

Table 1: Procedure of capturing the PUCV-DTF database.

Stage	Description	Time
Rest 1	Arrival at the laboratory and data acquisition (weight, height, etc.)	30 min
Meas. 1	Temperature and breath test measurement	1 min
Capt. 1	50 images of the subject (sober) are captured	1 min
Cons. 1	A can of beer is consumed	5 min
Rest 2	Stabilisation of metabolism in the laboratory	30 min
Meas. 2	Temperature and breath test measurement	1 min
Capt. 2	50 images of the subject (1 beer) are captured	1 min
Cons. 2	A can of beer is consumed	5 min
Rest 3	Stabilisation of metabolism in the laboratory	30 min
Meas. 3	Temperature and breath test measurement	1 min
Capt. 3	50 images of the subject (2 beer) are captured	1 min
Cons. 3	A can of beer is consumed	5 min
Rest 4	Stabilisation of metabolism in the laboratory	30 min
Meas. 4	Temperature and breath test measurement	1 min
Capt. 4	50 images of the subject (3 beer) are captured	1 min
Cons. 4	A can of beer is consumed	5 min
Rest 5	Stabilisation of metabolism in the laboratory	30 min
Meas. 5	Temperature and breath test measurement	1 min
Ending	50 images of the subject (4 beer) are captured	1 min

2.2. Participants. 46 individuals, 40 men and 6 women, were selected. The average age of individuals was 24 years with a standard deviation of approximately 3 years (the minimum age was 18 years, and the maximum was 29 years), being in good health without problems related to alcohol consumption. This analysis was performed by a screening test designed to exclude people who consume alcohol regularly.

2.3. Procedure. Subjects attended the robotics lab where they rested for 30 minutes to stabilise the metabolism to the temperature conditions of the laboratory. The subject then consumed a 355 mL can of 5.5° beer, waited another 30 minutes in the laboratory, and then repeated the procedure until four beers had been consumed. See Table 1 for the whole capturing procedure.

Upon completion of the procedure, subjects with about 0.8 g/L of blood alcohol (drunk) should remain in place until the alcohol percentage decreased to under 0.2 g/L. This was verified by measuring with a breath test. It should be noted that during the entire experiment a paramedic was in the lab to verify the status of the individuals.

2.4. Thermal Imaging. The thermal camera used was a FLIR TAU 2 [14] with a resolution of 640×480 pixels, a frame rate of 30 frames per second, thermal sensitivity of 50 mK, and a spectrum range between 7.5 and 13.5 μm. The database

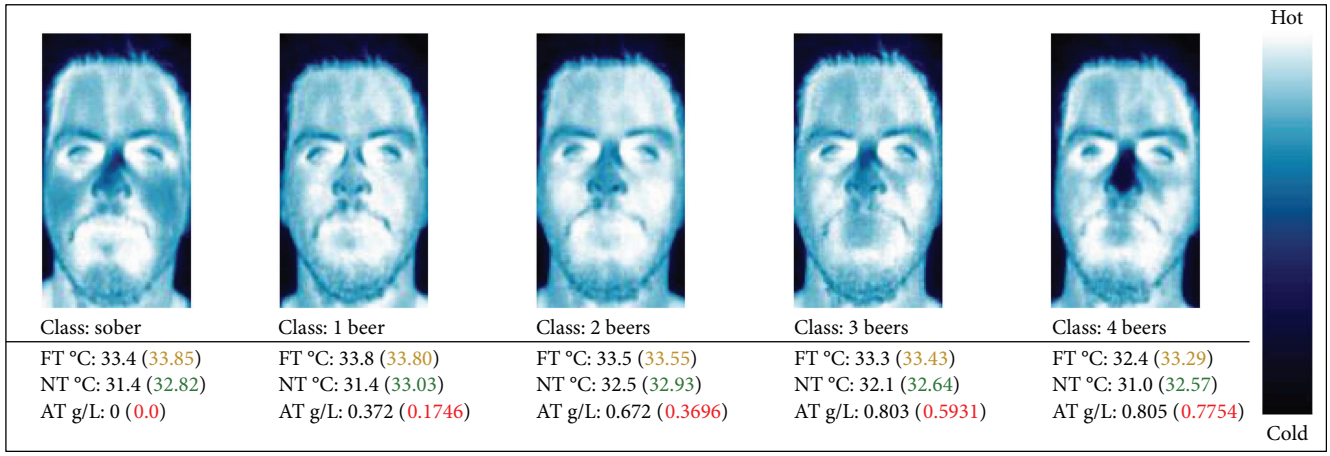


Figure 1: Images in pseudo colour to show how the face changes after consuming 4 beers. See the text for details.

included 46 individuals with five subsets, a total of 250 images per subject and 50 images per subset. The set is classified into 5 subsets: "Sober," "1 Beer," "2 Beers," "3 Beers," and "4 Beers," corresponding to the procedure of capturing the database, as summarised in Table 1. Following the acquisition of thermal imaging, a preprocessing was performed, where all the images were cropped and aligned according to the coordinates of the eyes, which were marked manually, giving a final resolution of 81×150 pixels. The thermal images were normalised using (1), which consists of applying a linear mapping to the pixel intensity values in the range $[N_{\min}, N_{\max}]$:

$$I_{\text{norm}}(i,j) = \frac{I_{(i,j)} - I_{\min}}{I_{\max} - I_{\min}} (N_{\max} - N_{\min}) + N_{\min}, \quad \forall (i,j) \in \Omega, \tag{1}$$

where $I_{\min}$ and $I_{\max}$ are the minimum and maximum values in the image: $I_{\min} = \min_{(i,j)\in\Omega} I(i,j)$, and $I_{\max} = \max_{(i,j)\in\Omega} I(i,j)$. In the experiments, the value range of [0,255] is used. An example of a subject with 5 subsets is shown in Figure 1. In order to highlight the information obtained from the alcohol consumption, colour has been applied to the images from the subsets. The figure shows also the forehead temperature (FT), the nose temperature (NT), and the alcohol tester (AT) measures. In the parenthesis, the average values of FT, NT, and AT for each of the classes are shown. Please note that the temperature of the thermal face image varies due to alcohol consumption. However, the nasal contrast varies when the subject has consumed beers. This effect could be attributable to either thermoregulation process or the breathing effect produced during exhalation or inhalation phase or a combination of both of them.

In terms of the alcohol concentration obtained in each breath test measurement, the range of values for each class can be observed in Table 2. Note that the values of alcohol concentration obtained show that the classes are overlapping due to the variability of subjects captured (different weight, height, age, sex, etc.). However, in our experiments, we want to classify whether the subject is sober or has consumed any beer (classes 1 Beer, 2 Beers, 3 Beers, or 4 Beers), regardless of the amount of alcohol concentration that the individual possesses. See in detail the values of temperature and alcohol test measurements in a document attached in the link of the database.

Table 2: Range of alcohol concentration by classes.

Class	Minimum (g/L)	Maximum (g/L)	Mean (g/L)
Sober	0	0	0
1 Beer	0	0.39	0.17
2 Beers	0.15	0.68	0.37
3 Beers	0.29	1.29	0.59
4 Beers	0.43	1.68	0.78

3. Feature Extraction and Classification

The feature extraction process consists of selecting local regions of a thermal face image and then extracting the information using dimensionality reduction methods. As indicated above, the process carried out in this study is related to the generation of a "DrunkSpace," as presented in [10]. In [10], the information from different regions of the face is extracted from a grid of 20 points. The problem with the grid proposed by [10] is that there are no biological details on the location of the feature points. For this reason and inspired by [15], a different grid of 22 points is chosen. In [15], the aim is to generate a thermographic map of the human face, for which a number of sensors are located on the face and neck of each subject. These 22 points are selected at positions where there are capillaries and veins that cross the face, as seen in any of the sample faces shown in Figure 2. Once the grid had been defined, the information is extracted from the thermal face images. As the information in the selected pixel of the grid may be subjected to noise, we decided to consider a neighborhood of 3×3 pixels around every point of the grid and compute the average intensity for each of the 22 regions of the face. Since the images

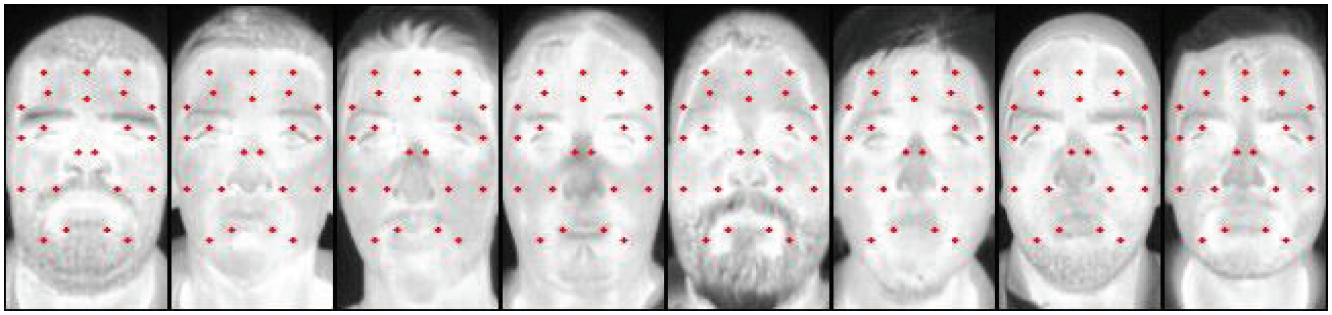


Figure 2: Feature extraction regions used for subjects in the database.

obtained from the database are aligned, the grid was used for all subjects of the database as a unique mask.

After having extracted the features of each thermal image, a feature vector of 22 dimensions was generated. Therefore, for a subject, there are 50 feature vectors for each class ("Sober," "1 Beer," "2 Beers," "3 Beers," and "4 Beers"). It is not recommended to use the 22-dimensional vector for classification due to its high dimensionality, the complexity of generating a hyperplane that separates all of these dimensions, and the high computational expense it would entail. Thus, to reduce the dimensionality, the Fisher linear discriminant analysis was used.

3.1. Fisher Linear Discriminant Analysis (FLD). First, it is assumed that there is a classification problem which involves two different classes (w_1 and w_2), and for each class, there are n_i m-dimensional samples. Thus, there is a set of n samples: $\mathbf{x} = \{\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \dots, \mathbf{x}^n\}$, n_1 corresponding to the class w_1 and n_2 to w_2, and so forth. The FLD method aims to obtain a transformation from the x-space to the y-space, through the linear projection of all the samples ($\mathbf{x}$) on a line, using the weights $\mathbf{w}$. However, the line to be selected must maximise the separability of the projected samples between the different classes. The linear combination that allows us to project the samples from the x-space to the y-space is represented in

$$y = \mathbf{w}^t \mathbf{x}, \tag{2}$$

where

$$\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}. \tag{3}$$

To find the adequate projection line, we must define a separation measurement between the projected data to then maximise this separation. The solution proposed by Fisher [16] is to maximise a function that represents the difference between the mean values of each class, normalised by a measurement of the variability inherent to each class. The objective function to maximise $J(\mathbf{w})$ can be represented as the function of two scatter matrices, $\mathbf{S}_W$ and $\mathbf{S}_B$ as shown in

$$J(\mathbf{w}) = \frac{\mathbf{w}^t \mathbf{S}_B \mathbf{w}}{\mathbf{w}^t \mathbf{S}_W \mathbf{w}}. \tag{4}$$

The $\mathbf{S}_W$ (the scatter matrix within classes) can be represented as the function of the original samples (x-space) or as the function of the projected samples. The $\mathbf{S}_B$ (the scatter matrix between classes) can be defined in terms of the original and projected samples. Both definitions are shown below together with the definition of the covariance matrix. See (5), (6), and (7) for the scatter matrices.

$$\mathbf{S}_i = \sum_{x \in w_i} (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^t, \tag{5}$$

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2, \tag{6}$$

$$\mathbf{S}_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^t, \tag{7}$$

where μ_i is the mean value of the original samples of the ith class and w is the projection weights. Finally, the optimal projection is obtained using the weights $\mathbf{w}^*$ (8):

$$\mathbf{w}^* = \arg \max_w \left(\frac{\mathbf{w}^t \mathbf{S}_B \mathbf{w}}{\mathbf{w}^t \mathbf{S}_W \mathbf{w}} \right) = \mathbf{S}_W{}^{-1} (\mu_1 - \mu_2). \tag{8}$$

This optimal solution $\mathbf{w}^*$ is given by the eigenvector(s) of $\mathbf{S}_X = \mathbf{S}_W{}^{-1}(\mu_1 - \mu_2)$, corresponding to the largest eigenvalue. Using FLD in our problem reduced the 22-dimensional vectors to two dimensions, while also maximising the distance between the means of the different classes and minimising the variance of each class. This is seen in the following example in Figure 3. Figure 3(a) shows a plane where two randomly chosen features are projected from a subject. In this figure, it is clear that not all classes are separable, since many of the features overlap between classes. However, when applying the FLD method, the DrunkSpace obtained (see Figure 3(b)) is completely separable and it is possible to identify projected clusters for each class.

3.2. Gaussian Mixture Model (GMM). Once the dimensionality reduction (with FLD) of the data had been carried out, an approach called the Gaussian mixture model (GMM) was then selected to perform the classification. A GMM is a probabilistic distribution whose probability density function is a linear combination of a finite quantity of Gaussian distributions. Each one of these Gaussian distributions represents a different class. In our case, four distributions were used as training classes (Sober, 1 Beer, 2 Beers, and 3 Beers) to generate the GMM. The remaining distribution (4 Beers) was used as a test set. The classification of the test data is

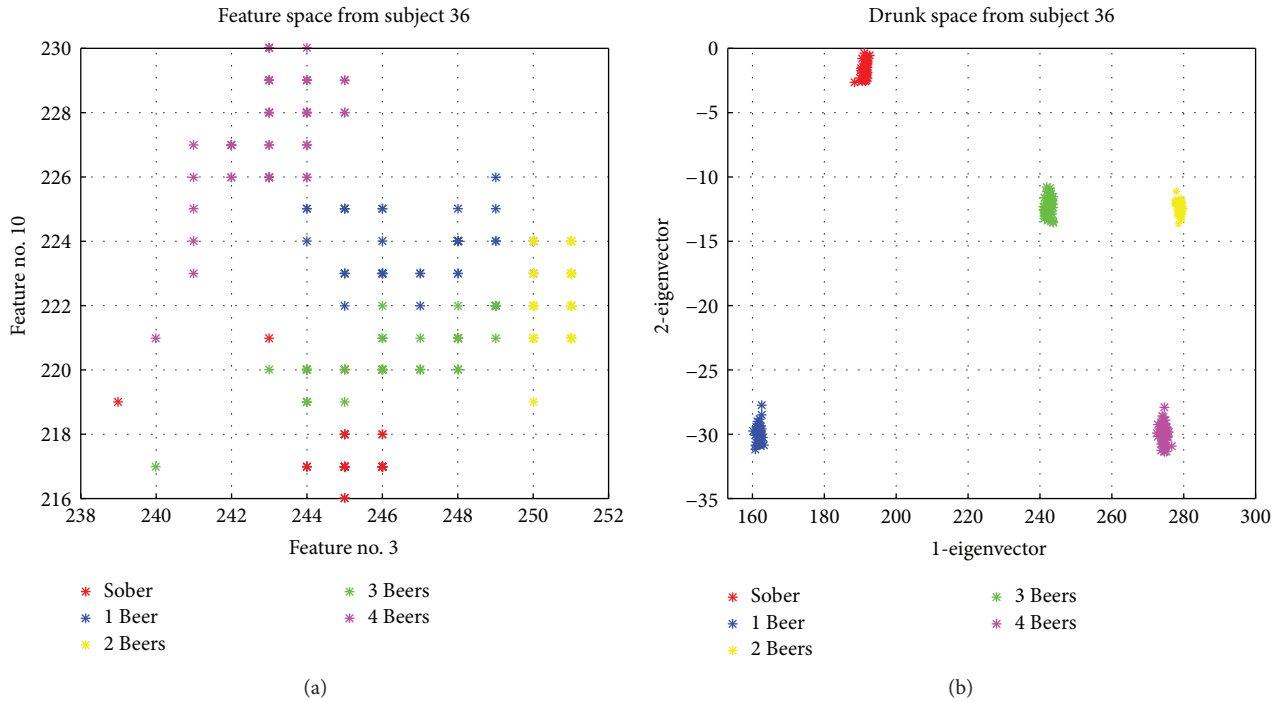


Figure 3: (a) Example of a feature space of two features randomly chosen for a subject. (b) DrunkSpace obtained using FLD for the same subject. The DrunkSpace obtained is completely separable for these two features.

performed by analysing the likelihood that the data belongs to each distribution of the GMM. Finally, the training data are assigned to the distribution (class) to which it is more likely to belong.

The theory on which this approach is based is briefly described below. Let Y be a D-dimensional real-valued random variable with a probability density function (pdf) that is written as a linear combination of elementary pdfs (see (8)). If the distributions that compose the mixture are Gaussian, the pdf is known as a Gaussian mixture.

$$p\frac{y}{\theta} = \sum_{i=1}^{I} a_i N(y \mid C = i, \beta_i), \tag{9}$$

where I represents the quantity of elementary components (C) of the mixture and θ represents the set of parameters $\theta = \{\alpha_1, \ldots, \alpha_I, \beta_1, \ldots, \beta_I\}$, where $\beta = \{\beta_1, \ldots, \beta_I\}$ is the set of parameters associated to each distribution that composes the mixture and $\alpha = \{\alpha_1, \ldots, \alpha_I\}$ is the weight of each distribution of the mixture. The Gaussian density is the components of the mixture, and the mean and covariance $\beta_i = \{\mu_i, \Sigma_i\}$ are represented by

$$N(y \mid C = i, \beta_i) = \frac{1}{2\pi^{D/2}|\Sigma_i|^{1/2}} \cdot \exp\left(-\frac{1}{2}(y - \mu_i)^t \sum_{i}^{-1}(y - \mu_i)\right). \tag{10}$$

The solution to the classification problem is explained below. First, consider a set of samples $y = \{y_1, \ldots, y_J\}$ where $y_j \in \mathrm{R}^{\mathrm{D}}$ is one of the j independent outcomes of the random variable Y; then, the likelihood of y is defined by the following likelihood function (for independent and identically distributed observations) given by (11).

$$L(\theta) = \prod_{j=1}^{J} p\left(y_j \mid \theta\right). \tag{11}$$

Now, the likelihood of y should be maximised. Using some algebraic transformations (see [17]), it is possible to represent the likelihood function as an objective function to be maximised, applied to the Gaussian Mixture Model (12):

$$\theta^* = \arg \max_{\theta} \sum_{j=1}^{J} \log\left\{\sum_{i=1}^{I} a_i N(y \mid C = i, \beta_i)\right\}. \tag{12}$$

This is a hard optimisation problem that is commonly solved using the expectation maximisation (EM) algorithm [18]. More information about Gaussian mixture models is available in [12, 13]. In the present study, the GMM procedure is implemented in Matlab R2015a and the EM algorithm was selected to perform the optimisation. Each Gaussian distribution (of the mixture) is defined by the clusters formed by the projected (FLD) samples of the different training classes ("Sober," "1 Beer," "2 Beers," and "3 Beers"), and the test data, which we want to classify, is the projected samples of the class "4 Beers."

Figure 4 shows an example of a GMM classifier for the same subject used in Figure 3. The figure shows the DrunkSpace classifier generated with the data from the training subsets: "Sober," "1 Beer," "2 Beers," and "3 Beers." The regions shown in Figure 4 were constructed evaluating the DrunkSpace projections from Fisher in the

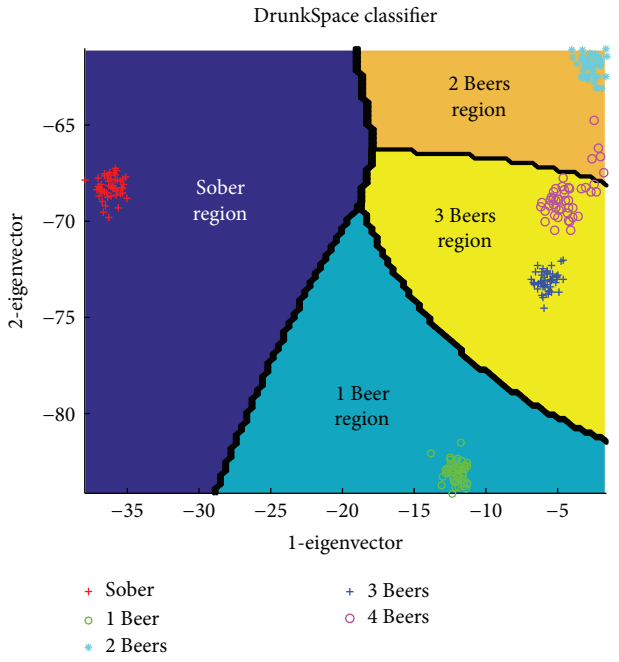


Figure 4: Example of a DrunkSpace classifier from subject 36. The circles represent the test subset "4 Beers." The probabilities obtained for this example are 0% for the "Sober" class, 0% for the "1 Beer" class, 16% for the "2 Beers" class, and 84% for the "3 Beers" class, indicating a high probability of identifying the subject as being in a drunk state.

GMM classifier for the training subsets; thus, one DrunkSpace classifier was generated. Note that the regions obtained represent the probability of being in one of the classes: "Sober," "1 Beer," "2 Beers," and "3 Beers." The subset "4 Beers" is used to validate the classification. Figure 4 shows the test set ("4 Beers") in magenta, which is classified primarily in the region belonging to the class "3 Beers." The probabilities obtained for this example are 0% for the "Sober" class, 0% for the "1 Beer" class, 16% for the "2 Beers" class, and 84% for the "3 Beers" class.

4. Experiment and Evaluation of the Proposed System

The proposal for this study consists of two stages: face recognition and identifying a drunk person. In Figure 5, the general outline of the proposed system is shown. The first stage determines the identity of individuals for further classification analysis. Once the faces of the subjects have been recognised in the first stage, the second stage is responsible for performing a feature extraction through the FLD method and then the classification of drunkenness is performed using the GMM classifier. The explanation of each of the steps is shown in detail below. The database used in the study is the PUCV-DTF, which is described in Section 3.

4.1. Stage 1. Face recognition is a crucial stage for the complete system of identifying drunk people because it determines the identity of the individuals in the database. The face recognition system used in this study was analysed using two current descriptors commonly used in the literature: the LBP descriptor [19] and the WLD descriptor [20]. Both methods use the histogram intersection distance (HI) as a dissimilarity measure.

The experiment consisted of using images from the PUCV-DTF database to generate a gallery set and test set. The gallery set is composed of the face images of the subjects in a sober state, while the test set is composed of images of the subjects after drinking beer ("1 Beer," "2 Beers," "3 Beers," and "4 Beers"). The result obtained by the recognition system is shown in Table 3. As can be seen in Table 3, both descriptors used get a 100% yield in the recognition rates for each test set, which is due to the nonexistence of temporal variability in the images of the database, because they were acquired in a lapse of three hours. However, the performance of the face recognition system may decline if the images were acquired in a higher time lapse [21–23].

4.2. Stage 2. Once the identity of the subject is recognised, we proceed to the stage of drunk state identification. To perform this task, the 2nd stage is divided into two substages: feature extraction and classification. The feature extraction substage involves the selection of the proper information from the face, which is then used to determine if the person is sober or drunk. The substage of classification consists of the generation of a classification subspace (called the DrunkSpace), where the extracted features will be projected, and then based on this projection, the classification of the state of the subject will be performed.

As mentioned in Section 3, the feature extraction is performed using a grid of 22 points. The FLD method is then used to reduce the dimensionality of the data by projecting the feature vectors from 22 dimensions to 2 dimensions. The classification space (DrunkSpace) is then generated for each subject based on the Gaussian mixture model approach (see Section 3). Each DrunkSpace was generated using Matlab R2015a. The training data used was that of the projections of the feature vectors from the images of the classes for the sober subject and the subject after drinking 1, 2, and 3 beers. The images of the subject after drinking 4 beers were used as the test set.

The experiment to perform the classification is carried out as follows: the test feature vector, corresponding to the set "4 Beers," is projected onto the DrunkSpace. This projection is performed using the same eigenvectors used to generate the DrunkSpace (FLD) of each subject (using the training sets "Sober," "1 Beer," "2 Beers," and "3 Beers"). Once the test data had been projected, the classifier returned the likelihood of the data which belong to each training class ("Sober," "1 Beer," "2 Beers," and "3 Beers"). The highest likelihood returned by the classifier indicates how the projected test data ("4 Beers") were classified. For instance, if the classifier returned the likelihood of the projected test data to belong to each training class which are 0.1, 0.2, 0.3, and 0.4 ("Sober," "1 Beer," "2 Beers," and "3 Beers," resp.), then it is considered that the projected training data were identified as a drunk subject, with at least 3 beers. The average classification rate for this experiment to all subjects is summarised in Table 4.

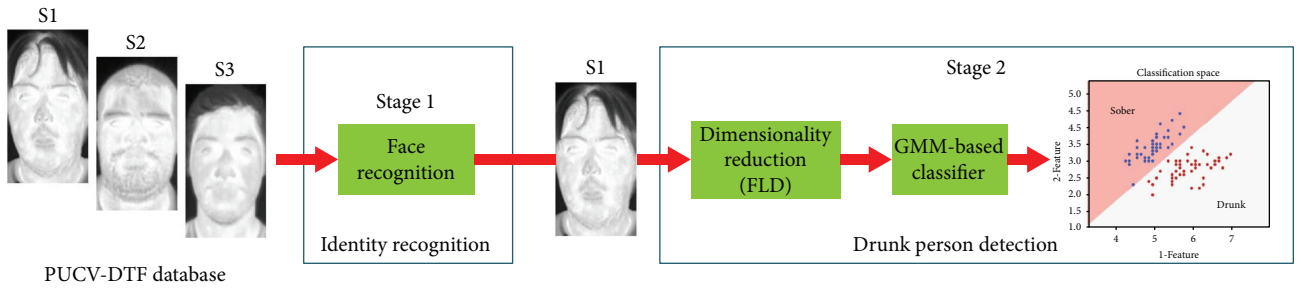


Figure 5: Proposed system outline.

Table 3: Recognition rates obtained by WLD and LBP methods using images from the PUCV-DTF database.

Method	Recognition rate [%]
WLD-HI	100.0
LBP-HI	100.0

Table 4: Results of the classification of drunk people using the PUCV-DTF database.

	Sober	1 Beer	2 Beers	3 Beers
Classification [%]	13.4	9.09	29.39	**48.48**

From the results shown in Table 4, it is important to note that the system achieved an identification rate of 86.96% for drunk people, if we consider correct classification when the training data was identified as 1, 2, or 3 beers. If we break down this 86.96% into the percentages corresponding to each "drunk" class, we observe that a 9.09% was classified in the "1 Beer" and 29.39% was classified in the "2 Beers" class, while 48.48% was classified in the "3 Beers" class. It is important to highlight the clear trend that can be observed relating to the 86.96% correct identification; this trend shows a progressive increase in the classification rates from the class "1 Beer" to the class "3 Beers." This can be attributed to the fact that the test data that are being classified correspond to subjects who drank 4 cans of beer, leading to the conclusion that the projected features in the DrunkSpace follow a certain dynamic and they are moving in certain regions of the DrunkSpace while the subject consumes alcohol. It is because of this that almost half of the test data ("4 Beers") were projected in the region of the DrunkSpace corresponding to the highest level of alcohol consumption.

5. Conclusions

This article presents a computer vision system that identifies people in a drunk state. The system is composed of two main stages, one for face recognition and the other for drunk classification. The face recognition stage provides the identity of an individual previously stored in a database, while the classification stage identifies the state of the individual, indicating if the subject has consumed alcohol. Inspired by [10], the classification stage uses the Fisher linear discriminant (FLD) method to reduce the dimensionality of the feature vectors and generate a subspace called "DrunkSpace." We then use a Bayesian classifier based on Gaussian mixture models (GMM) to identify whether or not the subject is in a drunk state.

The results obtained in this study show that the proposed system to identify drunk people achieves a success rate of approximately 87%; that is, the system is capable of identifying if a person drank at least one can of beer. In addition, the proposed system achieves recognition rates of 100% in the face recognition stage using the LBP method or the WLD method. It is important that the face recognition stage should be robust, because obtaining the correct identity of the test subject allows us to select the correct "DrunkSpace" corresponding to the test subject, facilitating the work for the GMM-based classifier that determines if the individual is in a drunk state.

It is important to mention that the good results obtained are mainly because the selected locations of the extracted features present results of metabolic changes in the face of the subjects and are also due to processes related to other biological factors, such as the thermoregulation, which can be observed with a thermal camera.

From obtained results, we hope to encourage other researchers to study the classification of people in a drunk state, because it would lead to noninvasive systems which can be beneficial to society. As a future study, we hope to generalise the problem of the classification of groups of people, that is, to generate a generic classifier which can be used to identify people in a drunk state independent of weight, sex, or height and not individually as was performed in this research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by FONDECYT under Grant 11130466, Grant 1161584, and Grant 11150476 and in part by Pontificia Universidad Católica de Valparaíso DI Regular Code under Grant 039.420/2017.

References

[1] B. Falk, R. Burstein, J. Rosenblum, Y. Shapiro, E. Zylber-Katz, and N. Bashan, "Effects of caffeine ingestion on body fluid balance and thermoregulation during exercise," *Canadian Journal of Physiology and Pharmacology*, vol. 68, no. 7, pp. 889–892, 1990.

[2] H. Kalant and A. Le, "Effects of ethanol on thermoregulation," *Pharmacology & Therapeutics*, vol. 23, no. 3, pp. 313–364, 1983.

[3] P. Buddharaju, I. T. Pavlidis, P. Tsiamyrtzis, and M. Bazakos, "Physiology-based face recognition in the thermal infrared spectrum," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 613–626, 2007.

[4] P. Buddharaju and I. Pavlidis, "Multi-spectral face recognition - fusion of visual imagery with physiological information," in *Face Biometrics for Personal Identification. Signals and Communication Technology*, R. I. Hammoud, B. R. Abidi and M. A. Abidi, Eds., pp. 91–108, Springer, Berlin, Heidelberg, 2007.

[5] P. Buddharaju, I. Pavlidis, and C. Manohar, "Face recognition beyond the visible spectrum," in *Advances in Biometrics*, pp. 157–180, Springer, London, 2008.

[6] P. Buddharaju, I. Pavlidis, and I. Kakadiaris, "Pose-invariant physiological face recognition in the thermal infrared spectrum," in *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pp. 53–60, New York, USA, 2006.

[7] G. Koukiou and V. Anasassopoulos, "Face locations suitable drunk persons identification," in *2013 International Workshop on Biometrics and Forensics (IWBF)*, pp. 1–4, Lisbon, Portugal, 2013.

[8] G. Koukiou and V. Anastassopoulos, "Facial blood vessels activity in drunk persons using thermal infrared," in *4th International Conference on Imaging for Crime Detection and Prevention 2011 (ICDP 2011)*, pp. 1–5, London, UK, 2011.

[9] G. Koukiou and V. Anastassopoulos, "Eye temperature distribution in drunk persons using thermal imagery," in *2013 International Conference of the BIOSIG Special Interest Group (BIOSIG)*, pp. 233–240, Darmstadt, Germany, 2013.

[10] G. Koukiou and V. Anastassopoulos, "Drunk person identification using thermal infrared images," *International Journal of Electronic Security and Digital Forensics*, vol. 4, no. 4, pp. 229–243, 2012.

[11] G. Koukiou and V. Anastassopoulos, "Neural networks for identifying drunk persons using thermal infrared imagery," *Forensic Science International*, vol. 252, pp. 69–76, 2015.

[12] J.-M. Marin, K. Mengersen, and C. P. Robert, "Bayesian modelling and inference on mixtures of distributions," *Handbook of Statistics*, vol. 25, pp. 459–507, 2005.

[13] B. G. Lindsay, "Mixture models: theory, geometry and applications," in *NSF-CBMS Regional Conference Series in Probability and Statistics*, vol. 5, p. i-iii+v-ix+1-163, Institute of Mathematical Statistics, Hayward, CA, USA, 1995.

[14] FLIR, "Tau 2 product specification," 2014, http://cvs.flir.com/tau2-product-spec.

[15] J. Rustemeyer, J. Radtke, and A. Bremerich, "Thermography and thermoregulation of the face," *Head & Face Medicine*, vol. 3, no. 1, p. 17, 2007.

[16] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[17] A. García Herrero, *Algoritmos para la estimación de modelos de mezclas Gaussianas*, Universidad de Cantabria, Spain, 2015.

[18] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[19] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[20] J. Chen, S. Shan, C. He et al., "WLD: A robust local image descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1705–1720, 2010.

[21] S. Farokhi, S. M. Shamsuddin, J. Flusser, and U. U. Sheikh, "Assessment of time-lapse in visible and thermal face recognition," *International Journal of Computers Communications & Control*, vol. 6, pp. 181–186, 2012.

[22] D. A. Socolinsky and A. Selinger, "Thermal face recognition over time," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*, pp. 187–190, Cambridge, UK, 2004.

[23] X. Chen, P. J. Flynn, and K. W. Bowyer, "IR and visible light face recognition," *Computer Vision and Image Understanding*, vol. 99, no. 3, pp. 332–358, 2005.

Hindawi
Journal of Sensors
Volume 2017, Article ID 8231314, 8 pages
https://doi.org/10.1155/2017/8231314

Research Article

Automatic Detection Technology of Sonar Image Target Based on the Three-Dimensional Imaging

Wanzeng Kong,[1] Jinshuai Yu,[1] Ying Cheng,[1] Weihua Cong,[2] and Huanhuan Xue[2]

[1]*College of Computer Science, Hangzhou Dianzi University, Hangzhou, China*
[2]*Hangzhou Institute of Applied Acoustics, Hangzhou, China*

Correspondence should be addressed to Wanzeng Kong; kongwanzeng@hdu.edu.cn

Received 28 April 2017; Revised 25 July 2017; Accepted 20 August 2017; Published 11 October 2017

Academic Editor: Paolo Mercorelli

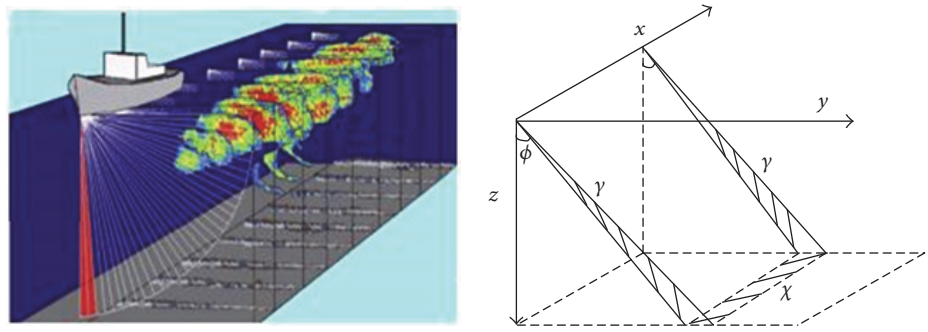
With 3D imaging of the multisonar beam and serious interference of image noise, detecting objects based only on manual operation is inefficient and also not conducive to data storage and maintenance. In this paper, a set of sonar image automatic detection technologies based on 3D imaging is developed to satisfy the actual requirements in sonar image detection. Firstly, preprocessing was conducted to alleviate the noise and then the approximate position of object was obtained by calculating the signal-to-noise ratio of each target. Secondly, the separation of water bodies and strata is realized by maximum variance between clusters (OTSU) since there exist obvious differences between these two areas. Thus image segmentation can be easily implemented on both. Finally, the feature extraction is carried out, and the multidimensional Bayesian classification model is established to do classification. Experimental results show that the sonar-image-detection technology can effectively detect the target and meet the requirements of practical applications.

1. Introduction

In the Second World War, the US Navy successfully escaped the Japanese seabed minefield by using the sonar equipment. Subsequently, many countries have begun to pay attention to the development of sonar technology, especially in the military field. With the increasing marine development, the exploration of the ocean was limited not only to military purposes but also to commercial and civilian purposes, such as submarine resource development, oil exploration, automatic mapping of submarine topography, and detection of fish stocks. In order to adapt to the underwater environment, intelligent underwater robot research has been carried out for the laying of submarine cables, underwater demining, and so on. As the current requirements for intelligent sonar equipment are getting higher and higher, now there are many underwater target recognition technology applications. Therefore, whether in the field of military or civilian, underwater target recognition technology will be one of the main technologies in the future of ship and ocean engineering to research.

Due to the complexity of the seabed environment, there are some problems such as poor contrast of the sonar image, the serious noise of the submarine reverberation, low resolution, strong interference, and less dark pixels [1], in addition to the object's inherent characteristics of reflecting sound waves which result in the incomplete target edge. Since there is too much noise in the sonar image, it is difficult to ensure the correctness of submarine target recognition if we only consider the visual point. As a result, it is easy to produce false alarms. Therefore, the development of a set of automatic detection technologies for sonar images which has low false alarm rate and high detection rate is particularly important [2].

Casselman et al. proposed a method of multilayer perceptual neural network detection with low false alarm rate and applied this method to the passive sonar detection [3]; Weber and Kruger studied the passive sonar lofar grams contour-enhancing technique and contrast enhancement technique based on neural network, which greatly improved the signal-to-noise ratio of the image and detected the target signal with signal-to-noise ratio of −17 dB with very accurate precision



(a) (b)

Figure 1: The composition of three-dimensional image data volume.

[4]; Howell and Wood proposed a passive sonar detection and recognition system based on hybrid neural networks [5]; Postolache et al. constructed intelligent passive sonar signal processing systems on Lab VIEW and FPGA platforms [6]. Mill Brown, of the University of Sheffield, UK, studied three time detectors based on the zero-crossing rate of the time-domain signal and the weak underwater acoustic signal was detected. Bertilone and Killeen [7], of the Royal Australian Naval Ship Defense Science and Technology Organization, use the passive band gap generalized energy detection hybrid model to detect the underwater noise, and its false alarm rate is reduced to less than 1%. Compared with the traditional energy detection method, the threshold is reduced by 8 dB.

2. The Composition of Three-Dimensional Image Data Volume

The three-dimensional sonar data consists of three dimensions: navigation, distance, and beam. When vessels sail at sea, sonar is used to detect underwater objects. The direction of the sailing vessel is the navigational dimension of the sonar data. The distance from the sonar to the bottom of the sea is the distance dimension of the sonar data. The angle of the acoustic wave divergence in the underwater is the beam dimension of the sonar data. In addition, the distribution of the beam is uniform. Since the three-dimensional sonar data can obtain the information of the three-dimensional space object, the image is clearer and more visible than the current image sonar. Three-dimensional sonar data "$\rho(\gamma, \phi, \chi)$" can transform to $P(x, y, z)$; navigation direction "χ" is the sequence of equal interval of 8 (sequence length can be adjusted, the maximum is 4096); heading angle ϕ is $[-40^\circ, 40^\circ]$; echo distance γ is the sequence of equal interval (sequence length can be adjusted, the maximum is 4096). Lateral horizontal width $Y = \gamma * \sin\phi$; vertical depth $Z = \gamma * \cos\phi$. Thus, the three-dimensional sonar image is made up of numerous beam's two-dimensional images [8, 9]. The component schematic diagram of three-dimensional sonar data is shown in Figure 1.

3. Sonar Image Processing

3.1. Preprocessing and Signal-to-Noise Ratio (SNR). Sonar image preprocessing is the foundation of the sonar image processing and an indispensable part of image recognition [10–13]. Image preprocessing mainly includes image denoising and enhancement. The function of sonar image preprocessing must be maximum to eliminate the influence of all kinds of noise, reduce the influence of noise on the target area, and, at the same time, strengthen the real target image in the water and the part of interest. The background noise of sonar image is divided into three categories, respectively, the environmental noise, reverberation noise, and white noise. In this paper, the sonar image preprocessing includes two steps: first, image denoising; second, bleaching process of sonar signal.

Image denoising methods mainly include neighborhood averaging method, self-adaptive smooth filtering, wavelet transformation denoising method, and median filtering denoising. According to the characteristics of the noise, we can choose one or more targeted denoising methods. In this paper, through the experimental analysis and judgment, we choose the median filtering method to the sonar image processing. The median filter is a kind of nonlinear filters; it can eliminate noise and at the same time keep the detail of the image. The steps are as follows:

(1) Moving the template in turn in the image until the center of the template overlaps with a pixel in the image

(2) Extracting the gray value of the pixels which correspond with the template

(3) Aligning these value in order

(4) Assigning the intermediate value or the average of the two middle values to the central pixel in the corresponding template.

In order to eliminate the isolated noise points in the image, we can make the approximate values replace the pixel values that are significantly different from the surrounding pixels by using the median filter.

Bleaching process of sonar signal refers to the balanced background which can reduce the interference of noise to a certain extent so that the data will be easy to deal with for subsequent processing. According to the characteristics

and properties of the sonar, choice of sonar signal bleaching processing formula is as follows:

$$X(t) = \int_{t-T/2}^{t+T/2} |x(t)|^{4/5}\, dt,$$
$$y(t) = \frac{x(t)}{X(t)}. \quad (1)$$

$x(t)$ refers to data of navigation direction or echo distance. T is the size of the window. Through the bleaching process, the noise signal is weakened obviously which is in favor of the image target recognition.

The signal-to-noise ratio is the ratio of the power spectrum of the signal to the noise, but power spectra are usually hard to measure. Therefore we can use the ratio of signal variance to noise variance to approximate the signal-to-noise ratio. According to the characteristics of sonar image, we can use the following formula to calculate the target's signal-to-noise ratio:

$$\mathrm{SNR} = 10 * \log \frac{\overline{S} - E_n}{V_{\mathrm{arn}}}. \quad (2)$$

$\overline{S}$ is the pixel integral value of the target area in sonar image, E_n is the local mean in the target area, and V_{arn} is the standard deviation of background in the target area. The larger the signal-to-noise ratio in the suspicious area, the greater the chance that the area turns out to be the true target. Sonar target detection technology calculates the signal-to-noise ratio of the suspicious target area, so as to achieve the purpose which is distinguishing the target and interference noise effectively, and largely avoid the noise interference.

3.2. Image Segmentation Algorithm Analysis. In the actual detection of the ocean, sonar needs not only to detect the object in the water but also to detect objects in the stratum. Due to the large differences between the water and stratum, the detection is much easier since there is less noise in the sea water, while the reverberation of stratum is relatively serious, which causes great difficultly in detecting the target. So we can separate the water and stratum and then process the sonar image in different regions, respectively.

In this paper, we use the OTSU algorithm based on the gray-level histogram to proceed image threshold segmentation. A good result of segmentation is obtained by using the corrosion expansion operator to process closure operation.

OTSU threshold segmentation method uses variance to find the best threshold value between the two kinds of pixels and uses variance between clusters to evaluate the segmentation results. The formula is as follows:

$$w(R) = \frac{\sum_{p\in R} f(x_1, y_1)^2}{\mathrm{Num}_t} + \frac{\sum_{q\in R} f(x_2, y_2)^2}{\mathrm{Num}_{\bar{t}}}. \quad (3)$$

Num_t and $\mathrm{Num}_{\bar{t}}$ represent the pixel number of the target area and nontarget area, respectively.

Image segmentation is an important part of sonar image processing [14, 15]. After the separation of water and stratum, the two parts of the image need to be segmented and then the target area is extracted. Since there is less noise in the water, there are many methods available, such as OTSU, iterative threshold method, and two-dimensional maximum entropy threshold segmentation. After experimental analysis, we use the iterative threshold method for threshold selection compared with two-dimensional maximum entropy threshold segmentation. The iterative method is based on the idea of approximation, and its steps are as follows:

(5) The maximum grayscales value and the minimum grayscales value of the image are denoted as $P_{\max}$ and $P_{\min}$, and the initial threshold value $T_0 = (P_{\max} + P_{\min})/2$.

(6) According to the threshold value $T_{(k)}$ ($k = 0, 1, 2, \ldots, k$), the image is divided into foreground and background, and mean gray values of the two parts are denoted as H_1 and H_2.

(7) A new threshold value $T_{(k+1)} = (H_1 + H_2)/2$ is obtained.

(8) If $T_{(k)} = T_{(k+1)}$, then the result is the threshold; otherwise go to (2), iterative calculation.

Entropy is a function that describes the state of the system and represents the average amount of information. At the same time, entropy is used to calculate the disorder in a system phenomenon and can be used as a measure of the degree of chaos. When performing the evaluation of the segmentation effect, the formula is as follows:

$$w(R) = \frac{\sum_{i=0}^{ng} H_t(i) \ln H_t(i)}{\mathrm{Num}_t} - \frac{\sum_{i=0}^{ng} H_{\bar{t}}(i) \ln H_{\bar{t}}(i)}{\mathrm{Num}_{\bar{t}}} + \ln \mathrm{Num}_t \mathrm{Num}_{\bar{t}}. \quad (4)$$

In the sonar image processing, the two-dimensional maximum entropy is often used to divide the threshold to obtain the optimal threshold. The two-dimensional maximum entropy method uses the two-dimensional histogram of pixel intensity and regional gray mean and finds the optimal threshold according to the maximum entropy.

3.3. Sonar Target Discrimination Based on Multidimensional Bayesian Classification Model. In fact, we need to not only detect the submarine target but also classify the target preliminary during the sonar sweeping. The main categories are cylindrical, spherical, cable-like targets. In this paper, feature extraction is used to obtain the characteristics of the target. After multiple experiments, the extracted length-width ratio, pixel value, and signal-to-noise ratio are analyzed to find that they conform to the Gaussian distribution. Therefore, this paper adopts a multidimensional Bayesian classification model [8] to classify the target. Bayesian classifier has the advantages of high classification accuracy and generalization ability [16]. At the same time, it does not need a lot of data as the training set. The formula is as follows:

$$P(w_i \mid x) = \frac{P(w_i \mid x)}{P(x)}. \quad (5)$$

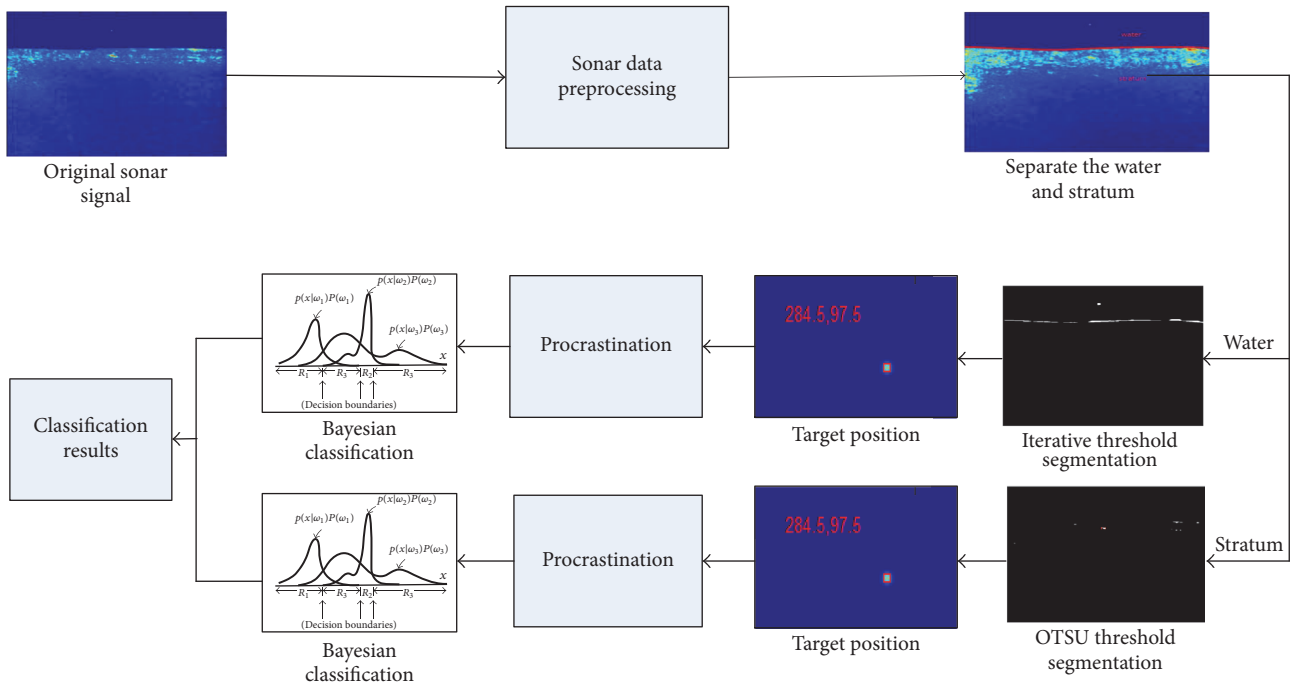


Figure 2: Sonar image processing flow chart.

$P(x \mid w_i)$ is the likelihood function, $P(w_i)$ is the prior probability, $\sum_j P(w_j)P(x \mid w_j)$ is the evidence factor, and $P(w_i \mid x)$ is the posterior probability. The complete naive Bayesian classification model is the simplest Bayesian classification model. Its class subgraph and attribute subgraph are empty sets, and each class variable and attribute variable are connected by a directed edge. This method is based on a simple assumption that class variables are independent of each other. The multidimensional Bayesian model is defined as follows:

$$\rho(c_1, c_2, \ldots, c_m \mid u_1, u_2, \ldots, u_n) = \alpha \sum_{i=1}^{m} \sum_{j=1}^{n} \rho(c_i)\, \rho(u_j \mid c_1, c_2, \ldots, c_m). \quad (6)$$

α is the regularization parameter, $U = (u_1, u_2, \ldots, u_n)$ is unclassified data, and $c = (c_1, c_2, \ldots, c_m)$ is the arbitrary value of the class variable. We can achieve a relatively accurate result of classifications which can meet the practical application by using multidimensional Bayesian classifier.

3.4. Method Summary. Figure 2 is a sonar image processing flow chart. The sonar target automatic detection based on three-dimensional imaging includes a series of processes, such as sonar imaged preprocessing, noise exclusion, separation of water and stratum, image segmentation, and target discrimination. Using the sonar image preprocessing and SNR, we can effectively achieve the goal of denoising and reinforce. The image difference is so obvious owing to the differences between water and stratum that the OTSU algorithm is used to separate the water and stratum. Since water images generally contain less noise interference, we can choose OTSU algorithm and iterative threshold method for image segmentation. However, the noise of the stratum images is serious. To obtain a finer effect of segmentation, the iterative threshold method is a better choice. Then, we extract the characteristic values of the target, such as length-width ratio, pixel value, and signal-to-noise ratio. Finally, the multidimensional Bayesian classification model is used to classify the target and realize the automatic detection of the whole sonar image. The flow chart is as in Figure 2.

4. Experiment

In order to verify the effectiveness and superiority of the proposed algorithm, this paper carries out simulation experiments on the three-dimensional imaging sonar data. The experimental data is all derived from Hangzhou Institute of Applied Acoustics. Firstly, check the noise suppression effect of sonar signal after bleaching process. Then the OTSU algorithm is used to separate the water and stratum so that the interface is found out. The image segmentation algorithm simulation experiment is carried out in the light of the characteristics of the sonar image of each part. The results of the two image segmentation methods are compared. The two methods are the iterative threshold method and the two-dimensional maximum entropy algorithm. Finally, the results of the classification are summarized by using the multidimensional Bayesian classification model.

4.1. Simulation Experiment of Sonar Signal Bleaching Process. Figure 3 is a comparison of the sonar signals before and after bleaching treatment. After the bleaching treatment, the background of the stratum's image is more balanced. Noise interference to a certain extent weakened which is conducive to the subsequent processing of the image.

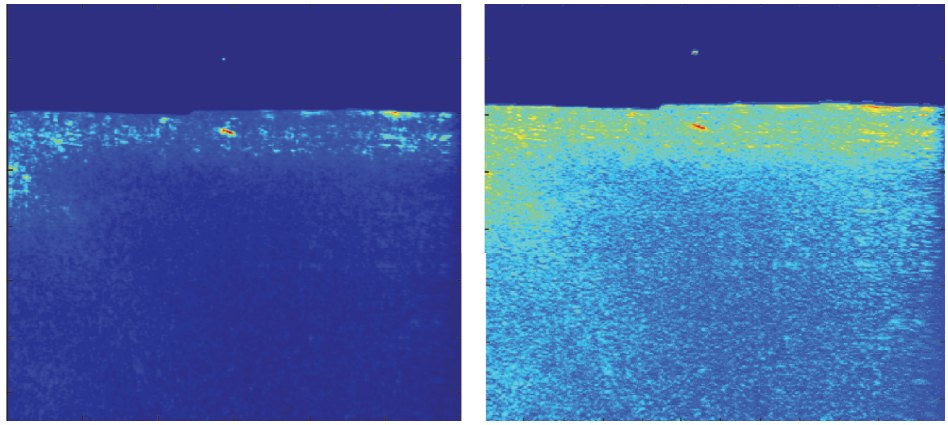


Figure 3: The comparison of the sonar signals before and after bleaching.

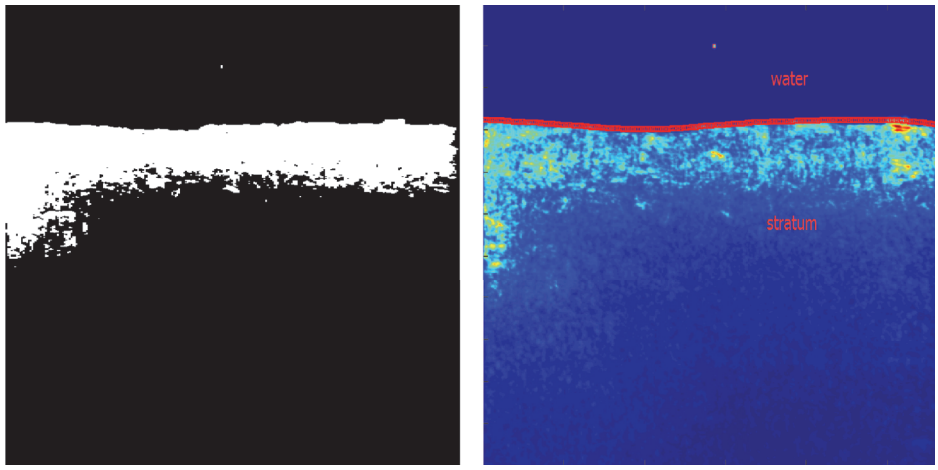


Figure 4: The results of water and stratum separation using OTSU.

4.2. Simulation Experiment of Water and Stratum Separation. Figure 4 is a graph of water and stratum separation; the left image is the OTSU binarized image. After binarization, it is clear to see the interface of water and stratum in the left image. We can effectively separate the water and stratum through mathematical morphology, such as corrosion expansion and opening and closing operations. The right image is the rendering of the segmentation. The red line in the figure is the interface, the upper part of the interface is the water, and the lower part is the stratum.

4.3. Comparison of Iteration Threshold and 2D Maximum Entropy Image Segmentation. Figures 5(a) and 5(b) are the images of the water and the stratum after image segmentation using the iterative threshold method. Figures 5(c) and 5(d) are the images of the water after image segmentation using the two-dimensional maximum entropy method. It is easy to find out that both the two methods can detect the spherical target in the water by the comparison of Figures 5(a) and 5(c), but the two-dimensional maximum entropy is easily disturbed by the interface. It can be seen from the comparison of Figures 5(b) and 5(d) that, in the segmentation of the stratum image, Figure 5(b) can better avoid the interference of the noise which is in favor of the detection of the target, but the noise interference is still more serious in the image using the two-dimensional maximum entropy threshold. In summary, the iterative threshold method has a better effect.

4.4. Multidimensional Bayesian Classification Model and K-Nearest Neighbor Target Classification Contrast Test. For the object classification of the sonar image, the most commonly used methods are the K-nearest neighbor method and the multidimensional Bayesian classification model method. Although K-nearest neighbor method is simple and intelligible, its computational complexity is too high to apply to multiclassification tasks. In reality, submarine targets are different not only in shape but also in distribution. The multidimensional Bayesian classification model can accommodate small-scale data samples. At the same time, the multidimensional Bayesian classification model can adapt to multiclassification tasks and incremental training. Therefore, it is reasonable to use the multidimensional Bayesian classification model to classify the targets.

Figure 6 shows the experimental results using the K-nearest neighbor method and the multidimensional Bayesian classification model within 40 images. Figure 7 is the

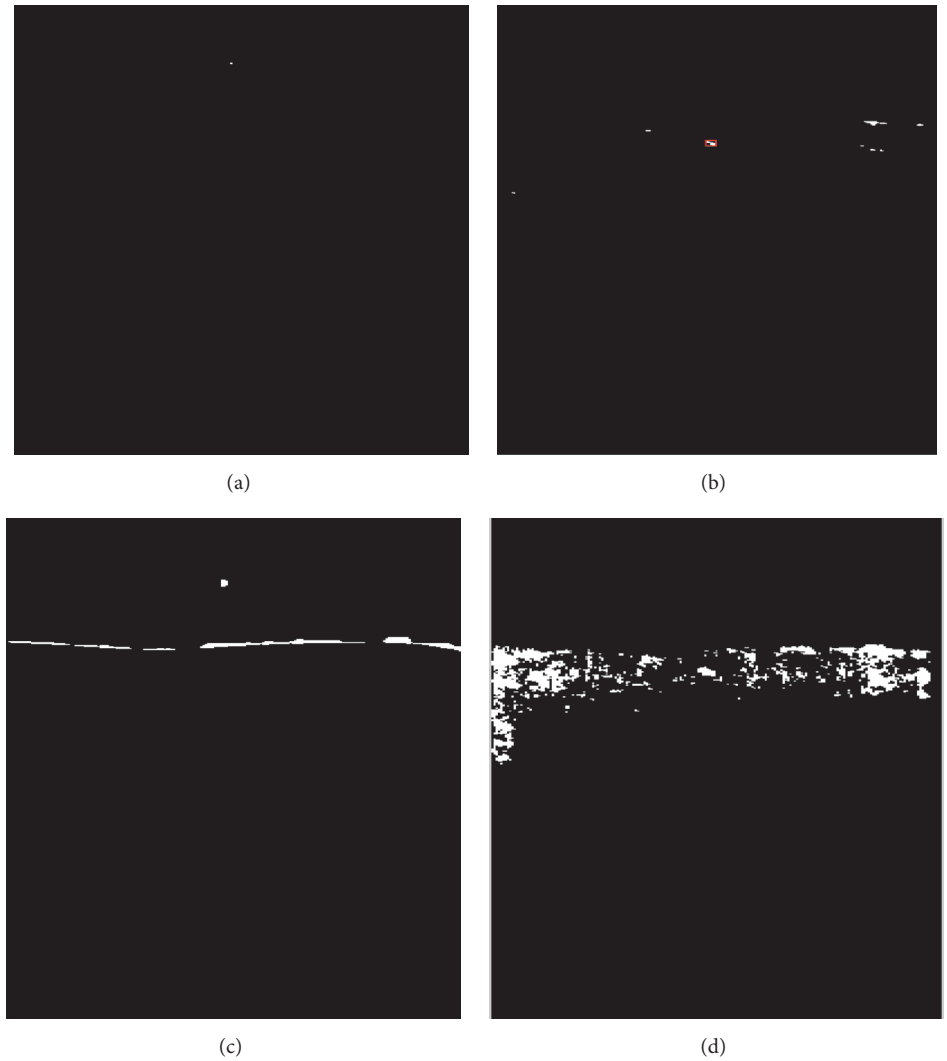


Figure 5: The comparison between the method we used and the two-dimensional maximum entropy.

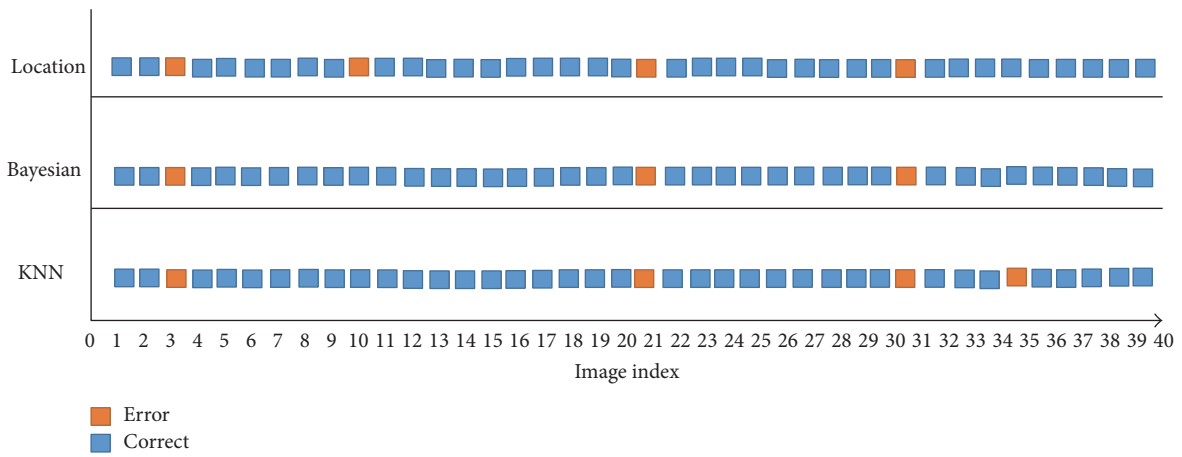


Figure 6: The classification result of the multidimensional Bayesian model and K-nearest neighbor method on a sonar dataset.

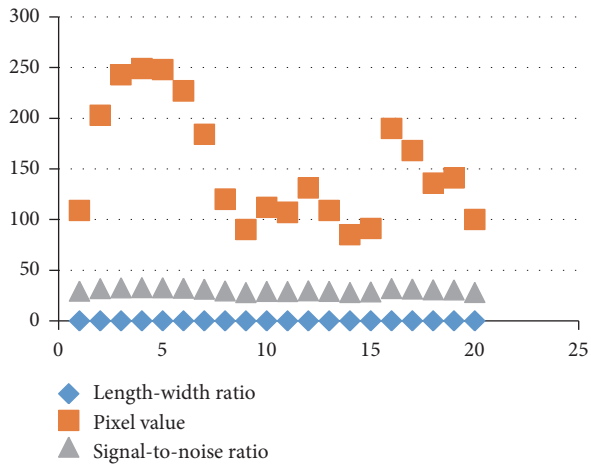


Figure 7: The characteristic distribution of columnar targets.

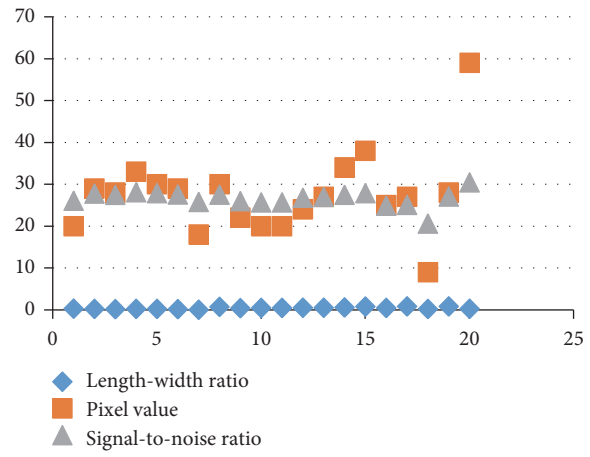


Figure 8: The characteristic distribution of spherical objects.

characteristic distribution of the columnar targets. Figure 8 is the characteristic distribution of the spherical targets. Classification of the columnar and spherical targets is based on the extracted length-width ratio, pixel value, and signal-to-noise ratio. According to the experimental results of the two classification methods, there are only three misjudgments existing within the 40 sonar images by using the multidimensional Bayesian classification model. Otherwise, there are four misjudgments existing within the 40 sonar images by using the K-nearest neighbor method. It can be seen from the experimental results that the multidimensional Bayesian classification model and the K-nearest neighbor method can basically satisfy the requirements of the sonar image target classification because of the large differences in characteristics between the columnar and spherical target. However, given the small number of submarine target samples, the multidimensional Bayesian classifier can better meet the needs of target classification.

5. Conclusions

In this paper, we achieve the automatic detection technology of target in the sonar image based on three-dimensional imaging. This technology includes a complete set of processes such as sonar image preprocessing, water and stratum separation, image threshold segmentation, and target classification recognition. At the same time, a better sonar image processing result is achieved, which satisfies the actual processing requirements of the sonar image. Besides, this technology not only improves the efficiency of the sonar image target detection and reduces the unnecessary labor, but also improves accuracy of the sonar image target detection to a certain extent and creates a low false alarm rate and high detection rate of the sonar target detection technology. This technology has been applied in Hangzhou Institute of Applied Physics.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant nos. 61671193 and 61102028), International Science & Technology Cooperation Program of China (Grant no. 2014DFG12570), and project funded by China Postdoctoral Science Foundation (Grant no. 2015M571878).

References

[1] J. Luo, H. Liu, C. Huang, J. Gu, S. Xie, and H. Li, "Denoising and tracking of sonar video imagery for underwater security monitoring systems," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO '13)*, pp. 2203–2208, IEEE, Shenzhen, China, December 2013.

[2] I. Mandhouj, H. Amiri, F. Maussang, and B. Solaiman, "Sonar image processing for underwater object detection based on high resolution system," in *Proceedings of the Workshop on Signal and Document Processing (SIDOP '12)*, vol. 845, Hammamet, Tunisia, March 2012.

[3] F. L. Casselman, D. F. Freeman, D. A. Kerrigan, S. E. Lane, N. H. Millstrom, and W. G. Nichols Jr., "A neural network-based passive sonar detection and classification design with a low false alarm rate," in *Proceedings of the IEEE Conference on Neural Networks for Ocean Engineering*, pp. 49–55, IEEE, Washington, DC, USA, August 1991.

[4] D. M. Weber and C. C. Kruger, "Detection of tonals in Lofargrams using connectionist methods," in *Proceedings of the 1993 IEEE International Conference on Neural Networks*, pp. 1662–1666, IEEE, San Francisco, Claif, USA, April 1993.

[5] B. P. Howell and S. Wood, "Passive sonar recognition and analysis using hybrid neural networks," in *Proceedings of the IEEE Celebrating the Past—Teaming Toward the Future*, pp. 1917–1924, San Diego, Claif, USA, September 2003.

[6] O. Postolache, P. Girao, and M. Pereira, "Underwater acoustic source localization based on passive sonar and intelligent processing," in *Proceedings of the IEEE Instrumentation & Measurement Technology Conference (IMTC '07)*, pp. 1–4, IEEE, Warsaw, Poland, May 2007.

[7] D. C. Bertilone and D. S. Killeen, "Statistics of biological noise and performance of generalized energy detectors for passive detection," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 2, pp. 285–294, 2001.

[8] J. Chen, Z. Gong, H. Li, and S. Xie, "A detection method based on sonar image for underwater pipeline tracker," in *Proceedings of the 2nd International Conference on Mechanic Automation and Control Engineering (MACE '11)*, pp. 3766–3769, Hohhot, China, July 2011.

[9] H. Cho, J. Gu, H. Joe, A. Asada, and S.-C. Yu, "Acoustic beam profile-based rapid underwater object detection for an imaging sonar," *Journal of Marine Science and Technology*, vol. 20, no. 1, pp. 180–197, 2015.

[10] L.-Y. Weng, M. Li, Z. B. Gong, and S. G. Ma, "Underwater object detection and localization based on multi-beam sonar image processing," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO '12)*, pp. 514–519, IEEE, Guangzhou, China, December 2012.

[11] M. López, J. Ramírez, J. M. Górriz et al., "Automatic system for Alzheimer's disease diagnosis using eigenbrains and bayesian classification rules," in *Bio-Inspired Systems: Computational and Ambient Intelligence*, vol. 5517 of *Lecture Notes in Computer Science*, pp. 949–956, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[12] A.-A. Saucan, C. Sintes, T. Chonavel, and J.-M. Le Caillec, "Model-based adaptive 3D sonar reconstruction in reverberating environments," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 2928–2940, 2015.

[13] I. Mandhouj, F. Maussang, B. Solaiman, and H. Amiri, "Sonar image preprocessing method based on homomorphic filtering," in *Proceedings of the OCEANS 2012 MTS/IEEE Hampton Roads Conference: Harnessing the Power of the Ocean*, Hampton Roads, Va, USA, October 2012.

[14] X.-F. Ye, Z.-H. Zhang, P. X. Liu, and H.-L. Guan, "Sonar image segmentation based on GMRF and level-set models," *Ocean Engineering*, vol. 37, no. 10, pp. 891–901, 2010.

[15] Y. Han, "A detection method for underwater pipeline tracker based on sonar image," *Computer Measurement & Control*, vol. 23, no. 2, pp. 539–541, 2015.

[16] H. B. Shi, Z. H. Wang, H. K. Huang et al., "A restricted double-level bayesian classification model," *Journal of Software*, vol. 15, no. 2, 2004.

Hindawi
Journal of Sensors
Volume 2017, Article ID 4562934, 11 pages
https://doi.org/10.1155/2017/4562934

Research Article

Vision System of Mobile Robot Combining Binocular and Depth Cameras

Yuxiang Yang, Xiang Meng, and Mingyu Gao

Department of Electronics and Information, Hangzhou Dianzi University, Hangzhou, China

Correspondence should be addressed to Mingyu Gao; mackgao@hdu.edu.cn

Received 27 April 2017; Revised 12 July 2017; Accepted 16 August 2017; Published 24 September 2017

Academic Editor: Wendy Flores-Fuentes

In order to optimize the three-dimensional (3D) reconstruction and obtain more precise actual distances of the object, a 3D reconstruction system combining binocular and depth cameras is proposed in this paper. The whole system consists of two identical color cameras, a TOF depth camera, an image processing host, a mobile robot control host, and a mobile robot. Because of structural constraints, the resolution of TOF depth camera is very low, which difficultly meets the requirement of trajectory planning. The resolution of binocular stereo cameras can be very high, but the effect of stereo matching is not ideal for low-texture scenes. Hence binocular stereo cameras also difficultly meet the requirements of high accuracy. In this paper, the proposed system integrates depth camera and stereo matching to improve the precision of the 3D reconstruction. Moreover, a double threads processing method is applied to improve the efficiency of the system. The experimental results show that the system can effectively improve the accuracy of 3D reconstruction, identify the distance from the camera accurately, and achieve the strategy of trajectory planning.

1. Introduction

With the development of the society, the application of the robots is rising rapidly in the society [1]. With the increase of the application field, the accuracy of the robot movement and the stability of the movement are very necessary for the promotion of the robot. That is how to improve the accuracy of the robot has been widely concerned, and more and more schools and companies have been involved in the research on how to improve the stability of the robot. 3D imaging technologies directly affect the accuracy of the robot movement. There exists a variety of 3D imaging technologies to acquire depth information about our world. In general, they can be categorized into two major classes: stereo matching methods and direct depth measuring methods.

Stereo vision [2–4] mainly researches on how to use imaging technology to obtain the distance information of the objects in the scene from the images with different views. At present, the basic method of stereo vision is to obtain the same scene image from two or more different perspectives and to obtain the corresponding disparity between pixels in different images and then it is simple to derive the target space position in the scene through the principle of triangulation. Binocular stereo vision simulates human visual system, using two 2D images of the same scene obtained by two cameras from different point of views. The goal of binocular stereo matching is to find out the corresponding matching points in the left and right image after calibration. The difference in X coordinate of the corresponding points is named as disparity, which can form a disparity map. The key objective of stereo matching methods is to reduce the matching ambiguities introduced by low-texture regions, and they can be generally classified into two categories: local matching methods and global matching methods. Researchers have done a wealth of work on stereo matching in the past few years. But, unfortunately, the fundamental problems in stereo such as occlusion and texture-less regions remain to be unsolved.

Direct depth measuring camera [5, 6] mainly uses TOF ranging principle. The system illumination light source emits light beam to the scene, and detection sensor detects reflected light signal and figures out the propagation time of the optical signal. Finally, spatial coordinates of the object are calculated by the relative position of the optical signal propagation velocity and light source and sensor. These sensors measure time delay between transmission of a light pulse and detection of the reflected signal on an entire frame once by using

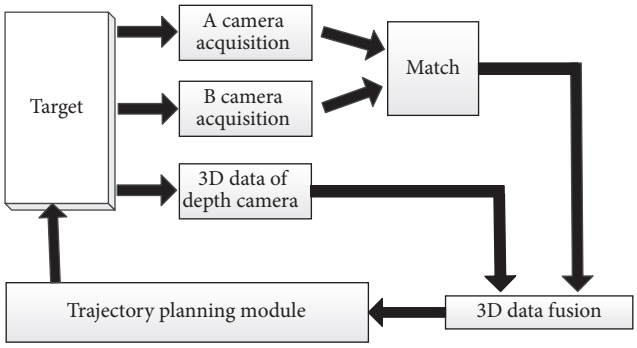


Figure 1: The proposed 3D reconstruction system.

extremely faster shutter. Hence TOF sensors can obtain near real-time scene depth. But, in the current generation, TOF sensors are limited in terms of resolution.

In general, binocular stereo vision system has the advantages of simple structure and high vertical resolution, but the stereo matching process is complex and has lots of matching errors in the weak texture. The vertical resolution of depth camera is independent of the distance of the scene in the imaging range, and it is relatively simple to achieve real-time three-dimensional image by using the data of depth camera, but the resolution and stability of TOF depth camera are lower by the limitation of hardware.

In this paper, a novel vision system of mobile robot is proposed, which can achieve more accurate 3D data by fusing stereo matching and direct depth measurement. In the proposed 3D reconstruction system, an image processing machine is used to simulate the human brain, and two parallel cameras and a depth camera are used to simulate human eyes as shown in Figure 1. Specifically, the image processing host utilizes binocular stereo cameras and achieves 3D reconstruction data based on stereo matching. Then image processing host will fuse the 3D reconstruction date of stereo matching with the data of depth camera and achieve 3D reconstructed data with high precision. Finally, it can figure out the physical distance of the scene objects from the camera by using this 3D data and achieve the strategy of trajectory planning [7, 8] to improve the stability of the robot. Moreover, double threads processing method is applied to improve the efficiency of the system. When the robot works, image processing machine obtains 3D reconstruction data in short time; the latest motion route is designed to guide the robot running. The experiments demonstrate that the 3D data fusion between the depth camera and stereo camera can meet the requirement of actual applications, but also it can improve the accuracy of matching. And, in practice, it is proved that the effect is very good and can improve the accuracy of 3D reconstruction.

The rest of this paper is organized as follows: Section 2 introduces the design of the system. In Section 3, we describe the framework of the 3D reconstruction system. In Section 4, experimental results are detailed and justify the superiority of our approach. Finally, conclusions are made in Section 5.

2. Design of the System

As shown in Figure 2, the whole system is composed of five parts: two color cameras of the same type, one depth camera, one image processing host, one mobile robot, and one mobile control host. Two color cameras are placed in the top of the mobile robot on the same horizontal line, and depth camera is placed below the left camera in the same vertical line. Three cameras communicate with image processing host through three USB3.0 line; the mobile robot is driven by two brushless DC motors and they are controlled by the mobile robot control host. The mobile robot control host and the image processing host communicate with each other through the serial port. In normal operation, the system is initialized. The three cameras start to take pictures of the same scene synchronously and transmit the image data to the image processing host through USB3.0 in real-time. Image processing host, first of all, utilizes the correlation algorithm [9, 10] to deal with two color images and calculate the disparity data of two images. Second, the image processing host matches the 3D image data of the low-resolution depth camera with the left image data of the color cameras. And, then, the image processing machine projects the spatial point detected by the TOF camera to the matching view. Finally, the 3D reconstruction data of the binocular camera is fused with the 3D data of the depth camera to obtain the 3D reconstruction data in high precision, by using which the trajectory planning is carried out. When the image processing host is in the process, image capturing thread updates and refreshes the scene data ceaselessly to ensure planning in short time. When image processing thread is completed, the trajectory planning data will be sent to the mobile control host through the serial port by the thread; mobile control host will control mobile robot to avoid obstacle.

As shown in Figure 2, the horizontal placement of two color cameras is particularly important for the accuracy of binocular matching. We can obtain relative rotation parameters and relative translation parameters by using a MATLAB toolbox for calibrating [11, 12]. The parameters will be used to correct the images and will make the images in the line alignment, and the overlapping area of the two images will be cut [13, 14]. The purpose of these preprocessing operations is to improve the accuracy of stereo matching and improve the matching time.

From Figure 2, we can also see the relative position of the depth camera and the left color camera. In order to make the scene reconstruction and fusion more accurate, it is necessary to register the data obtained by depth camera and color camera; namely, it is essential to match between the depth camera and the left color camera [15–17]. An effective camera registration method is developed between the depth camera and the left color camera in this paper. Firstly, we calibrate the two images (one is from left color camera; the other is from depth camera) to get the rotation parameters and relative translation parameters. Then, the image processing host uses the parameters to project the point of the depth camera to the image of left color camera. At last, a novel fusion method is proposed to get the high precision depth data.

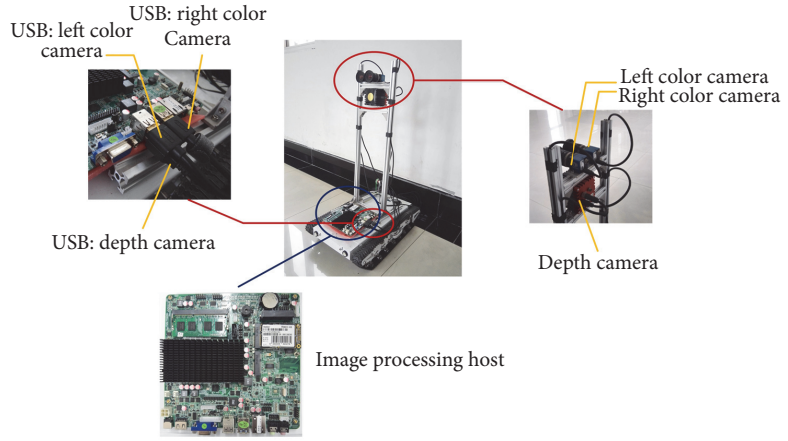


Figure 2: The components of the system.

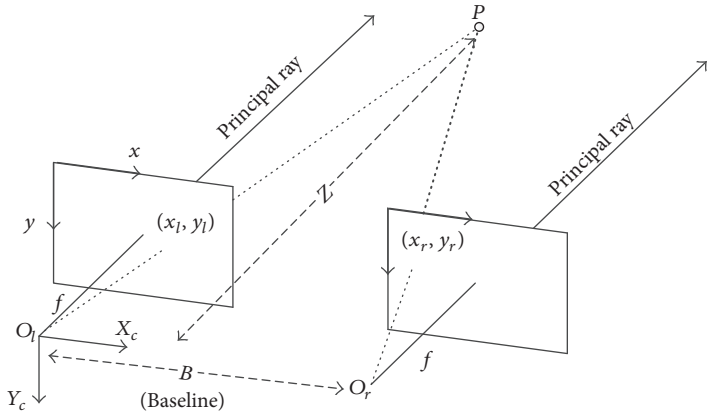


Figure 3: Analysis of binocular ranging.

3. 3D Fusion Reconstruction System

3.1. Binocular Ranging and Calibration. Let (u, v) be the pixel coordinate of color image; (X_c, Y_c, Z_c) is the corresponding coordinate in the color camera coordinate system. Based on the principle of small hole imaging,

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 \\ 0 & \frac{1}{dy} \end{bmatrix} \begin{bmatrix} f & 0 \\ 0 & f \end{bmatrix} \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}. \quad (1)$$

Here, f is the focal length of color camera; (c_x, c_y) are the coordinates of the principal point. d_x and d_y are physical sizes of pixel in the horizontal and vertical directions, respectively.

Define $f_x = f/d_x$, $f_y = f/d_y$; we can obtain

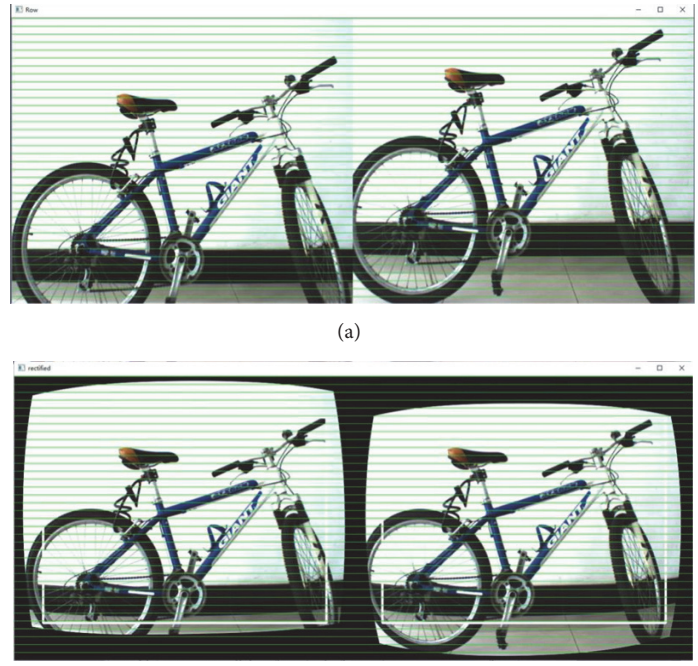
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \\ 1 \end{bmatrix}. \quad (2)$$

Here, f_x, f_y, c_x, c_y are the internal parameters of color camera. In this paper, the internal parameters of left and right cameras are obtained using the MATLAB Zhengyou Zhang calibration toolbox:

$$\begin{aligned} &\left[f_x, f_y, c_x, c_y\right]_{\text{left camera}} \\ &\quad = [728.33946, 802.46120, 366.05260, 264.87057]. \\ &\left[f_x^r, f_y^r, c_x^r, c_y^r\right]_{\text{right camera}} \\ &\quad = [725.16501, 798.07585, 343.09124, 219.13864]. \end{aligned} \quad (3)$$

As shown in Figure 3, the coordinates of the target point in the left view are (x_l, y_l), the coordinates of the target point in the left view are (x_r, y_r), binocular ranging mainly utilizes the differences of the target point horizontal coordinates on the direct images of two views (i.e., the disparity [18]: $d = x_l - x_r$), and the disparity has an inverse relationship with distance Z from the target point to the image plane:

$$\begin{aligned} &\frac{B}{Z} = \frac{B-d}{Z-f} \Longrightarrow \\ &Z = \frac{f \times B}{d}. \end{aligned} \quad (4)$$



(a)

(b)

Figure 4: Line alignment correction: (a) camera images before the correction; (b) camera images after the correction.

Hence, the key of binocular ranging is to get the disparity map between left and right images. Various binocular stereo matching methods have been proposed to calculate the disparity maps. Before binocular stereo matching, binocular correction should be applied to align the lines of left and right images strictly, making the lines exactly at the same level. Therefore, the points of the left image and the corresponding points of right image have the same number of rows. Because of this, one-dimensional search of rows can find the corresponding point for stereo matching, which will save the cost of computation. Specifically, left and right images can be rectified to achieve line alignment using the relative rotation matrix R and translation matrix T, which are obtained using the MATLAB Zhengyou Zhang calibration toolbox in this paper:

$$R = \begin{bmatrix} 0.999957 & 0.003964 & 0.008409 \\ -0.003916 & 0.999976 & -0.005686 \\ -0.008431 & 0.005653 & 0.999948 \end{bmatrix},$$
$$T = \begin{bmatrix} -47.36289 \\ -0.01925 \\ 0.47293 \end{bmatrix}. \quad (5)$$

As shown in Figure 4, it can be seen that the images after line alignment calibration can well achieve the line alignment. In this paper, then, the BM stereo matching algorithm is applied to obtain the disparity map because of the strict requirement about algorithm time.

3.2. Depth Registration. In the system, the model of the TOF depth camera is SR4000 camera, produced by the Swiss company Mesa Imaging. The TOF depth camera uses a modulator to emit infrared light from multiple angles. When the light returns to the sensor, it is possible to obtain the distance information for each pixel by the round trip time in real-time. The TOF depth camera can obtain depth information of the scene in real-time while the resolution of the camera is just 144 × 176. Hence, in this paper, a fusion system is developed to integrate depth camera and binocular stereo matching and improve the quality of 3D reconstruction.

Before fusion, the data on the depth map need to be aligned pixel by pixel to the color pixel. The registration process depends on the result of camera calibration [19, 20]. The parameters for registration are the conversion matrixes between the two coordinate systems [21, 22]. In this paper, an effective registration method is applied to align the depth image to the color image. Specifically, with the intrinsic parameters of the color camera we can mark the spatial position of each experimental checkerboard in the color camera coordinate system. Then, the depth camera can capture the position of the plate, which is coplanar with the checkerboard in physical space as shown in Figure 5. And, finally, we can obtain the projection matrix between the depth camera and the color camera by gradient descent optimization method.

Assume that the space point P is (X_c, Y_c, Z_c) under the color camera coordinate system and (X_d, Y_d, Z_d) under the depth camera coordinates.

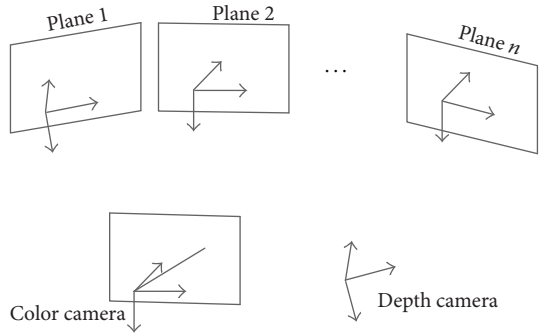


Figure 5: The comparison of the depth camera coordinate system and the color camera coordinate system.

Using formula (2), we can obtain

$$f_x \times \frac{X_c}{Z_c} + c_x = u,$$
$$f_y \times \frac{Y_c}{Z_c} + c_y = v. \tag{6}$$

Then we can obtain formulas (7) and (8):

$$\frac{X_c}{Z_c} = \frac{u - c_x}{f_x}, \tag{7}$$
$$\frac{Y_c}{Z_c} = \frac{v - c_y}{f_y}. \tag{8}$$

Conversion formula between color camera and depth camera coordinate system:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R_d \begin{bmatrix} X_d \\ Y_d \\ Z_d \end{bmatrix} + T_d. \tag{9}$$

Here, R_d and T_d are the relative rotation and translation matrixes between left color camera and the depth camera.

Expanding formula (9), we can get

$$X_c = r_{11}X_d + r_{12}Y_d + r_{13}Z_d + t_1,$$
$$Y_c = r_{21}X_d + r_{22}Y_d + r_{23}Z_d + t_2, \tag{10}$$
$$Z_c = r_{31}X_d + r_{32}Y_d + r_{33}Z_d + t_3.$$

Combining formula (7) and (8), we can obtain

$$\frac{X_c}{Z_c} = \frac{r_{11}X_d + r_{12}Y_d + r_{13}Z_d + t_1}{r_{31}X_d + r_{32}Y_d + r_{33}Z_d + t_3} = \frac{u - c_x}{f_x},$$
$$\frac{Y_c}{Z_c} = \frac{r_{21}X_d + r_{22}Y_d + r_{23}Z_d + t_2}{r_{31}X_d + r_{32}Y_d + r_{33}Z_d + t_3} = \frac{v - c_y}{f_y}. \tag{11}$$

The extrinsic parameters R_d and T_d can be calculated from formulas (11) using gradient descent optimization method. The accuracy of R_d and T_d plays an important role in the accuracy of the system, since the depth information of the TOF camera is determined by the phase difference between the emitting infrared light and reflected infrared light. If the reflected light intensity is higher, the credibility of the depth value is higher. By contraries, if the light is largely absorbed, the obtained depth value may be far from the true value. So the depth value of black block is less precise while the depth value of white block is more realistic. Therefore, we can only extract the depth data of the white checkerboard for subsequent processing. And we can extract depth value of the white block pixel by setting the gray level threshold.

Moreover, due to the noise of TOF camera points on the same plane will have a slight deviation. In order to obtain the one closest to the real value of the depth, we fit the three-dimensional coordinate data of the same template by using equation $aX + bY + CZ + d = 0$. The optimal equation coefficients a, b, c, d are obtained by least squares method [23, 24]. In order to get more accurate extrinsic parameters R_d and T_d, we take many groups of pictures to do the same operation above. Figure 6(a) reports some images of the chess board map of left color camera; Figure 6(b) shows gray images of the chess board map of depth camera; Figure 6(c) shows depth images of the chess board map of depth camera. After the calculation process above, the parameters R_d and T_d are obtained as follows:

$$R_d = \begin{bmatrix} -1.0353572 & 0.0191776 & 0.0209886 \\ 0.00625004 & -0.8740292 & 0.0792625 \\ -0.0206831 & -0.0268092 & 0.7136297 \end{bmatrix},$$
$$T_d = \begin{bmatrix} -0.033684 \\ -0.021450 \\ 0.450148 \end{bmatrix}. \tag{12}$$

Using R_d and T_d, the depth image can be registered to the color image effectively as shown in Figure 7.

3.3. Depth Fusion Algorithm. As shown in Figure 7(c), lots of depth regions need to be filled after the registration. Hence, an effective depth fusion algorithm is proposed in this paper as shown in Figure 8. Firstly, the bilinear interpolation is applied to the registered depth map. After interpolation, there still exist black holes to be filled as shown in the Figure 9(d). Then, a fusion method is proposed based on the result of binocular stereo matching to improve the accuracy. Specifically, we traverse the pixel of depth image after interpolation, when the pixel value is zero, and then look for the corresponding value of the binocular matching result as shown in Figure 9(c). If the value of binocular matching is valid, the result of binocular matching will be filled in the depth image (before filling, the disparity result of binocular stereo matching should be transformed into the actual depth value). Then, an initial fusion depth image is obtained as shown in Figure 9(e).

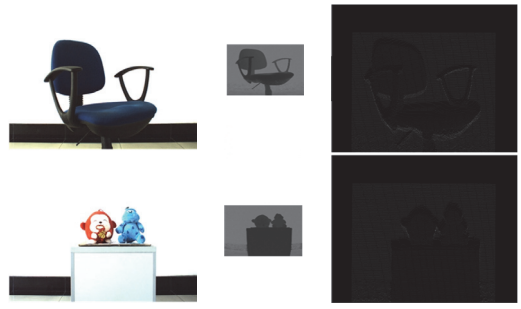
Moreover, a filling method based on mean-shift segmentation is applied further based on the following prior assumption:

(1) The pixels with similar colors around a region are likely to have similar depth.

(2) World surfaces are piecewise smooth.

(a) (b) (c)

Figure 6: The images used to obtain the parameters R_d and T_d between left color camera and the depth camera. (a) Gray images of the chess board map of left color camera; (b) images of the chess board map of depth camera; (c) depth images of the chess board map of depth camera.



(a) (b) (c)

Figure 7: Depth registration: (a) the left color image; (b) the low-resolution depth image; (c) depth image registered with color image.

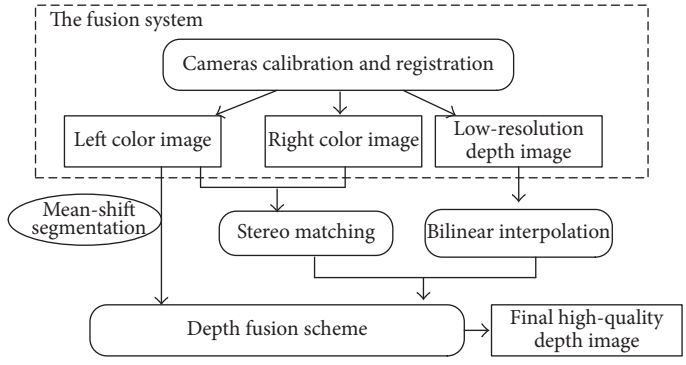


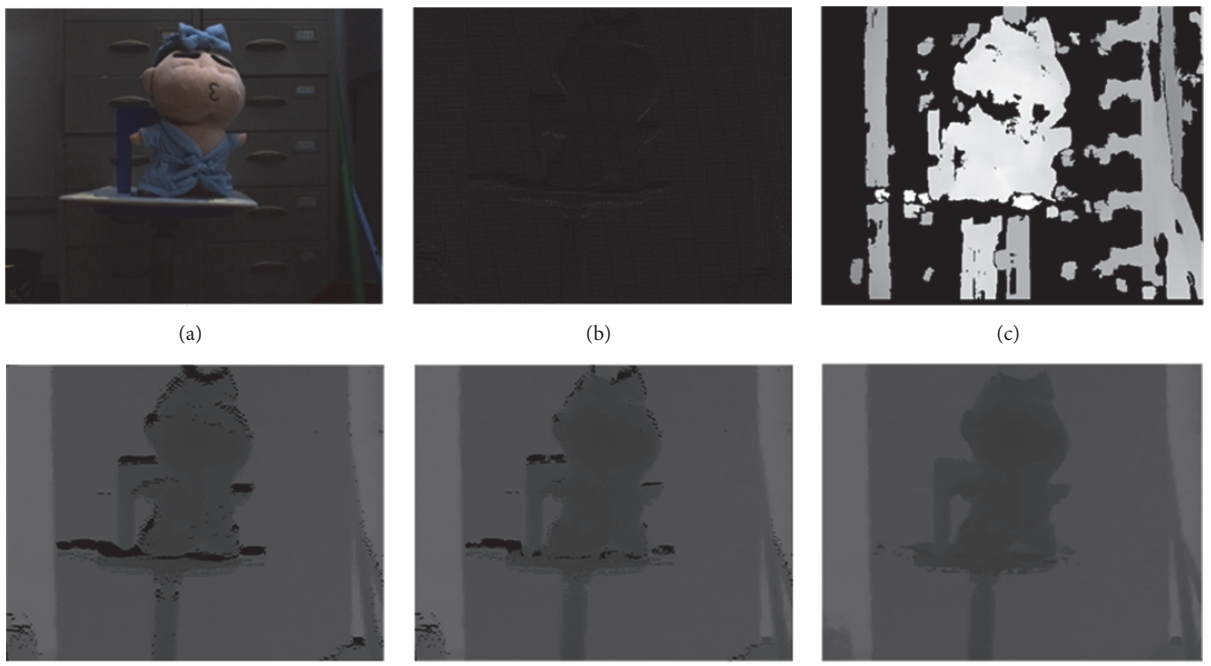
Figure 8: The framework of our fusion system.

Specifically, the mean-shift segmentation algorithm is used in the left color image. Then the pixels in the remaining black holes are assigned using the maximum probability value of the nonzero pixels in the same color segmentation region. Finally, the final depth image can be obtained after median filtering as shown in Figure 9(f). It can be seen that the quality of the final depth image is improved obviously.

4. Experimental Results

The system utilizes a mobile robot, two of the same model Mercury (MERCURY) Series-High Speed 30 surveillance cameras made by Daheng, China (http://www.daheng-image.com/), and a Swiss Mesa Imaging SR4000 depth camera as based hardware. Image processing system runs in VS2012 based on OpenCV2.4.9 and the industrial control panel is used as the image processing host. In the basis above, we tested the system proposed in this paper. After the system is initialized, the program starts running. First, the processing host reads the parameters into the memory, and then the system starts to capture the images and display images (only display in test mode). The host will match the images by using the parameters in memory and obtain a new three-dimensional reconstruction data. Finally, the system can complete the trajectory planning via these data to avoid obstacles.

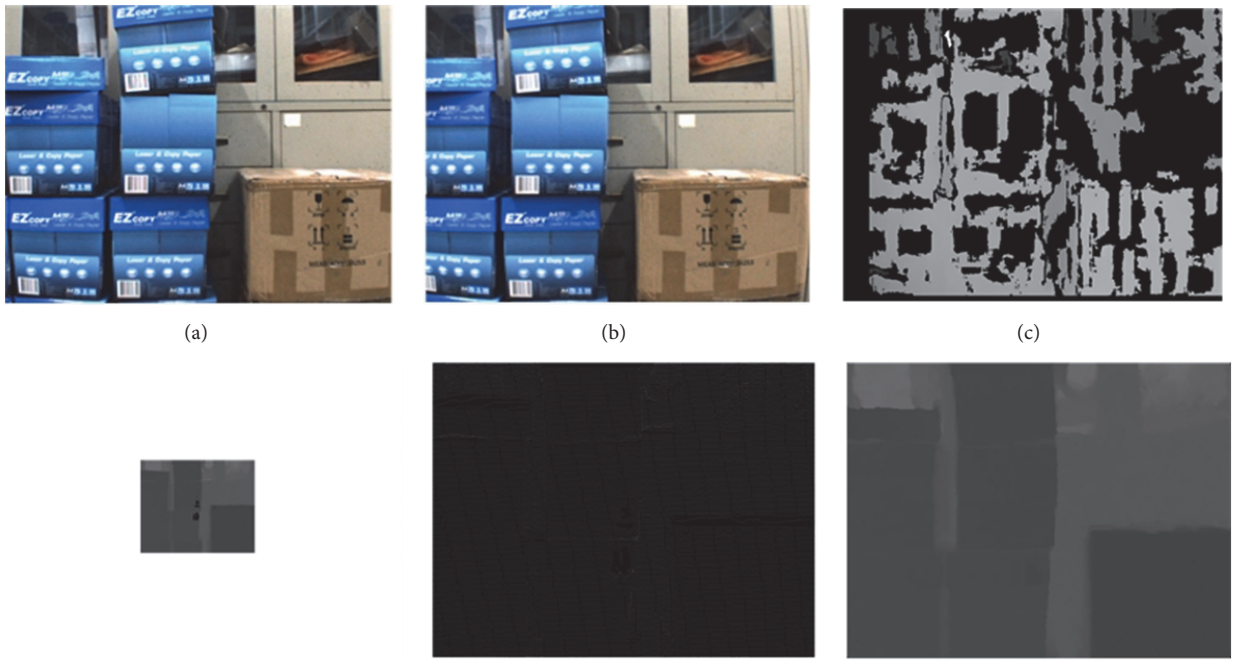
Firstly, experiments are carried out to test the system. As shown in Figures 10–12, the system can obtain satisfying depth images after fusion.



(a) (b) (c)

(d) (e) (f)

Figure 9: The effect of the proposed depth fusion algorithm: (a) the left color image; (b) the registered depth image; (c) disparity image of binocular stereo matching (BM algorithm); (d) depth image of interpolation; (e) the initial fusion depth image; (f) the final depth image.



(a) (b) (c)

(d) (e) (f)

Figure 10: Result of the proposed system: (a) left color image; (b) right color image; (c) disparity map; (d) low-resolution depth image from depth camera; (e) depth image registered with the left color image; (f) the final fusion depth result.

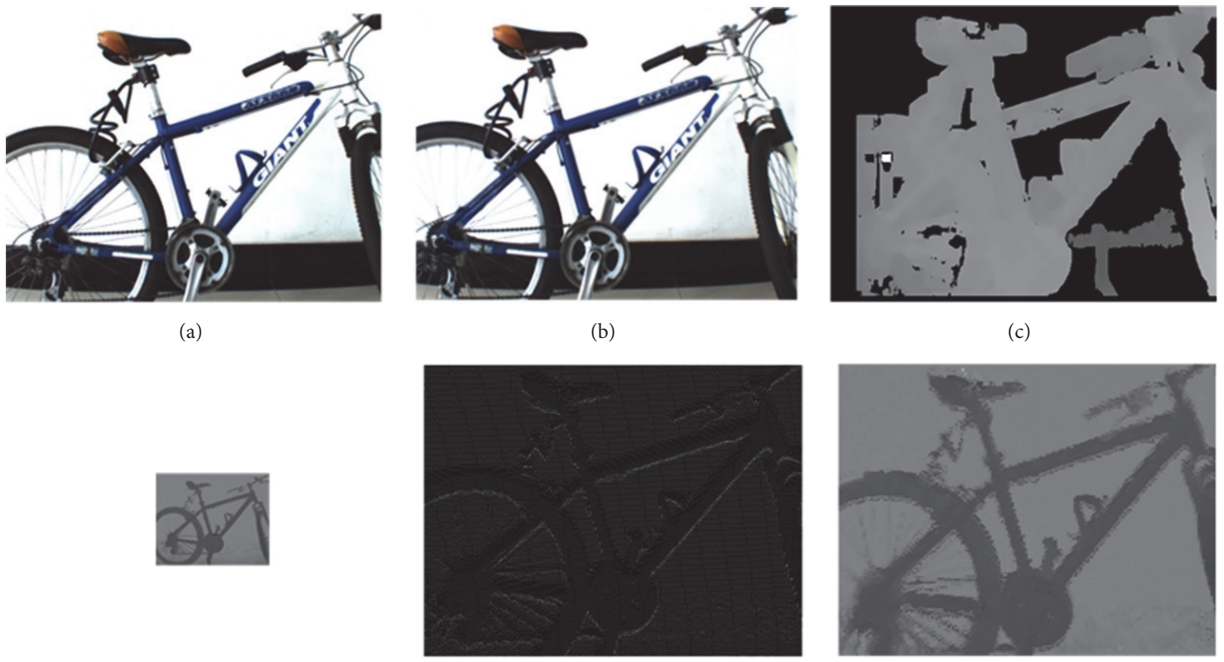


Figure 11: Result of the proposed system: (a) left color image; (b) right color image; (c) disparity map; (d) low-resolution depth image from depth camera; (e) depth image registered with the left color image; (f) the final fusion depth result.

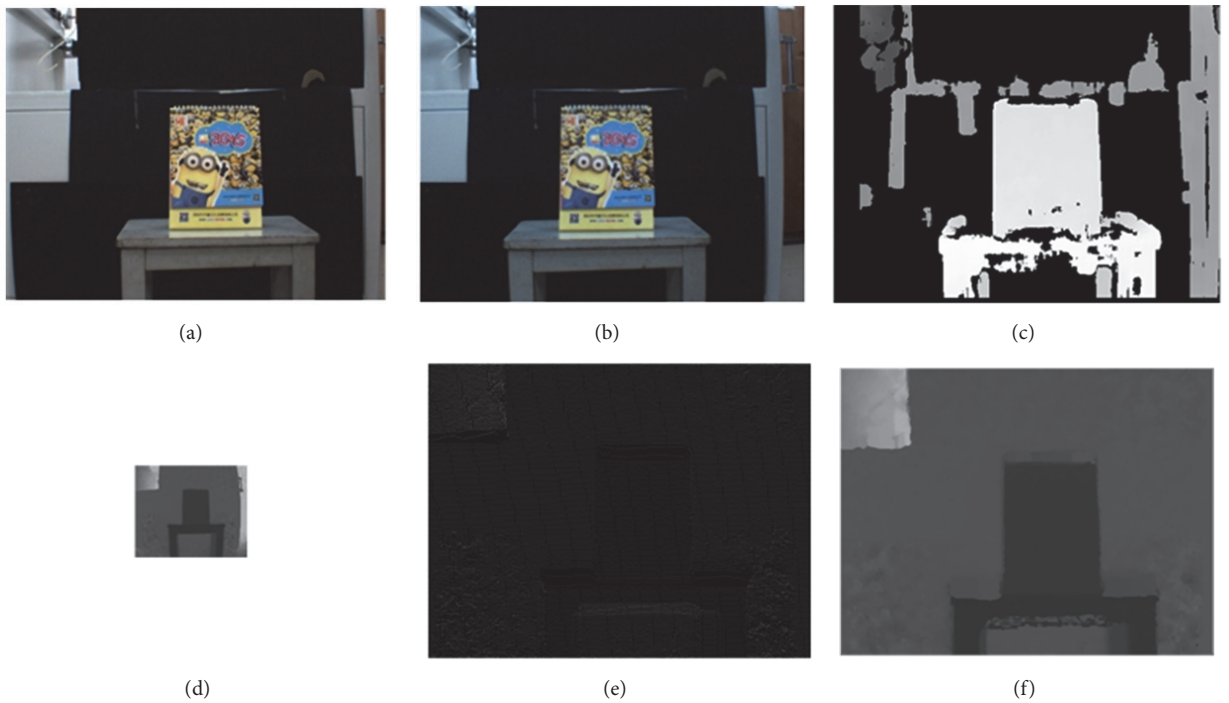


Figure 12: Result of the proposed system: (a) left color image; (b) right color image; (c) disparity map; (d) low-resolution depth image from depth camera; (e) depth image registered with the left color image; (f) the final fusion depth result.

Table 1: Accuracy experiments for the system.

Regions	Object coordinates (cm) (x, y, z) The coordinate of our system	Actual coordinate	Error
I	(15.82, 67.48, 237.46)	(15.71, 67.54, 237.40)	(0.11, −0.06, 0.06)
	(13.05, 73.21, 257.63)	(13.01, 73.09, 257.65)	(0.04, 0.12, −0.02)
	(15.87, 66.72, 238.17)	(15.75, 66.81, 238.09)	(0.12, −0.09, 0.08)
II	(−27.11, 20.18, 238.90)	(−27.23, 20.10, 239.10)	(0.12, 0.08, −0.20)
	(−57.24, 20.33, 204.07)	(−57.36, 20.40, 204.28)	(0.12, −0.07, −0.21)
	(−76.99, 77.70, 273.17)	(−77.15, 77.65, 273.05)	(0.16, 0.05, 0.12)
III	(−4.43, −58.88, 206.88)	(−4.70, −58.90, 206.90)	(0.27, 0.02, −0.02)
	(−10.33, −55.01, 193.18)	(−10.10, −55.00, 193.34)	(−0.23, −0.01, −0.16)
	(−66.33, −58.17, 216.09)	(−66.21, −58.36, 216.10)	(−0.12, 0.19, −0.01)
IV	(36.26, −18.20, 207.18)	(36.30, −18.20, 207.25)	(−0.04, 0, −0.07)
	(59.41, −18.18, 207.01)	(59.40, −18.30, 207.00)	(0.01, 0.12, 0.01)
	(73.41, −4.99, 209.19)	(73.35, −5.00, 209.05)	(0.06, 0.01, 0.14)

Table 2: Processing time.

Serial number	Time/s
1	0.521100
2	0.529700
3	0.596000
4	0.579333
5	0.549250
6	0.553500
7	0.547000
8	0.567750
9	0.514250
10	0.555250

Then, the accuracy of the system is compared in Table 1. "The coordinate of our system" is the coordinate value of an object obtained using our system. "Actual coordinate" is obtained from actual measurements (the depth camera coordinate is defined as the real-world coordinate). As shown in Table 1, the MSE of the measurement experiments is 0.019, 0.0076, and 0.0133 for X, Y, Z directions, respectively. The accuracy of our system can well meet the requirements of practical applications.

Then, we analyze the time performance of the system. As shown in Table 2, the average processing time is 0.5513133 seconds, which can well meet the requirements of actual applications, such as mobile robot avoidance.

In the system, software (i.e., image processing system) takes VS2012 as the development platform, which is based on OpenCV2.4.9. In order to observe the change of scene and view the effect of fusion, we code the program based on MFC. The program uses multithreaded control mode. The camera part is controlled by a separate thread to facilitate real-time acquisition, which easily refreshes the buffer of the image in real-time. The fusion part is controlled by another separate processing thread. Acquisition thread and processing thread will not affect each other, which can guarantee the high efficiency of processing and avoid delay. The MFC interface

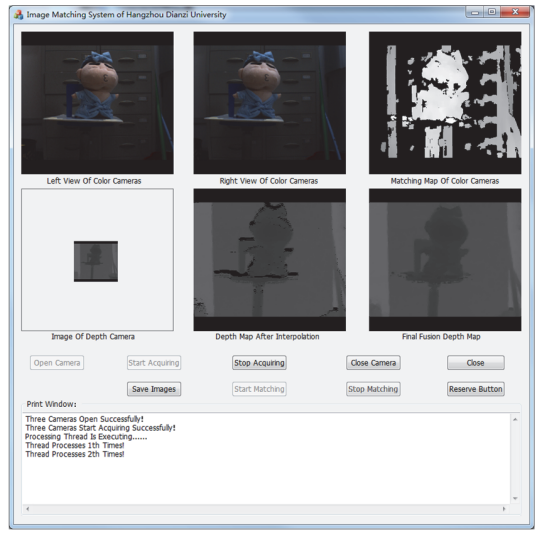


Figure 13: The running result of the system.

and running results are shown in Figure 13 (use button "Start Matching" for the choice of whether to start the integration; you can use the button "Stop Matching" to end the integration). The current system uses a multithreaded way to satisfy the strict requirement of time cost; later, we will try to use GPU acceleration method to make time cost lower.

5. Summary

In this paper, the visual processing section is used to acquire the three-dimensional scene data of the front of the mobile robot. The three cameras will transmit the image data and the 3D data of the depth camera to the image processing host via USB3.0. Image processing host uses calibration data to calibrate the images and maps the depth data to the left color image after dimensional registration. At the same time, the host will match the two color images by using correlation algorithm. And then the system will fuse the calibrated 3D

data of the depth camera with the matching result of the two color images to achieve the three-dimensional scene data with higher accuracy. Finally, the high precision 3D data is used to guide the mobile robot to avoid obstacles.

The system is proposed based on the following two observations: (1) Real-time and accuracy of binocular matching are a contradiction, and it is difficult to ensure that both real-time and high accuracy meet the requirements. It can be used in the scenes where the accuracy requirements are not very strict. But, it is hard to meet the requirements of high precision mobile robots. (2) The three-dimensional camera data of the TOF depth camera is accurate, and the real-time of the camera is very high too, but due to the structural constraints, the resolution is very low, only $144 * 176$, which makes it hard to meet the requirements of mobile robots either. However, the difficulty of this system is to accurately register the depth data to the left color picture. In order to reduce the impact of various objective factors such as noise, this paper calculates the R_d and T_d parameters matrix by taking multiple sets of images. The test results show that the system can meet the requirements of mobile robot trajectory planning. The system uses double threads processing method to improve efficiency of the system. And, later, we will try to use GPU acceleration method to ensure real-time.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Natural Science Foundation of China (U1609216 and 61401129) and Zhejiang Natural Science Foundation of China (LY17F010020).

References

[1] C. Cadena, L. Carlone, H. Carrillo et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] N. Q. Ann, M. H. Achmad, L. Bayuaji, M. R. Daud, and D. Pebrianti, "Study on 3D scene reconstruction in robot navigation using stereo vision," in *Proceedings of the 2016 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, pp. 72–77, Selangor, Malaysia, October 2016.

[3] S. A. Waskitho, A. Alfarouq, S. Sukaridhoto, and D. Pramadihanto, "FloW vision: Depth image enhancement by combining stereo RGB-depth sensor," in *Proceedings of the 5th International Conference on Knowledge Creation and Intelligent Computing, KCIC 2016*, pp. 182–187, November 2016.

[4] A. A. Ahmed, M. K. Elbashir, and A. A. Osman, "Distance alert system using Stereo vision and feature extraction," in *Proceedings of the 2017 International Conference on Communication, Control, Computing and Electronics Engineering, ICCCCEE 2017*, January 2017.

[5] M. K. Singh, K. S. Venkatesh, and A. Dutta, "Accurate rough terrain modeling from fused 3D point cloud data," in *Proceedings of the 12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control, INDICON 2015*, December 2015.

[6] O. S. Gedik and A. A. Alatan, "3D tracking using visual and range data," in *Proceedings of the 2012 20th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, Mugla, Turkey, April 2012.

[7] W. Liu, Z. Li, L. Li, and F. Wang, "Parking Like a Human: A Direct Trajectory Planning Solution," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10.

[8] P. Pharpatara, B. Herisse, and Y. Bestaoui, "3-D trajectory planning of aerial vehicles using RRT∗," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1116–1123, 2017.

[9] J. Cheng, C. Leng, J. Wu, H. Cui, and H. Lu, "Fast and accurate image matching with cascade hashing for 3D reconstruction," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 1–8, June 2014.

[10] K. Lu, X. Wang, Z. Wang, and X. Li, "Image-based 3d models reconstruction," in *Proceedings of the IET International Communication Conference on Wireless Mobile and Computing (CCWMC 2011)*, pp. 482–485, Shanghai, China, 2011.

[11] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[12] A. Fetić, D. Jurić, and D. Osmanković, "The procedure of a camera calibration using Camera Calibration Toolbox for MATLAB," in *Proceedings of the 35th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2012*, pp. 1752–1757, hrv, May 2012.

[13] X. Cao and H. Foroosh, "Camera calibration using symmetric objects," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3614–3619, 2006.

[14] M. Shimizu and M. Okutomi, "Calibration and rectification for reflection stereo," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, June 2008.

[15] V. Gandhi, J. Čech, and R. Horaud, "High-resolution depth maps based on TOF-stereo fusion," pp. 4742–4749.

[16] Y.-S. Kang and Y.-S. Ho, "Disparity map generation for color image using TOF depth camera," in *Proceedings of the 5th 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 3DTV-CON 2011*, May 2011.

[17] Y.-S. Kang and Y.-S. Ho, "High-quality multi-view depth generation using multiple color and depth cameras," in *Proceedings of the 2010 IEEE International Conference on Multimedia and Expo, ICME 2010*, pp. 1405–1408, July 2010.

[18] P. An, Z. Zhang, and L. Shi, "Theory and experiment analysis of disparity for stereoscopic image pairs," in *Proceedings of the 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, ISIMP 2001*, pp. 68–71, May 2001.

[19] J. Jung, J.-Y. Lee, Y. Jeong, and I. S. Kweon, "Time-of-Flight Sensor Calibration for a Color and Depth Camera Pair," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1501–1513, 2015.

[20] F. Basso, A. Pretto, and E. Menegatti, "Unsupervised intrinsic and extrinsic calibration of a camera-depth sensor couple," in *Proceedings of the 2014 IEEE International Conference on Robotics and Automation, ICRA 2014*, pp. 6244–6249, June 2014.

[21] H. Yu, K. Zhao, Y. Wang, L. Kan, M. Sun, and W. Jia, "Registration and fusion for ToF camera and 2D camera reading," in *Proceedings of the 2013 Chinese Automation Congress, CAC 2013*, pp. 679–684, November 2013.

[22] J. Kim, S. Park, S. Kim, and S. Lee, "Registration method between ToF and color cameras for face recognition," in *Proceedings of the 2011 6th IEEE Conference on Industrial Electronics and Applications, ICIEA 2011*, pp. 1977–1980, June 2011.

[23] Y. Zheng and S. Peng, "A practical roadside camera calibration method based on least squares optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 831–843, 2014.

[24] D. Buchczik and W. Ilewicz, "Evaluation of calibration results using the least median of squares method in the case of linear multivariate models," in *Proceedings of the 21st International Conference on Methods and Models in Automation and Robotics, MMAR 2016*, pp. 800–805, September 2016.

Hindawi
Journal of Sensors
Volume 2017, Article ID 8765450, 12 pages
https://doi.org/10.1155/2017/8765450

Research Article

Global Measurement Method for Large-Scale Components Based on a Multiple Field of View Combination

Yang Zhang, Wei Liu, Zhiguang Lan, Zhiyuan Zhang, Fan Ye, Haiyang Zhao, Xiaodong Li, and Zhenyuan Jia

Key Laboratory for Precision and Non-Traditional Machining Technology of the Ministry of Education, Dalian University of Technology, Dalian 116024, China

Correspondence should be addressed to Wei Liu; lw2007@dlut.edu.cn

Received 1 March 2017; Revised 7 July 2017; Accepted 24 July 2017; Published 17 September 2017

Academic Editor: Vera Tyrsa

Considering the limited measurement range of a machine vision method for the three-dimensional (3D) surface measurement of large-scale components, a noncontact and flexible global measurement method combining a multiple field of view (FOV) is proposed in this paper. The measurement system consists of two theodolites and a binocular vision system with a transfer mark. The process of multiple FOV combinations is described, and a new global calibration method is proposed to solve the coordinate system unification issue of different instruments in the measurement system. In addition, a high-precision image acquisition method, which is based on laser stripe scanning and centre line extraction, is discussed to guarantee the measurement efficiency. With the measured 3D data, surface reconstruction of large-scale components is accomplished by data integration. Experiments are also conducted to verify the precision and effectiveness of the global measurement method.

1. Introduction

In the field of modern manufacturing and assembly, the accuracy of large-scale components of dimensions is required to guarantee automatic assembly of high-end equipment. To ensure that the components are manufactured as designed, the dimensions of the components have to be accurately measured [1, 2]. However, the measurement range for large components is too large to rapidly complete onsite measurements using a single instrument. For large high-precision parts that are used in aerospace and aviation, mark points are not allowed to be pasted on the surfaces of the components; that is, the measuring methods that only work with the marks pasted on the surfaces are not suitable. Large-scale components, such as wings of airplanes and large antenna radomes, are usually clamped onto a bracket to prevent deformation prior to assembly. As a result, the surfaces of these components are partly blocked from view, and the area of interest for the measurement instrument is invisible from a specific viewing angle. Thus, studies of noncontact and flexible global measurement methods for 3D surface measurement of large-scale components are important for high-quality assembly in aerospace and aviation.

Machine vision measurement has such advantages as noncontact, high efficiency, and high accuracy, so it has been extensively applied to industry [3–5]. Recently, researchers have proposed large-scale component measurement methods that are based on machine vision [6–8]. There are three main machine vision methods using in the measurement of large aviation components; they are photogrammetry [9–11], structured light method, and binocular vision [12–15]. Photogrammetry is a method to collect quantity of reflective marks pasted on the surface of a part using a single camera and reconstruct the marks to represent the surface of the part. A system called V-STARS adopted this method to measure the lay-up tool of Boeing B320 aircraft [16]. The system consists of a high resolution camera and reflective marks which need to be affixed onto the surface of the measured part. The marking points can be reconstructed by the images captured by the camera. The system has the advantages of good portability. However, pasting the marking points is time-consuming, and it is difficult to measure the entire surface using marking points, especially the boundary of the surface;

structured light method is to reconstruct the projecting light by building the triangulation between the monocular camera and the active projecting light (laser/grating). During the measurement, the camera and the position of the projecting light have to remain relatively still, which brings a problem that, to guarantee the measurement accuracy, the FOVs for measurement are a few. To measure parts using large FOV, measurement equipment needs to be driven by highly accurate movement mechanisms to scan parts, as a novel 3D scanning measurement technique for large components proposed by Jiang et al. [17]. However, in field of aviation assembly, limited to complex environments, the method that uses large and highly accurate movement mechanisms to achieve on site measurement for large components is impractical. A German company Gom measures large components with structured lights and photogrammetry and has developed a measurement system called ATOS based on profilometry with grating and another system called TRITOP based on photogrammetry with reflective marks, respectively. NASA combined the ATOS system with the TRITOP system to measure X-38 aircraft [18], whereas too many moves of the method are needed, and quantity of mark points has to be stuck on the surfaces of parts ahead of measurement. Binocular vision is a method that uses binocular cameras to collect features projected onto surface to be measured and reconstruct the surface; a light probe based large FOV 3D vision measurement system proposed by Feng and Wei, which is based on binocular vision, can measure the curved surface of a large-scale object [19]. However, keeping the aided target steady during the measuring process is difficult, and only the coordinates of the points on the curved surface can be measured using this method. The measuring process is complex, and the measurement accuracy is unstable. Therefore, a novel global measurement method for 3D surfaces of large-scale components, which is based on a multiple field of view (FOV) combination, is proposed in this paper.

First, the principle and measurement process of the proposed method are introduced. Second, the calibration method of the measurement system is presented. Third, the algorithms of laser stripe extraction, image matching, and reconstruction are described. Last, the flexibility and precision of the measurement method are verified by experiments, as discussed in Section 5.

2. Measurement Principle

The measurement system is a binocular vision system that consists of two theodolites and a transfer mark that is designed for the transition of views. In the measuring process, two theodolites that are placed in the back are employed as the global control station for field combination and data integration. The binocular vision system with the transfer mark is placed in the front to capture an image of the component surface. The global measurement coordinate system is established on the left theodolite to ensure that the binocular vision system can be placed in any position in the measurement range of the theodolites. By transferring the surface information that is captured by cameras in different positions into the global coordinate system, high-precision data integration can be achieved. The measurement principle of the system is shown in Figure 1.

As the measurement system consists of different types of instruments, the measurement coordinate systems, respectively, established on these instruments are different as well. To reconstruct an entire surface, the coordinates that are measured by these instruments must be unified in the global coordinate system. The *Craig* expression method of coordinate transformation is adopted in this paper to express the transformation relation [20]. The coordinate vector of any point P in the coordinate system F is expressed as ${}^{F}\mathbf{P}$

$$ {}^{F}\mathbf{P} = {}^{F}\overrightarrow{\mathrm{OP}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Longleftrightarrow \overrightarrow{\mathrm{OP}} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{1} $$

where (x, y, z) is the coordinate of point P. Two theodolites are employed as the global measurement control station of the measurement system. The global coordinate system $o_{\mathrm{G}}x_{\mathrm{G}}y_{\mathrm{G}}z_{\mathrm{G}}$ is established on the left theodolite.

The binocular vision system consists of two industry complementary metal oxide semiconductor (CMOS) cameras. The coordinate system of the left camera—$o_{\mathrm{CL}}x_{\mathrm{CL}}y_{\mathrm{CL}}z_{\mathrm{CL}}$—is established on the left camera, with the origin on the optical centre of the camera, and the coordinate axis is in the same directions as the axis of the image sensor. Similarly, the right camera coordinate system—$o_{\mathrm{CR}}x_{\mathrm{CR}}y_{\mathrm{CR}}z_{\mathrm{CR}}$—is established on the right camera.

In the process of measurement, the large component to be measured must be divided into multiple fields of view and measured several times; thus, the cameras must be moved to different positions. To connect the mobile camera coordinate systems with the fixed global coordinate system, a transfer mark is taken into use of the binocular vision system. The relative position between the mark and the cameras remains unchanged during the measurement. By measuring the feature points on the mark, two theodolites in the back can obtain the position and the direction of the binocular vision system. The coordinate system of the transfer marks $o_{\mathrm{M}}x_{\mathrm{M}}y_{\mathrm{M}}z_{\mathrm{M}}$ and ${}^{\mathrm{M}}_{\mathrm{G}}\mathbf{H}$ is the transformation matrix from the transfer mark coordinate system to the global coordinate system.

The measurement process of the proposed system is as follows: first, the system is calibrated, which includes the calibration of two theodolites, the binocular cameras, and their transformation relation which is calibrated using the transfer mark. Second, the surface of the large-scale component is artificially divided into several parts and separately measured by the binocular cameras at different positions; based on binocular vision method with assisted laser, the 3D point cloud data of measured part is obtained [21]. Third, feature points on the transfer mark are measured by two theodolites for coordinate system transformation. Last, the data of every measured part obtained by the cameras at different positions are integrated into the global coordinate system. With fusion of the obtained data, the component with overall size is measured. With the data obtained, the entire surface of the large component can be reconstructed.

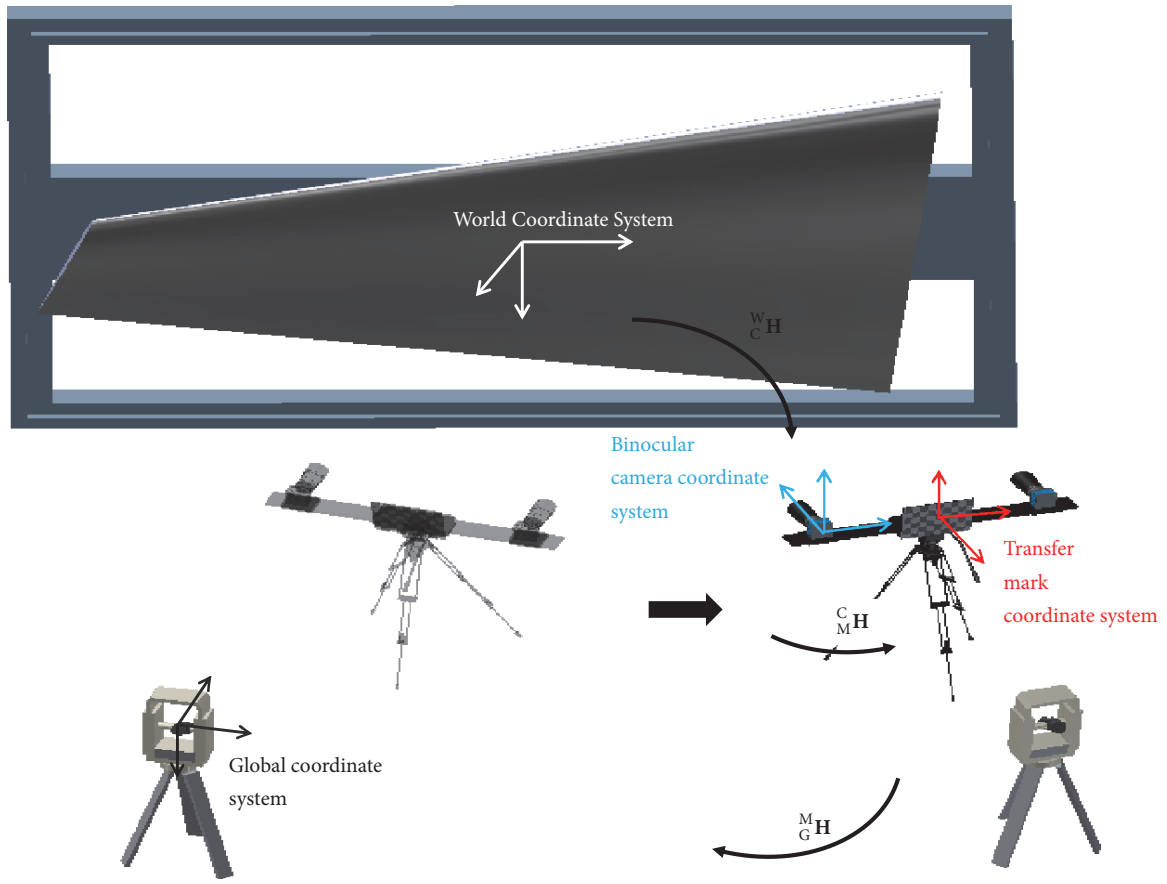


Figure 1: Schematic of the measurement system.

3. Calibration of the Measurement System

The calibration of the measurement system is the foundation of the 3D measurements. The proposed measurement system consists of different instruments. Thus, the high-precision calibration of all instruments is essential for ensuring the accuracy of large field measurements.

To measure the large surface of a component, the binocular vision system must be moved several times to acquire regional characteristic information. Due to the changed measuring position, the spatial location of the camera coordinate system varies in the meantime. Thus, an accurate transformation relationship between the camera coordinate system and the global coordinate system must be established.

In this paper, the global coordinate system is established on the left theodolite. The two-theodolite system is calibrated according to the two-theodolite spatial three-dimensional (3D) measurement model. The binocular cameras are calibrated using Zhang's calibration method [22]. By measuring the coordinates of the feature points on the transfer mark with the theodolites, the relation among the theodolites, the binocular cameras, and the transfer mark can be obtained. In this manner, global calibration is accomplished.

3.1. Calibration of Two Theodolites. The global coordinate system is established on the left theodolite according to the perspective projection model [23, 24]. As shown in Figure 2, the 3D coordinate system $o_{TL}x_{TL}y_{TL}z_{TL}$ of the left theodolite is constructed with the origin on the observation centre o_{TL}. The two-dimensional (2D) image coordinate system $O_{TL}X_{TL}Y_{TL}$ of the left theodolite is established at $z_{TL} = 1$ in $o_{TL}x_{TL}y_{TL}z_{TL}$. The relation between $O_{TL}X_{TL}Y_{TL}$ and $o_{TL}x_{TL}y_{TL}z_{TL}$ can be calculated based on the perspective projection model as

$$\begin{bmatrix} X_{TL} \\ Y_{TL} \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} x_{TL} \\ y_{TL} \\ z_{TL} \end{bmatrix} = \begin{bmatrix} \cot \alpha_{TL} \\ \dfrac{\tan \varphi_{TL}}{\sin \alpha_{TL}} \\ 1 \end{bmatrix}, \tag{2}$$

where λ is the scale factor, α_{TL} is the horizontal angle, and φ_{TL} is the vertical angle obtained by the theodolite. For ensuring the accuracy of measurement, the degree of the horizontal angle and the vertical angle should not be about 0 degrees or 90 degrees. (X_{TL}, Y_{TL}) and (x_{TL}, y_{TL}, z_{TL}) are the coordinates in the 2D image coordinate system of the left theodolite and the coordinates in the 3D coordinate system of the left theodolite, respectively.

Similarly, the 3D coordinate system $o_{TR}x_{TR}y_{TR}z_{TR}$ and the 2D image coordinate system $O_{TR}X_{TR}Y_{TR}$ of the right theodolite are established, and the relationship between these two systems is calculated.

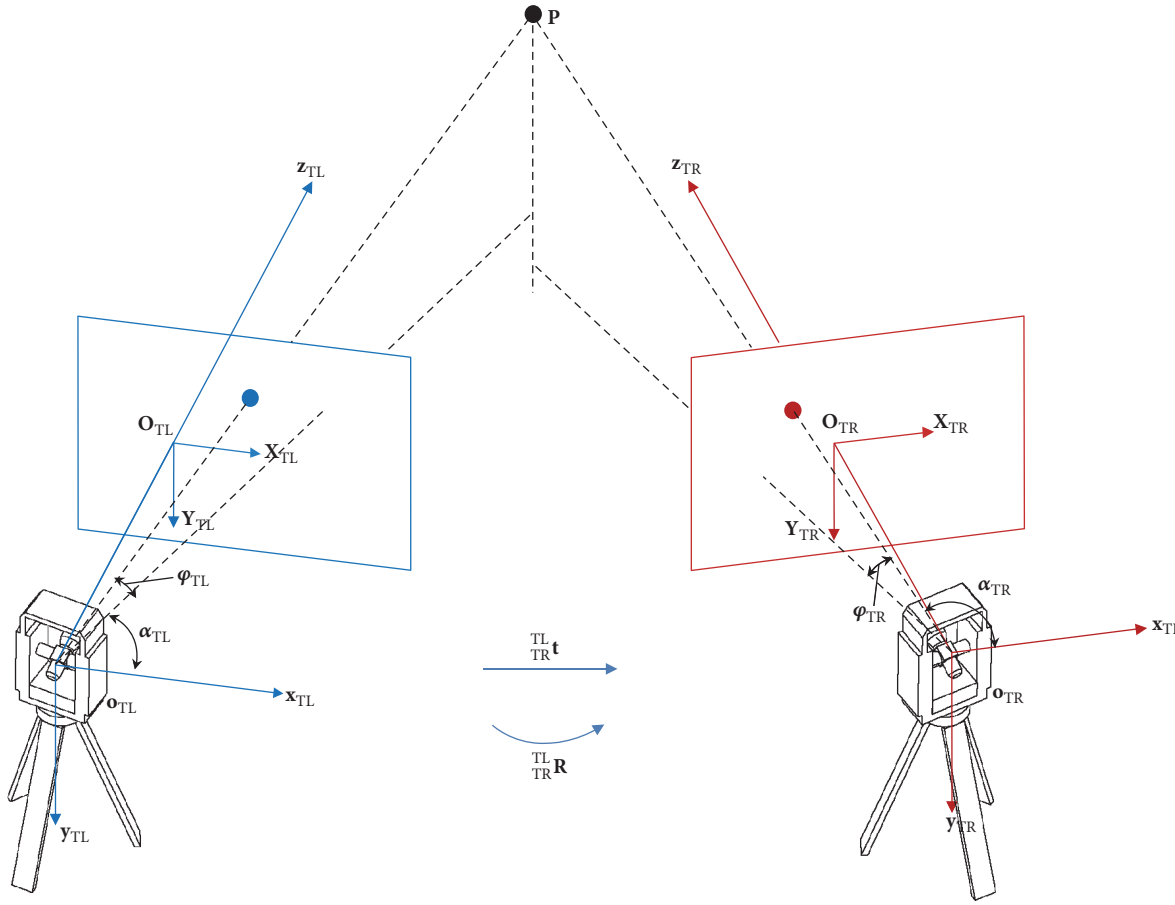


Figure 2: Coordinate systems for the two theodolites.

The coordinate vector of point P in $o_{TL}x_{TL}y_{TL}z_{TL}$ is expressed as ${}^{TL}\mathbf{P}$, and ${}^{TR}\mathbf{P}$ is the coordinate vector of point P in $o_{TR}x_{TR}y_{TR}z_{TR}$. The relationship between ${}^{TL}\mathbf{P}$ and ${}^{TR}\mathbf{P}$ is expressed as

$$\begin{bmatrix} {}^{TR}\mathbf{P} \\ 1 \end{bmatrix} = {}^{TL}_{TR}\mathbf{H} \begin{bmatrix} {}^{TL}\mathbf{P} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{TL}_{TR}\mathbf{R} & {}^{TL}_{TR}\mathbf{t} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} {}^{TL}\mathbf{P} \\ 1 \end{bmatrix}, \quad (3)$$

where ${}^{TL}_{TR}\mathbf{R} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}$, ${}^{TL}_{TR}\mathbf{t} = [t_1 \ t_2 \ t_3]^T$, ${}^{TL}_{TR}\mathbf{R}$ is the coordinate rotation matrix, and ${}^{TL}_{TR}\mathbf{t}$ is the coordinate translation matrix.

According to (2) and (3), we can obtain

$$\begin{aligned} &(t_1 - X_{TR}t_3)(r_7X_{TL}Y_{TR} + r_8Y_{TL}Y_{TR} + r_9Y_{TR} - r_4X_{TL} \\ &\quad - r_5Y_{TL} - r_6) = (t_2 - Y_{TR}t_3)(r_7X_{TL}X_{TR} \\ &\quad + r_8X_{TR}Y_{TL} + r_9X_{TR} - r_1X_{TL} - r_2Y_{TL} - r_3). \end{aligned} \quad (4)$$

By measuring a target of certain length thrice using the two theodolites, the rotation matrix ${}^{TL}_{TR}\mathbf{R}$ and translation matrix ${}^{TL}_{TR}\mathbf{t}$ are obtained.

In the global coordinate system, the coordinate vector of any point P in the FOV is expressed as

$${}^{G}\mathbf{P} = \begin{bmatrix} \cot\alpha_{TL} \cdot {}^{G}z \\ \dfrac{\tan\beta_{TL}}{\sin\alpha_{TL}} \cdot {}^{G}z \\ \dfrac{t_1 - \cot\alpha_{TR}}{\cot\alpha_{TR}(r_7\cot\alpha_{TL} + r_8\tan\varphi_{TL}/\sin\alpha_{TL} + r_9) - (r_1\cot\alpha_{TL} + r_2\tan\varphi_{TL}/\sin\alpha_{TL} + r_3)} \end{bmatrix}, \quad (5)$$

where α_{TL} is the horizontal angle, φ_{TL} is the vertical angle obtained by the left theodolite, and α_{TR} is the horizontal angle obtained by the right theodolite.

3.2. Calibration of Binocular Cameras. Two industrial cameras with high resolution are employed in the 3D surface measurement system. To measure the surface of the large-scale

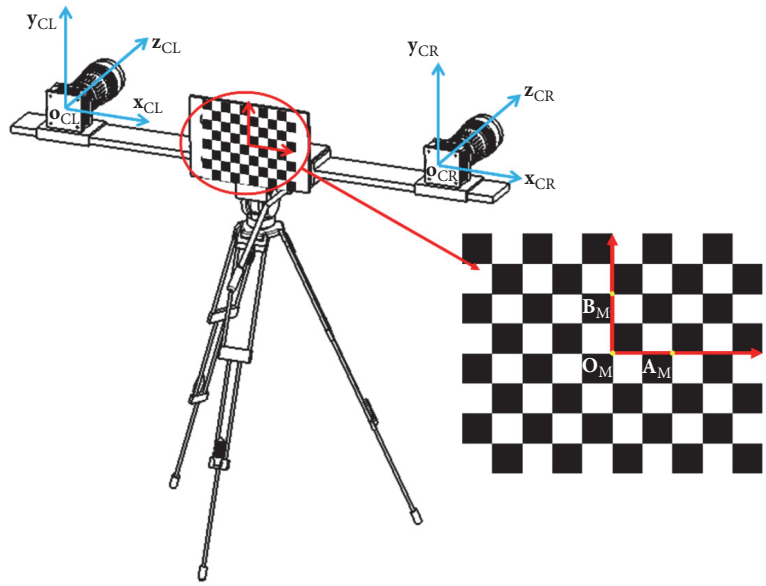


FIGURE 3: Binocular-camera measurement system with the transfer mark.

component that is blocked, the binocular cameras must be moved to different positions.

The pixel coordinates vector of point P in the image coordinate system is ${}^{C}\mathbf{p} = [{}^{C}u_P, {}^{C}v_P]^T$, and the coordinates vector in the World Coordinate System is ${}^{W}\mathbf{P} = [{}^{W}x_P, {}^{W}y_P, {}^{W}z_P]^T$. After the calibration of the cameras, the relation between ${}^{C}\mathbf{p}$ and ${}^{W}\mathbf{P}$ can be described by

$$\begin{bmatrix} {}^{C}\mathbf{p} \\ 1 \end{bmatrix} = \mathbf{K} \cdot \mathbf{H} \cdot \begin{bmatrix} {}^{W}\mathbf{P} \\ 1 \end{bmatrix} = \mathbf{K} \cdot \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0^T & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{W}\mathbf{P} \\ 1 \end{bmatrix}, \quad (6)$$

where $\mathbf{K}$ is the intrinsic matrix of the binocular vision system and $\mathbf{H}$ is the extrinsic matrix. $\mathbf{R}$ is the coordinate rotation matrix and $\mathbf{t}$ is the coordinate translation matrix. The coordinate system of the left camera is considered as the global coordinate system, so the transformation matrix, the rotation matrix, and the translation matrix from the coordinate system of the right camera to the global coordinate system are set to be ${}^{CR}_{CL}\mathbf{H}$, ${}^{CR}_{CL}\mathbf{R}$, and ${}^{CR}_{CL}\mathbf{t}$, respectively.

3.3. Calibration of the Relationship between the Binocular Cameras and the Transfer Mark. In the process of measurement, the transfer mark is essential for the transformation from the camera coordinate system to the global coordinate system.

As the transfer mark is not in the opposite direction to the binocular cameras, the feature points on the mark cannot be captured by the cameras. The transformation relation between the camera coordinate system and the transfer mark coordinate system cannot be directly established. As a result, the calibration process must be divided into the following two steps.

(1) Transformation from the Transfer Mark Coordinate System to the Global Coordinate System. To guarantee the transformation precision, a high-precision checkerboard is employed as the transfer mark. The checkerboard is placed in the opposite direction to the binocular cameras on the bracket, as shown in Figure 3.

The coordinate system of the transfer mark $o_M x_M y_M z_M$ is established with the origin at the centre point of the checkerboard O_M. The directions of the X_M axis and the Y_M axis are defined, as shown in Figure 3. The direction of Z_M axis is defined by the coordinate right-hand rule.

By measuring the feature points on the transfer mark with two theodolites, the transformation matrix ${}^{M}_{G}\mathbf{H}$ between the transfer mark coordinate system and the global coordinate system is acquired. The transformation relation can be expressed as

$$\begin{bmatrix} {}^{G}\mathbf{P} \\ 1 \end{bmatrix} = {}^{M}_{G}\mathbf{H} \begin{bmatrix} {}^{M}\mathbf{P} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{M}_{G}\mathbf{R} & {}^{M}_{G}\mathbf{t} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} {}^{M}\mathbf{P} \\ 1 \end{bmatrix}, \quad (7)$$

where ${}^{M}_{G}\mathbf{R}$ is the rotation matrix from the transfer mark coordinate system to the global coordinate system and ${}^{M}_{G}\mathbf{t}$ is the translation matrix.

(2) Transformation from the Binocular-Camera Coordinate System to the Global Coordinate System. The binocular-camera coordinate system is constructed on the left camera. The x_{CL} axis on the image plane is horizontal, and the y_{CL} axis on the image plane is vertical, as shown in Figure 3. First, feature points on the transfer mark are measured by two theodolites and reconstructed in the global coordinate system. Second, the same points are measured by binocular cameras and reconstructed in the binocular-camera coordinate system. With the two groups of coordinates,

the transformation relation ${}^{\mathrm{C}}_{\mathrm{G}}\mathbf{H}$ between the two coordinate systems can be written as follows:

$$\begin{bmatrix} {}^{\mathrm{G}}\mathbf{P} \\ 1 \end{bmatrix} = {}^{\mathrm{C}}_{\mathrm{G}}\mathbf{H} \begin{bmatrix} {}^{\mathrm{C}}\mathbf{P} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{\mathrm{C}}_{\mathrm{G}}\mathbf{R} & {}^{\mathrm{C}}_{\mathrm{G}}\mathbf{t} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} {}^{\mathrm{C}}\mathbf{P} \\ 1 \end{bmatrix}, \tag{8}$$

where ${}^{\mathrm{C}}_{\mathrm{G}}\mathbf{R}$ is the rotation matrix from the binocular-camera coordinate system to the global coordinate system and ${}^{\mathrm{C}}_{\mathrm{G}}\mathbf{t}$ is the translation matrix.

(3) Transformation from the Binocular-Camera Coordinate System to the Transfer Mark Coordinate System. As ${}^{\mathrm{M}}_{\mathrm{G}}\mathbf{H}$ and ${}^{\mathrm{C}}_{\mathrm{G}}\mathbf{H}$ were determined in the preceding section, the transformation matrix ${}^{\mathrm{C}}_{\mathrm{M}}\mathbf{H}$ from the binocular-camera coordinate system to the transfer mark coordinate system can be obtained based on the invariance of the spatial vector. The transformation relation can be expressed as follows:

$$\begin{bmatrix} {}^{\mathrm{M}}\mathbf{P} \\ 1 \end{bmatrix} = {}^{\mathrm{C}}_{\mathrm{M}}\mathbf{H} \begin{bmatrix} {}^{\mathrm{C}}\mathbf{P} \\ 1 \end{bmatrix} = {}^{\mathrm{M}}_{\mathrm{G}}\mathbf{H}^{-1}\, {}^{\mathrm{C}}_{\mathrm{G}}\mathbf{H} \begin{bmatrix} {}^{\mathrm{C}}\mathbf{P} \\ 1 \end{bmatrix}. \tag{9}$$

After the calibration, the relative position between the binocular cameras and the transfer mark remains unchanged. In the process of measurement, data transformation from the binocular-camera coordinate system to the global coordinate system is achieved by calibrating ${}^{\mathrm{M}}_{\mathrm{G}}\mathbf{H}$ with the two theodolites.

4. Acquisition and Reconstruction of a Large-Scale Three-Dimensional Surface of a Component

4.1. Local Image Processing and Reconstruction. In a single FOV, the feature information of the component is measured using a laser scanning method. The assisted laser stripes that are projected on the surface of the component are captured by a binocular vision measurement system.

Prior to reconstructing the laser stripes, the centres of the stripes are extracted using a grey centroid method [21]. The corresponding centre points of the left and right images are matched by the polar constraint of the cameras of the binocular vision system. The matching equation is expressed as

$$\mathbf{x}_{\mathrm{l}}^T \mathbf{F} \mathbf{x}_{\mathrm{r}} = 0, \tag{10}$$

where x_{l} is the coordinate vector of the laser stripe centre point in the left image and x_{r} is the coordinate vector of the laser stripe centre point in the right image. The fundamental matrix $\mathbf{F}$ can be calculated using two calibration target points with an exact distance [25].

After matching the feature points of the left and right images, reconstruction of the image can be realized according to the binocular reconstruction principle. The coordinate vector of any point P in the camera coordinate system can be expressed as

$${}^{\mathrm{C}}\mathbf{P} = \begin{bmatrix} \dfrac{{}^{\mathrm{C}}z^{\mathrm{CL}}X_P}{f_1} \\ \dfrac{{}^{\mathrm{C}}z^{\mathrm{CL}}Y_P}{f_1} \\ \dfrac{f_1\left(f_2 t_2' - {}^{\mathrm{CR}}Y_P t_3'\right)}{{}^{\mathrm{CR}}Y_P\left(r_7'^{\mathrm{CL}}X_P + r_8'^{\mathrm{CL}}Y_P + r_9' f_1\right) - f_2\left(r_4'^{\mathrm{CL}}X_P + r_5'^{\mathrm{CL}}Y_P + r_6' f_1\right)} \end{bmatrix}, \tag{11}$$

where ${}^{\mathrm{CL}}_{\mathrm{CR}}\mathbf{R} = \begin{bmatrix} r_1' & r_2' & r_3' \\ r_4' & r_5' & r_6' \\ r_7' & r_8' & r_9' \end{bmatrix}$ and ${}^{\mathrm{CL}}_{\mathrm{CR}}\mathbf{t} = [t_1' \;\; t_2' \;\; t_3']^T$. ${}^{\mathrm{CL}}_{\mathrm{CR}}\mathbf{R}$ is the rotation matrix from the right camera relative to the left camera, and ${}^{\mathrm{CL}}_{\mathrm{CR}}\mathbf{t}$ is the translation matrix from the right camera to the left camera; they can be obtained by the calibration of the binocular cameras. $({}^{\mathrm{CL}}X_P, {}^{\mathrm{CL}}Y_P)$ are the coordinates of point P in the image coordinate system of the left camera; $({}^{\mathrm{CR}}X_P, {}^{\mathrm{CR}}Y_P)$ are the coordinates of point P in the image coordinate system of the right camera. f_1 and f_2 are the effective focal length of the left camera and the effective focal length of the right camera, respectively.

4.2. Reconstruction of a 3D Surface of a Large-Scale Component. A binocular vision system is utilized to capture the large-scale component in different positions. Measuring the spatial positions of the feature points on the transfer mark with the two theodolites, ${}^{\mathrm{M}}_{\mathrm{G}}\mathbf{H}$ in different positions can be calculated. With the intrinsic parameters that are calibrated in the laboratory and the extrinsic parameters that are calibrated onsite, the local coordinates are transferred to global coordinates according to (8) and (10). The global reconstruction of a 3D surface of a large-scale component can be realized.

5. Measurement Experiments

The global measurement system for the 3D surface of a large-scale component was set up in a laboratory, as shown in Figure 4. The binocular vision system in the front consists of two CMOS cameras (Vieworks, VC-12 MC-M/C 65 with

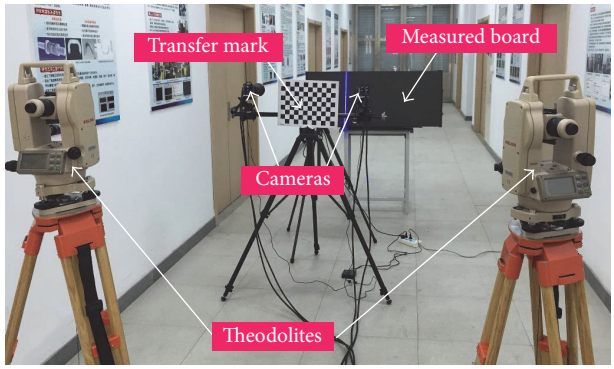


FIGURE 4: Global measurement system for 3D surface measurement.

35 mm lenses and a resolution of 3072 × 4096, the principled limitation of the resolution is about 0.5 mm) and a transfer mark (checkerboard of 12 × 9 checkers with an interval of 30 mm and an accuracy of 0.5 μm,). Two theodolites (Kolida, accuracy of 1″) are set in the back. According to the presented global calibration method, the theodolites and the relationship between the mark and the binocular cameras are precisely calibrated in the laboratory. The reconstruction experiment is conducted with this measurement system.

5.1. Calibration of the Global Measurement System and Accuracy Evaluation. According to the calibration method proposed in Section 3, the calibration experiments of the measurement system are conducted in the laboratory.

(1) Calibration of the System Parameters. The transformation matrix ${}^{C}_{M}\mathbf{H}$ is calibrated. Based on the coordinates of six spatial feature points and the orthogonal constraint condition of the rotation matrices, the rotation matrix and the translation matrix of the two-theodolite system are obtained as follows:

$$
{}^{TR}_{TL}\mathbf{R} = \begin{bmatrix} 0.8493 & -0.0001 & 0.5279 \\ 0.0091 & 0.9999 & -0.0144 \\ -0.5278 & 0.0170 & 0.8492 \end{bmatrix},
$$
$$
{}^{TR}_{TL}\mathbf{t} = \begin{bmatrix} -1576.7 \\ -17.3 \\ 496.3 \end{bmatrix}. \tag{12}
$$

The global coordinate system is constructed on the left theodolite.

The calibration results of the binocular cameras are shown as follows:

The intrinsic parameters of the left and right cameras are

$$
\mathbf{K}_{CL} = \begin{bmatrix} 6216.12 & 0 & 2120.73 \\ 0 & 6222.63 & 1578.84 \\ 0 & 0 & 1 \end{bmatrix},
$$
$$
\mathbf{K}_{CR} = \begin{bmatrix} 6199.83 & 0 & 2036.88 \\ 0 & 6202.82 & 1536.43 \\ 0 & 0 & 1 \end{bmatrix}, \tag{13}
$$

where $\mathbf{K}_{CL}$ and $\mathbf{K}_{CR}$ are the intrinsic parameters matrices of the left camera and the right camera, respectively.

The extrinsic parameters of the binocular cameras are

$$
{}^{CR}_{CL}\mathbf{R} = \begin{bmatrix} 0.876 & 0.007 & 0.482 \\ 0.020 & 0.998 & -0.050 \\ -0.482 & 0.054 & 0.875 \end{bmatrix},
$$
$$
{}^{CR}_{CL}\mathbf{t} = \begin{bmatrix} -905.92 \\ 2.94 \\ 204.82 \end{bmatrix}. \tag{14}
$$

According to the calibration method that was discussed in Section 3.2, ${}^{C}_{M}\mathbf{H}$ is obtained as follows:

$$
{}^{C}_{M}\mathbf{H} = \begin{bmatrix} 0.9890 & -0.1422 & -0.2400 & 107.02 \\ 0.1292 & 0.9772 & 0.1939 & 239.12 \\ 0.0713 & 0.1606 & 0.9626 & 278.80 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{15}
$$

(2) Calibration of Global Parameters. Based on the calibration principle of the two theodolites, the global measurement system is calibrated on one spot. Feature points on the transfer mark are measured using the two theodolites. Thus, ${}^{M}_{G}\mathbf{H}$ can be obtained as follows:

$$
{}^{M}_{G}\mathbf{H} = \begin{bmatrix} 0.0153 & 0.9768 & 0.2135 & -408.42 \\ 0.9987 & -0.0048 & -0.0499 & -465.07 \\ -0.0477 & 0.2140 & -0.9757 & 3905.85 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{16}
$$

With ${}^{C}_{M}\mathbf{H}$ and ${}^{M}_{G}\mathbf{H}$, any point in the FOV can be reconstructed in the global coordinate system.

To evaluate the measurement accuracy, a long one-dimensional (1D) target with two characteristic points (the distance between the two points is 1225.0214 mm) is considered. The measurement FOV is divided into two parts. The evaluation process is as follows: first, the target is placed in front of the cameras, as shown in Figure 5. Second, we used the cameras to measure the left characteristic point of the target and then moved the cameras to the second position to measure the right characteristic point of the target. Last, the coordinates of the two characteristic points that were measured by the binocular cameras at two positions are integrated into the global coordinate system. To guarantee repeatability, the 1D target is placed at three different positions and the evaluation of this process is repeated three times. The targets that were reconstructed in the global coordinate system are shown in Figure 6. To evaluate the

(a) (b)

Figure 5: Target measurement experiment. (a) 1D long target with two feature points. (b) First measurement position of the cameras.

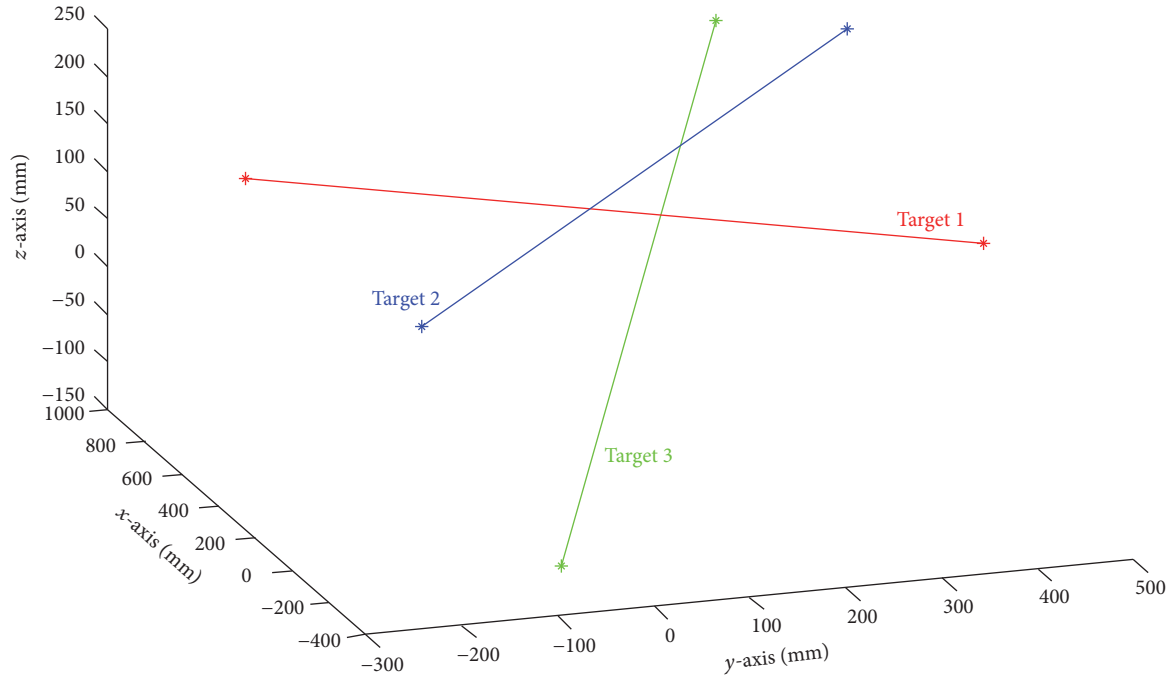


Figure 6: Reconstruction of the target at three different positions.

accuracy of the measurement system, the deviation between the real length and the measured length of the 1D target is calculated and listed in Table 1. The results indicate a maximum global measurement deviation of 0.103% for the proposed method.

5.2. 3D Measurement of the Global Measurement System. To verify the feasibility of the proposed method, a standard flat part with the size of 600 mm × 800 mm is measured in the laboratory. Before the experiments, the standard flat part was measured using a three-coordinate measuring machine (Zeiss Prismo Navigator) to obtain the measuring reference data. By comparing the reconstruction results with the actual value, the construction accuracy of the global measurement system is validated. The experimental system is shown in Figure 7. According to the calibration calculation of global system, the calibration results of binocular cameras and global system are illustrated in Tables 2 and 3, respectively.

In the experiment, the measurement FOV is divided into four parts based on the size of the plate. Next, the four parts of the board are measured by the respective cameras at different positions. Images of the laser stripes, which are projected on the board, are captured by the cameras. The centres of laser stripes are extracted and matched based on image processing method in Section 4.1. By monitoring the transfer mark with the two theodolites, the local 3D data from two measurements are united with the global coordinate system. The results of feature extraction and transformation matrix of different positions are shown in Table 4. The reconstructed

Table 1: Evaluation of the measurement accuracy (mm).

Position	x/mm	y/mm	z/mm	Measured value/mm	Standard value/mm	Deviation/%
1	375.4370	−234.7837	166.1296	1224.63		0.032%
	−185.4221	852.2557	107.0487			
2	−203.9730	−205.5046	131.8487	1223.753	1225.0214	0.103%
	437.1248	834.3868	203.9114			
3	−79.5582	−312.6632	−114.0780	1224.81		0.017%
	283.6261	801.1597	233.7018			

Table 2: The calibration results of binocular cameras.

	Left camera	Right camera
Intrinsic matrix	$K = \begin{bmatrix} 5051.8 & 0 & 2093.7 \\ 0 & 5049.7 & 1589.0 \\ 0 & 0 & 1 \end{bmatrix}$	$K = \begin{bmatrix} 4999.1 & 0 & 2065.0 \\ 0 & 4998.6 & 1539.5 \\ 0 & 0 & 1 \end{bmatrix}$
Transformation matrix	${}^{\mathrm{CL}}_{\mathrm{CR}}\mathbf{H} = \begin{bmatrix} 0.8863 & -0.0795 & 0.4562 & -569.14 \\ 0.0694 & 0.9968 & 0.0389 & -23.749 \\ -0.4579 & -0.0028 & 0.8890 & 103.70 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	

Table 3: The calibration results of global system.

	${}^{\mathrm{TL}}_{\mathrm{TR}}\mathbf{H}$	${}^{\mathrm{C}}_{\mathrm{M}}\mathbf{H}$
Transformation matrix	${}^{\mathrm{TL}}_{\mathrm{TR}}\mathbf{H} = \begin{bmatrix} 0.9997 & 0.0020 & -0.0252 & 1392.6 \\ -0.0020 & 1.0000 & -0.0001 & -14.512 \\ 0.0252 & 0.0002 & 0.9997 & 42.612 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	${}^{\mathrm{C}}_{\mathrm{M}}\mathbf{H} = \begin{bmatrix} 0.9618 & -0.0248 & 0.2726 & -254.48 \\ 0.0005 & -0.9957 & -0.0925 & -10.851 \\ 0.2738 & 0.0891 & -0.9577 & -218.60 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

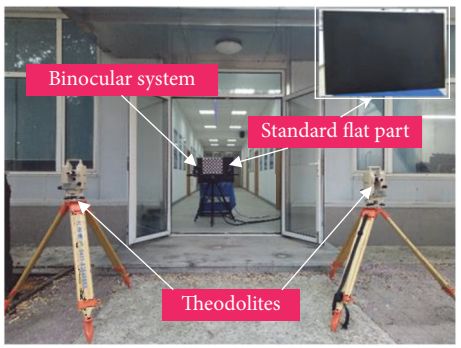


Figure 7: Reconstruction of the target at different positions.

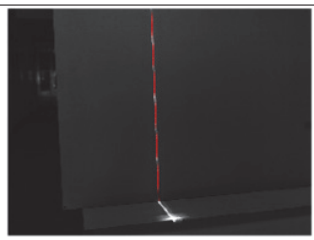
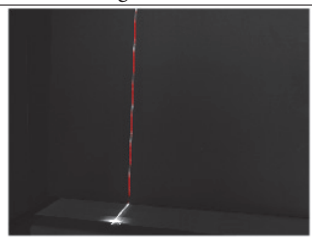
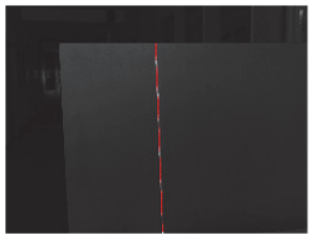
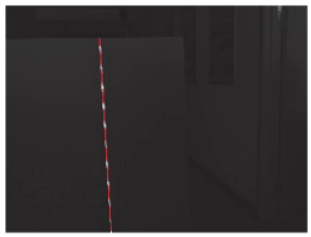
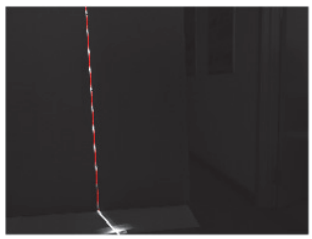
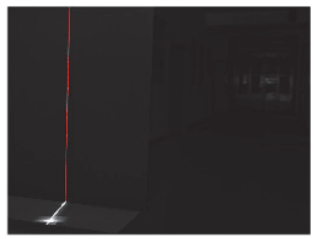
results are shown in Figure 8. The 3D reconstruction points of measured part are compared with the ideal standard plane. The experimental results reveal the construction deviation is 0.14% with the corresponding measurement field.

To verify the validity of the system, a curved composite part is measured. The part is shown in Figure 9(a) and the reconstruction result is shown in Figure 9(b). The results indicate that the proposed system can effectively measure curved components.

6. Conclusions

In this paper, a global measurement method that is based on a multiple FOV combination was proposed for measuring the 3D surface of a large-scale component. Compared with existing methods, the proposed method has the advantage of high efficiency and no requirement for pasting a target mark on the component. As the cameras with the transfer mark can be placed at any position in the measurement range of theodolites, the measurement system is more flexible for large-scale component measurement. The experimental results in the laboratory indicate that a maximum accuracy of the measurement system of 0.103% can be attained when the length of the 1D target is approximately 1.225 m. For the large board measurement experiment, the reconstruction accuracy is less than 0.14%. Thus, the proposed method is practicable and suitable for measuring the 3D surface of a large-scale component in an industrial site. The measurement system can be employed in the assembly process of large-scale industry components. Additional research is suggested to improve the

Table 4: The results of feature extraction and transformation in different positions.

Position	Results of feature extraction		Transformation matrix ${}^{M}_{G}\mathbf{H}$
	Left camera	Right camera	
1			${}^{M}_{G}\mathbf{H}_1 = \begin{bmatrix} 0.9900 & 0.0209 & -0.1398 & 922.80 \\ 0.0185 & -0.9997 & -0.0178 & -609.54 \\ -0.1401 & 0.0151 & -0.9900 & 3258.3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
2			${}^{M}_{G}\mathbf{H}_2 = \begin{bmatrix} 0.9926 & 0.0230 & -0.1190 & 926.98 \\ 0.0211 & -0.9996 & -0.0177 & -752.53 \\ -0.1194 & 0.0150 & -0.9927 & 3262.8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
3			${}^{C}_{M}\mathbf{H}_3 = \begin{bmatrix} 0.9997 & 0.0187 & -0.0156 & 1480.4 \\ 0.0184 & -0.9996 & -0.0211 & -751.85 \\ -0.0160 & 0.0208 & -0.9997 & 3256.4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
4			${}^{C}_{M}\mathbf{H}_4 = \begin{bmatrix} 0.9996 & 0.0162 & -0.0222 & 1475.8 \\ 0.0157 & -0.9996 & -0.0237 & -608.88 \\ -0.0226 & 0.0233 & -0.9995 & 3251.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

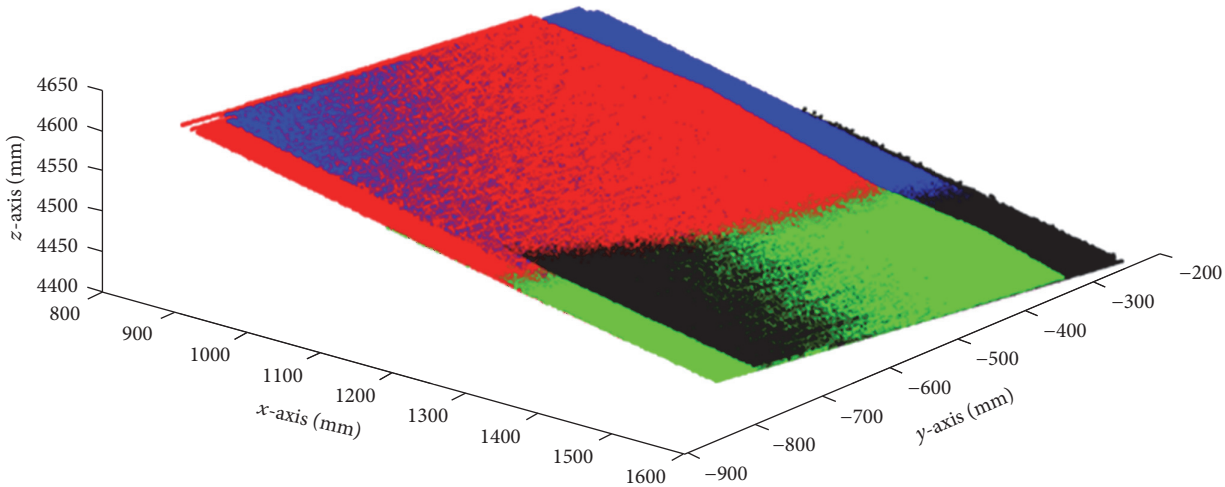
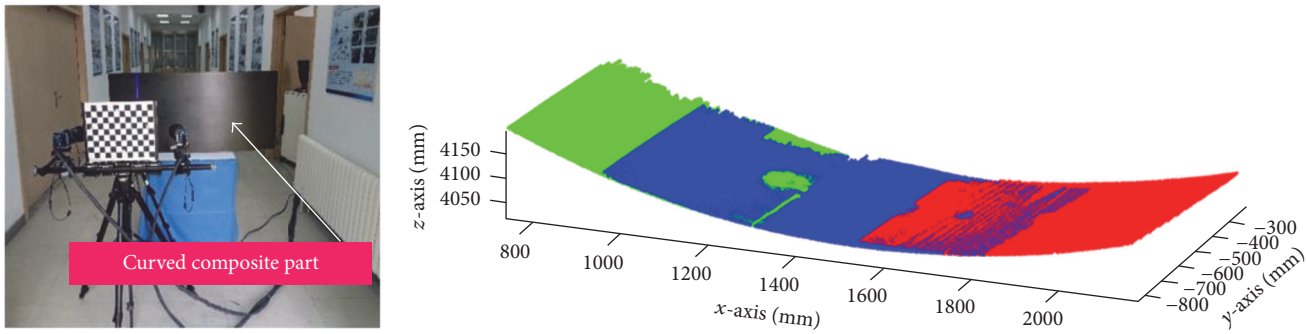


Figure 8: Reconstruction results of the large plate in the global coordinate system.



(a) (b)

FIGURE 9: Experimental results of a curved composite part in the global coordinate system. (a) The measured curved composite part. (b) Reconstruction results.

global measurement accuracy by optimizing the calibration process and improving the calibration precision.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This paper is supported by the National Basic Research Program of China 973 Project (Grant no. 2014CB046504), the National Science Foundation for Outstanding Young Scholars of China (no. 51622501), the National Natural Science Foundation of China (Grant no. 51375075), and the Science Fund for Creative Research Groups (Grant no. 51621064).

References

[1] H. Kieu, T. Pan, Z. Wang, M. Le, H. Nguyen, and M. Vo, "Accurate 3D shape measurement of multiple separate objects with stereo vision," *Measurement Science and Technology*, vol. 25, no. 3, Article ID 035401, 2014.

[2] Z. Liu, J. Zhu, L. Yang, H. Liu, J. Wu, and B. Xue, "A single-station multi-tasking 3D coordinate measurement method for large-scale metrology based on rotary-laser scanning," *Measurement Science and Technology*, vol. 24, no. 10, Article ID 105004, 2013.

[3] S. Aghaie, S. Khanmohammadi, H. Moghadam-Fard, and F. Samadi, "Adaptive vision-based control of robot manipulators using the interpolating polynomial," *Transactions of the Institute of Measurement and Control*, vol. 36, no. 6, pp. 837–844, 2014.

[4] F. Chen, G. M. Brown, and M. Song, "Overview of three-dimensional shape measurement using optical methods," *Optical Engineering*, vol. 39, no. 1, pp. 10–22, 2000.

[5] D. Ponsa, J. Serrat, and A. M. López, "On-board image-based vehicle detection and tracking," *Transactions of the Institute of Measurement and Control*, vol. 33, no. 7, pp. 783–805, 2011.

[6] E. Shi, J. Guo, H. Zhou, and W. Shao, "Study on on-line measurement technology for large-scale sheet parts with free-form surface," *Chinese Journal of Scientific Instrument*, vol. 9, article 004, 2009.

[7] Y. Ling and Y. Jiahe, "The 3D surface measurement and simulation for turbine blade surface based on color encoding structural light," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 8, no. 3, pp. 273–280, 2015.

[8] J. Shi, Z. Sun, and S. Bai, "Large-scale three-dimensional measurement via combining 3D scanner and laser rangefinder," *Applied Optics*, vol. 54, no. 10, pp. 2814–2823, 2015.

[9] C. Mei, Z. Fei, and Y. Zhang, "Virtual surface measurement of large deployable space antenna structure," *Advanced Materials Research*, vol. 479-481, pp. 2586–2592, 2012.

[10] C. Schwartz, R. Sarlette, M. Weinmann, M. Rump, and R. Klein, "Design and implementation of practical bidirectional texture function measurement devices focusing on the developments at the university of Bonn," *Sensors*, vol. 14, no. 5, pp. 7753–7819, 2014.

[11] Z. Jia, X. Ma, W. Liu et al., "Pose measurement method and experiments for high-speed rolling targets in a wind tunnel," *Sensors*, vol. 14, no. 12, pp. 23933–23953, 2014.

[12] K. Iwata, Y. Sando, K. Satoh, and K. Moriwaki, "Application of generalized grating imaging to pattern projection in three-dimensional profilometry," *Applied Optics*, vol. 50, no. 26, pp. 5115–5121, 2011.

[13] F. J. Brosed, J. J. Aguilar, D. Guillomïa, and J. Santolaria, "3D geometrical inspection of complex geometry parts using a novel laser triangulation sensor and a robot," *Sensors*, vol. 11, no. 1, pp. 90–110, 2011.

[14] Q. K. Dang, Y. Chee, D. D. Pham, and Y. S. Suh, "A virtual blind cane using a line laser-based vision system and an inertial measurement unit," *Sensors*, vol. 16, no. 1, article 95, 2016.

[15] Z.-F. Zhang, Z. Gao, Y.-Y. Liu et al., "Computer vision based method and system for online measurement of geometric parameters of train wheel sets," *Sensors*, vol. 12, no. 1, pp. 334–346, 2012.

[16] Aerospace Applications, http://www.geodetic.com/applications/aerospace.aspx .

[17] H. Jiang, H. Zhao, and X. Li, "High dynamic range fringe acquisition: a novel 3-D scanning technique for high-reflective surfaces," *Optics and Lasers in Engineering*, vol. 50, no. 10, pp. 1484–1493, 2012.

[18] Fuselage and Cabin, http://www.gom.com/industries/aerospace/fuselage-cabin.html.

[19] P. Feng and Z.-Z. Wei, "Light probe based large FOV 3D vision measurement system," *Guangxue Jingmi Gongcheng/Optics and Precision Engineering*, vol. 21, no. 9, pp. 2217–2224, 2013.

[20] J. J. Craig, *Introduction to Robotics. Mechanics and Control*, Series in Electrical and Computer Engineering: Control Engineering, Addison-Wesley, Reading, Mass, USA, 1989.

[21] W. Liu, Y. Zhang, F. Yang et al., "A measurement method for large parts combining with feature compression extraction and directed edge-point criterion," *Sensors*, vol. 17, no. 1, article 40, 2017.

[22] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[23] F. Zhou, G. Zhang, J. Jiang, B. Wu, and S. Ye, "Three-dimensional coordinate measuring systemwith binotheodolites on site," *Chinese Journal of Mechanical Engineering*, vol. 1, article 032, 2004.

[24] Z. Zhang, J. Zhu, N. Geng, H. Zhou, and S. Ye, "The design of double-theodolite 3D coordinate measurement system," *Chinese Journal of Sensors and Actuators*, vol. 5, article 014, 2010.

[25] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2nd edition, 2003.

Hindawi
Journal of Sensors
Volume 2017, Article ID 3515418, 8 pages
https://doi.org/10.1155/2017/3515418

Research Article

Multisource Data Fusion Framework for Land Use/Land Cover Classification Using Machine Vision

Salman Qadri,[1] Dost Muhammad Khan,[1] Syed Furqan Qadri,[2] Abdul Razzaq,[3] Nazir Ahmad,[1] Mutiullah Jamil,[4] Ali Nawaz Shah,[1] Syed Shah Muhammad,[5] Khalid Saleem,[4] and Sarfraz Ahmad Awan[5]

[1]*Department of Computer Science & IT, The Islamia University of Bahawalpur, Punjab 63100, Pakistan*
[2]*Key Laboratory of Photo-Electronic Imaging Technology and System, School of Computer Science, Beijing Institute of Technology (BIT), Beijing 100081, China*
[3]*Department of Computer Science, NFC IET, Multan, Punjab 60000, Pakistan*
[4]*Department of Computer Sciences, Quaid-i-Azam University, Islamabad 45320, Pakistan*
[5]*Department of Computer Science, Virtual University of Pakistan, Lahore, Punjab 54000, Pakistan*

Correspondence should be addressed to Salman Qadri; salman.qadri@iub.edu.pk

Received 21 April 2017; Revised 25 July 2017; Accepted 8 August 2017; Published 11 September 2017

Academic Editor: Julio Rodriguez-Quiñonez

Data fusion is a powerful tool for the merging of multiple sources of information to produce a better output as compared to individual source. This study describes the data fusion of five land use/cover types, that is, bare land, fertile cultivated land, desert rangeland, green pasture, and Sutlej basin river land derived from remote sensing. A novel framework for multispectral and texture feature based data fusion is designed to identify the land use/land cover data types correctly. Multispectral data is obtained using a multispectral radiometer, while digital camera is used for image dataset. It has been observed that each image contained 229 texture features, while 30 optimized texture features data for each image has been obtained by joining together three features selection techniques, that is, Fisher, Probability of Error plus Average Correlation, and Mutual Information. This 30-optimized-texture-feature dataset is merged with five-spectral-feature dataset to build the fused dataset. A comparison is performed among texture, multispectral, and fused dataset using machine vision classifiers. It has been observed that fused dataset outperformed individually both datasets. The overall accuracy acquired using multilayer perceptron for texture data, multispectral data, and fused data was 96.67%, 97.60%, and 99.60%, respectively.

1. Introduction

The conventional methodologies are present to measure and monitor the land use/land cover (LU/LC) for regional and global environment changes [1]. The real-time LU/LC data is very important for resource management, future prediction, crop growth assessment, and sustainable development [2]. Although conventionally LU/LC data is collected through field base survey, remote sensing data collection has its own importance due to time, accuracy, and transparency factors and so forth. During the last decade, space-borne multispectral data have proved more beneficial over ground and airborne data for land monitoring, assessment, and accurate information due to their increased spectral resolution. Previously single source dataset is mostly used for LU/LC classification but recently multisource dataset is used for better overall accuracy results. Land cover is a primary factor that plays an important role for physical and chemical variation in environment. The change in LU/LC can be accurately identified by monitoring the regional and global classification maps continuously. When remote sensing data is used along with ground truth data then it provides reliable and cost-effective LU/LC information. Remote sensing mostly used the synthetic aperture radar (SAR) data for LU/LC information but cloudy weather is one of the major obstacles to acquire the information through optical imagery. It has been strengthened the significance of new tools and techniques for acquiring LU/LC thematic information from

remote sensing data [3]. In recent years, satellite-based remote sensing data have been very hot research area for earth scientists. Many researchers have worked on combining the spectral and optical data, which enhanced discrimination power of integrated data and their overall classification accuracy results [4, 5], and described the simple fused model for land cover classification which is named fused mixture model. The spatial and temporal adaptive-reflectance fusion model (STARFM) was proposed by [6], which gave the better accuracy results. For earth observation applications, remotely accessed sensor base multispectral data provides better large-scale information as compared to optical data [7]. The fusion techniques enhance the operational capabilities of dataset with respect to other tuning factors and overall accuracy results [8]. In data fusion, two or more datasets are merged together to acquire one single dataset with the entire dataset features individually [9]. The low resolution multispectral dataset is fused with high resolution optical radar dataset to get the better results in terms of spatial resolution and overall classification accuracy [10]. Huang with his companion described that LU/LC is the coarse dataset in spatial resolution and changes frequently when observing through remote sensing and it is very difficult to measure and monitor the change accurately [11]. Different image fusion techniques with their constraints in implementation stages are discussed by [12, 13]. They proved quantitatively that fusion plays important role in better interoperational capabilities and reduces the ambiguity linked with the data acquired by different sensors or by same sensor with temporal variation. Quartz rich and poor mineral types are identified by using the image fusion method with the implementation of supervised classifier maximum likelihood (ML) and acquired overall accuracy and kappa coefficient of 96.53% and 0.95, respectively [14].

In this study, it has been tried to design a framework for analyzing the potential of multispectral dataset fused with texture feature dataset for the discrimination of different LU/LC classes.

2. Study Area

This study explains the data fusion technique for LU/LC classification instead of traditional ground base field surveys. All the experimentations have been performed at Islamia University of Bahawalpur Punjab province (Pakistan) located at $29^{\circ}23'44''$N and $71^{\circ}41'1''$E. This study describes the LU/LC monitoring, management, and classification using fused dataset generated by the combination of photographic and multispectral radiometric data, which is mostly bare and deserted rangeland. It would provide accurate results for LU/LC cover changes and prediction for better crop yield assessment.

3. Dataset

For this study, multispectral dataset is obtained by using the device named Multispectral Radiometer Crop Scan (MSR5). It gives data which is equivalent to the Satellite Landsat TM (Thematic Mapper) [15]. It has five spectral bands, that is, blue (B), green (G), red (R), near infrared (NIR), and shortwave infrared (SWIR) ranges from 450 nanometers to 1750 nanometers, while digital photographic data are acquired by a high resolution digital NIKON Coolpix camera having 10.1-megapixel resolution.

4. Material and Methods

The objective of this study is to analyze the five types of LU/LC multispectral data with the digital photographic data. A multisource data fusion frame work is designed to classify the subjective LU/LC type's data accurately. Different image processing techniques have been applied on photographic data, that is, color to gray scale conversion, enhanced contrast, and image sharpening procedure. A still camera is mounted at 4-feet height stand and acquired five types of LU/LC images dataset. For image dataset, 20 images of each LU/LC with the dimension of 4288×3216 pixels with 24-bit depth of jpg format have been acquired. To increase the size of image dataset, 5 nonoverlapping region of interests (ROIs) with different pixels size, that is, (32×32), (64×64), (128×128), (256×256), and (512×512), have been taken on each image with the dimension of (4288×3216) and a total of 100 (20×5) images of above discussed sizes have been developed for each land type and a dataset containing 500 images on five types of LU/LC has been developed for experimentations. Similarly, for multispectral dataset, five spectral bands data are acquired and each band comprises visible ranges, that is, B, G, and R, from 400 nanometers to 700 nanometers, invisible bands near infrared (NIR) ranging from 750 nanometers to 900 nanometers, and shortwave infrared ranges from 1350 nanometers to 1650 nanometers. The multispectral dataset are acquired using a device multispectral radiometer (MSR5) serial number 566. For (MSR5) dataset, it has been observed that 100 scans of each of the LU/LC types and a total of 500 scans are acquired on the same location where digital images have been acquired. To avoid sun shadow effect, whole data gathering process has been completed at noon time (1.00 pm to 2.30 pm) under clear sky.

Experimentation. This study is unique because there is no need for any special laboratory setup. For image dataset, prior to further processing, different sizes of images have been converted from color to gray scale images (8 bits) and stored in bitmap (.bmp) format because MaZda software works better to calculate texture features in this format. The contrast level of grayscale images has been enhanced by using the image converter software. Now image dataset has been ready to calculate the first-order histogram and second-order texture parameters. MaZda software has been used to calculate 9 histogram features and 11 second-order texture features (Haralick) using gray level cooccurrence matrix (GLCM) in four dimensions, that is, 0°, 45°, 90°, and 135° up to 5-pixel distance and calculated 220 $(11 \times 4 \times 5)$ texture features with 9 histogram features and 229 features in total for each ROI. It has been observed that total 114500 (229×500) features space for whole image dataset have been calculated [16]. It is important to be mentioned here that it

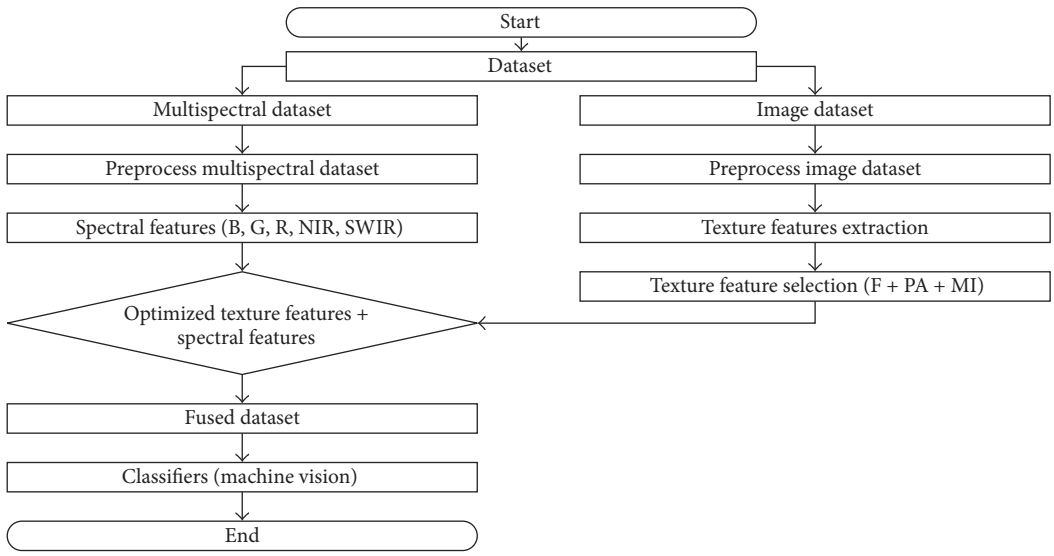


Figure 1: Proposed framework for LU/LC fused dataset classification.

is not so easy to handle this large-scale feature space that is why three feature selection techniques, namely, Fisher (F), Probability of Error plus Average Correlation Coefficient (PE + AC), and Mutual Information (MI), have been employed to extract optimized features dataset. These three techniques have been merged together (F + PA + MI) and extracted thirty most discriminant features (10 features by each technique) out of 229 features space for each (ROI) image dataset. All the experimentations have been performed using MaZda software version 4.6 with Weka data mining tool version (3.6.12) on Intel® Core i3 processor 2.4 gigahertz (GHz) with a 64-bit operating system [17].

Proposed Methodology. The proposed methodology has been described in Figure 1. First data fusion algorithm has been described with all procedural steps.

Data Fusion Algorithm

Start main

{

Input ϵ Multispectral and Photographic land use/Land cover dataset

For {

Step 1 to Step 7

Step 1 = Photographic and multispectral datasets ϵ five land types.

Step 2 = Data Preprocessing

Step 3 = Developed co-occurrence matrix for photographic dataset and extract texture features

Step 4 = Multispectral dataset with five spectral bands ϵ visible and invisible wavelength

Step 5 = Three feature selection techniques, fisher (F), probability of error plus average correlation (POE + AC) and mutual information (MI) are merged (F + PA + MI) and employed on photographic dataset.

Step 6 = Extract 30 optimized texture features dataset

Step 7 = 30 optimized texture features + 5 spectral features ϵ fused dataset

End For

}

Step 8 = Machine vision classifiers are employed on fused dataset

Output = Land classification Results

End main

}

Now Figure 1 describes the proposed methodology in detail. At first step, two different types of datasets are acquired, that is, image dataset and multispectral dataset. The second step employs different image preprocessing filters, that is, Sobel or Laplacian, to sharpen the images and extract first-order and 2nd-order texture features. In step three, optimized features dataset has been acquired by implementing three combined feature selection techniques (F + PA + MI) and 30 most discriminant features are extracted. These 30 optimized texture features are shown in Table 1. It has been observed in Table 2 that the Mutual Information (MI) based selected texture features are very much correlated, namely, "inverse difference moment," but these features have variation in interpixel distance and dimension and, due to this variation, their computed values are also different. For every pixel distance (d) and angular dimension (θ), the different calculated values are acquired for this texture feature which is "inverse difference moment." For this study, we have taken $d = 1, 2, 3, 4$ and 5-pixel distance with angle dimension $\theta = 0^\circ$, 45°, 90°, and 135°. So, as a result, we cannot ignore any value of the given texture features. MI based texture features values

Table 1: Integrated texture feature selection table (F + PA + MI) for ROI (512 × 512).

F + PA + MI
1 S(0, 4)InvDfMm
2 S(0, 5)InvDfMm
3 S(0, 3)InvDfMm
4 S(4, 4)InvDfMm
5 S(0, 2)InvDfMm
6 S(5, 5)InvDfMm
7 S(3, 3)InvDfMm
8 S(2, 2)InvDfMm
9 S(3, 3)InvDfMm
10 S(0, 1)InvDfMom
11 S(0, 2)SumEntrp
12 S(0, 5)DifEntrp
13 Perc.01%
14 S(1, 0)Correlate
15 S(5, 5)Entropy
16 S(5, 5)SumAverg
17 Skewness
18 S(0, 3)AngScMom
19 S(0, 2)SumVarnc
20 S(1, 0)InvDfMom
21 S(0, 3)Correlate
22 S(0, 3)Contrast
23 S(0, 4)Correlate
24 S(2, 2)Correlate
25 S(2, 2)Contrast
26 S(0, 4)Contrast
27 S(0, 1)Entropy
28 S(0, 5)Correlate
29 S(0, 2)Correlate
30 S(0, 3)SumVarnc

actually describe the LU/LC dataset into its own direction and as whole these features disclose the entire texture patterns. It has been discussed by many researchers [10–14] that five control features, that is, window size, texture derivative(s), input channel (i.e., spectral channel to measure the texture), quantization level of output channel (8 bits, 16 bits, and 32 bits), and the spatial components, that is, interpixel distance and angle during cooccurrence matrix computation, play a very important role during the analysis of texture features.

In the fourth step, these 30 texture features are combined with 5 multispectral datasets and a fused dataset is developed with the combination of two different sources of data [18].

Table 1 describes the optimized texture feature dataset while Table 2 describes the multispectral feature dataset.

In the last step, this fused dataset has been deployed to different machine vision classifiers, that is, artificial neural network (MLP), Naïve Bayes (NB), Random Forest (RF), and J48; here j48 is the implementation of C4.5 algorithm of decision tree in Weka software. Figure 1 describes the multisource data fusion framework for LU/LC classification.

5. Results and Discussion

It has been observed that, as discussed above for image dataset, four ROIs with different pixel sizes, that is, 32 × 32, 64 × 64, 128 × 128, and 256 × 256, do not give satisfactory results for classification. The overall classification accuracy of less than 75% has been observed by implementing the MLP, NB, j48, and RF classifiers on the basis of these four ROIs which have not been acceptable, while, on ROI 512 × 512, the promising results for image data classification are provided. Finally to generate the fused dataset, the ROI of size 512 × 512 has been merged with multispectral dataset. For classification, different machine vision classifiers have been employed on this fused dataset using Weka software version (3.6.12), that is, Multilayer Perceptron (MLP), Naïve Bayes (NB), Random Forest (RF), and J48 [19]. These machine vision classifiers are employed on optimized fused dataset. Before deploying the fused dataset on Weka software, it has been converted into the Attribute Relation File Format (ARFF). This fused dataset has also been compared to both individual texture and multispectral dataset. These machine vision approaches have the potential to analyze the fused dataset. For this fused dataset, it has been separated into 66% for training and 34% for testing with 10-fold cross-validation method and same strategy also has been implemented for individual datasets, namely, multispectral data and texture. Besides this, quite a few other performance evaluating factors, that is, mean absolute error (MAE), root mean squared error (RMSE), confusion matrix, true positive (TP), false positive (FP), receiver-operating characteristic (ROC), time complexity (T), and overall accuracy (OA), have also been calculated. At first, the fused dataset for LU/LC classification has been employed with different machine vision classifiers, namely, MLP, RF, NB, and J48 with an optimized set of 35 features that have shown different accuracy results. The overall accuracy with different performance oriented parameters are shown in Table 3.

Table 4 represents a confusion matrix of fused dataset; it includes the information which is extracted by deploying the MLP classifier and diagonal of table shows the maximal values which are placed in five different LU/LC classes. MLP shows the best overall accuracy among different employed classifiers.

Fused dataset LU/LC classification graph of MLP is shown in Figure 2. This shows that each type of dataset has 100 data instances (ROIs) and these ROIs or data have been classified into their five classes. Graphically data have been classified into five LU/LC classes, that is, "blue color" for fused dataset, "green color" for texture, and "red color" for multispectral dataset. Figure 2 explained the LU/LC data classification in MLP graph. Similarly, for texture and multispectral dataset, the same classifiers with same strategy have been employed as discussed in the above fused dataset. For texture dataset, 30 optimized texture features [20] have been deployed while, for multispectral dataset, 5 spectral features have been individually implemented. It has been observed that, for both texture and multispectral dataset, MLP classifier has shown the higher overall accuracy as compared to the others deploying classifiers. As a result, the deployed MLP

Table 2: Multispectral feature table.

MSR Types	Blue	Green	Red	Near infrared	Shortwave infrared
MSR5 (generic)	450–520 nm	520–600 nm	630–690 nm	760–930 nm	1550–1750 nm
MSR5 (S. number 566)	485 nm	560 nm	660 nm	830 nm	1650 nm

Table 3: Fused dataset classification table.

Classifiers	Kappa statistics	TP rate	FP rate	ROC	MAE	RMSE	Time (sec)	OA
Multilayer perceptron	0.995	0.996	0.001	1	0.0048	0.0303	4.86	99.6%
Random Forest	0.994	0.995	0.002	1	0.0195	0.064	0.58	99.5%
J48	0.9675	0.974	0.007	0.986	0.0104	0.1011	0.24	97.4%
Naïve Bayes	0.8950	0.916	0.021	0.99	0.0351	0.1814	0.01	91.6%

Table 4: Fused dataset confusion table for multilayer perceptron (MLP).

Classes	Bare land	Desert rangeland	Fertile cultivated land	Green pasture	Sutlej basin river land
Bare land	100	0	0	0	0
Desert Rangeland	1	99	0	0	0
Fertile Cultivated land	0	0	99	0	1
Green pasture	0	0	0	100	0
Sutlej basin river Land	0	1	0	0	99

Table 5: Texture data classification table.

Classifiers	Kappa statistics	TP rate	FP rate	ROC	MAE	RMSE	Time (sec)	OA
Multilayer perceptron	0.9702	0.976	0.059	0.989	0.0189	0.0969	2.26	96.67%
Random Forest	0.8782	0.902	0.025	0.987	0.0835	0.1756	0.40	90.35
J48	0.7536	0.810	0.050	0.890	0.0850	0.2760	0.20	80.34%
Naïve Bayes	0.6960	0.757	0.060	0.941	0.0988	0.3062	0.03	75.70%

classifier showed the higher overall accuracy with others performance evaluating parameters including kappa coefficient, TP, FP, ROC, MAE, RMSE, and time complexity factors [21, 22]. The overall accuracy with different performance evaluating parameters is shown in Table 5.

Table 6 represents a confusion matrix for texture dataset; it contains the information which is actual and predicted data for MLP classifier. MLP shows the best overall accuracy among different employed classifiers. Texture dataset LU/LC classification graph of MLP is shown in Figure 2. This shows that each type of dataset has 100 data instances (ROIs) and these ROIs or data have been classified into their five classes.

Figure 2 explained the LU/LC data classification in MLP graph. The overall accuracy of multispectral dataset with different performance evaluating parameters with details is shown in Table 7 [23].

It contains the information which is actual and predicted data for MLP classification system. MLP shows the best overall accuracy among different employed classifiers for multispectral datasets. MLP confusion table for multispectral dataset is shown in Table 8.

Multispectral LU/LC dataset classification graph of MLP is shown in Figure 2. This shows that each type of dataset has 100 data instances or (ROIs) and these ROIs or data have been classified into their five classes. Figure 2 explained the LU/LC data classification in MLP graph.

Finally, a comparative LU/LC classification graph of fused, multispectral, and texture dataset using MLP classifier is shown in Figure 3. This shows that each type of dataset has 100 data instances (ROIs) and these ROIs or data have been classified into their five classes. The classification graph for MLP classifier is shown in Figure 3. It has been observed that fused dataset has relatively better overall accuracy as compared to multispectral and texture dataset [24]. It shows that data fusion plays a vital role for better land assessment, management, and accurate monitoring purposes [25, 26].

6. Conclusions

This study is focused on the classification of five different types of LU/LC datasets. Four data mining classifiers, that is, MLP, RF, NB, and J48, have been employed on fused, texture, and multispectral dataset. These three types of dataset (fused, texture, and multispectral) have been examined for overall accuracy in classification with some other performance evaluating factors as discussed above in Results and Discussion. All the classifiers have shown satisfactory results, but multilayer perceptron (MLP) result was considerably better among all of them. It has been observed that, after deploying MLP, an overall accuracy of 96.67% for texture data, 97.60% for multispectral data, and 99.60 for fused dataset has been observed. Fused dataset has shown better overall accuracy

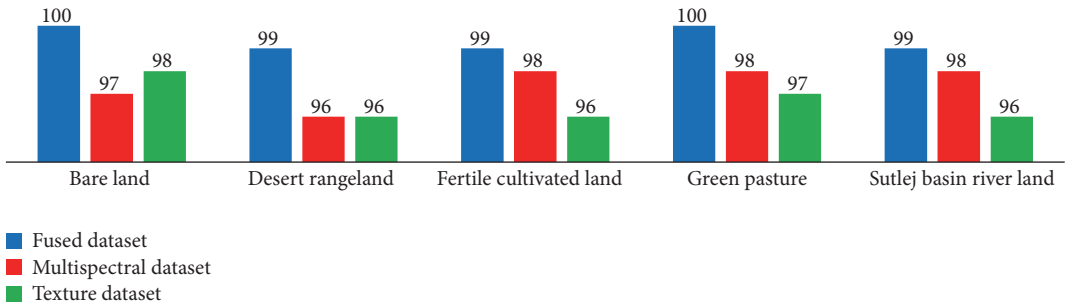


Figure 2: Classification graph using multilayer perceptron (MLP).

Table 6: Confusion table of texture data for multilayer perceptron (MLP).

Classes	Bare land	Desert rangeland	Fertile cultivated land	Green pasture	Sutlej basin river land
Bare land	98	2	0	0	0
Desert rangeland	1	96	2	0	1
Fertile cultivated land	0	1	96	2	1
Green pasture	0	1	2	97	0
Sutlej basin river land	1	2	1	0	96

Table 7: Multispectral data classification table.

Classifiers	Kappa statistics	TP rate	FP rate	ROC	MAE	RMSE	Time (sec)	OA
Multilayer perceptron	0.9743	0.964	0.009	0.997	0.0240	0.0904	0.45	97.60%
Random Forest	0.9340	0.950	0.014	0.992	0.0395	0.1332	0.13	94.70%
J48	0.9170	0.934	0.020	0.965	0.0296	0.1610	0.03	93.30%
Naïve Bayes	0.7710	0.818	0.045	0.962	0.0730	0.2550	0.02	81.75%

Table 8: Confusion table of multispectral data for multilayer perceptron (MLP).

Classes	Bare land	Desert rangeland	Fertile cultivated land	Green pasture	Sutlej basin river Land
Bare land	97	2	0	0	1
Desert rangeland	1	96	2	0	1
Fertile cultivated land	0	0	98	1	1
Green pasture	0	0	2	98	0
Sutlej basin river land	1	0	1	0	98

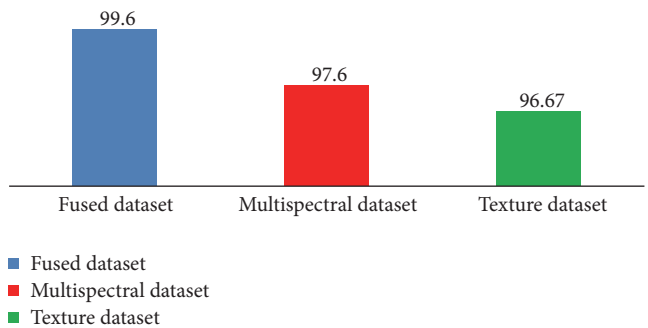


Figure 3: A comparison classification graph of fused, multispectral, and texture dataset.

among all types of dataset. It has been observed that final classification results of three datasets are not differing too much but other performance evaluating factors, that is, kappa statistics, RMSE, TP, FP, MAE, and execution time also play an important role for analysis. It is worth mentioning here that photographic data (texture data) is the visual data and its visual frequency ranges from 400 nm to 700 nm which has classification accuracy of 96.67% while multispectral data include the visual plus nonvisual data (IR and SWIR) and nonvisual frequency ranges from 750 nm to 1650 nm and attained classification accuracy of 97.60%, while fused dataset which integrated both types of data, that is, multispectral and statistical texture, acquired better overall accuracy which is 99.60% as compared to multispectral and texture dataset. Finally, it is observed that as dataset features values have been increased, the overall accuracy results have also been observed better and this shows that multisource data integration significantly improves the analysis and classification of LU/LC types and the employed classification framework is a powerful tool to generate reliable, comprehensive, and accurate results for LU/LC classification. In addition, it has been observed that this method can be used for decision-making, future prediction, and quick and accurate analysis of land use and land cover, when employing sophisticated rules on multisource LU/LC datasets. In future, the effect of variation in light intensity with incident light angle will be verified.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

The main concept and experimental work were performed by Salman Qadri and Dr. Dost Muhammad Khan made critical revisions and approved the final version. All other coauthors reviewed and approved the final paper.

Acknowledgments

The authors highly acknowledge the Department of Computer Science & IT, Islamia University of Bahawalpur, Pakistan, for providing all experimental facilities and convenient environment to accomplish this study and they especially thank Mr. Muzammil ur Rehman, Lecturer at DCS & IT, Islamia University of Bahawalpur, Pakistan, and Mr. Dell Nantt, CROPSCAN Corporation, Minnesota, USA, for their technical support.

References

[1] B. L. Turner, E. F. Lambin, and A. Reenberg, "The emergence of land change science for global environmental change and sustainability," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 52, pp. 20666–20671, 2007.

[2] S. Qadri, D. M. Khan, F. Ahmad et al., "A Comparative study of land cover classification by using multispectral and texture data," *BioMed Research International*, vol. 2016, Article ID 8797438, 2016.

[3] M. Simard, G. De Grandi, S. Saatchi, and P. Mayaux, "Mapping tropical coastal vegetation using JERS-1 and ERS-1 radar data with a decision tree classifier," *International Journal of Remote Sensing*, vol. 23, no. 7, pp. 1461–1474, 2002.

[4] L. Busetto, M. Meroni, and R. Colombo, "Combining medium and coarse spatial resolution satellite data to improve the estimation of sub-pixel NDVI time series," *Remote Sensing of Environment*, vol. 112, no. 1, pp. 118–131, 2008.

[5] D. Liu and R. Pu, "Downscaling thermal infrared radiance for subpixel land surface temperature retrieval," *Sensors*, vol. 8, no. 4, pp. 2695–2706, 2008.

[6] F. Gao, J. Masek, M. Schwaller, and F. Hall, "On the blending of the landsat and MODIS surface reflectance: predicting daily landsat surface reflectance," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 8, pp. 2207–2218, 2006.

[7] G. Cheng, L. Guo, T. Zhao, J. Han, H. Li, and J. Fang, "Automatic landslide detection from remote-sensing imagery using a scene classification method based on boVW and pLSA," *International Journal of Remote Sensing*, vol. 34, no. 1, pp. 45–59, 2013.

[8] L. C. Chen, T.-A. Teo, Y.-C. Shao, Y.-C. Lai, and J.-Y. Rau, "Fusion of LIDAR data and optical imagery for building modeling," in *Proceedings of the International Archives of Photogrammetry and Remote Sensing*, vol. 35, pp. 732–737, 2004.

[9] A. K. Saraf, "IRS-1C-LISS-III and PAN data fusion: an approach to improve remote sensing based mapping techniques," *International Journal of Remote Sensing*, vol. 20, no. 10, pp. 1929–1934, 1999.

[10] G. Simone, A. Farina, F. C. Morabito, S. B. Serpico, and L. Bruzzone, "Image fusion techniques for remote sensing applications," *Information Fusion*, vol. 3, no. 1, pp. 3–15, 2002.

[11] X. Huang and L. Zhang, "An SVM ensemble approach combining spectral, structural, and semantic features for the classification of high-resolution remotely sensed imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 1, pp. 257–272, 2013.

[12] L. I. Kuncheva, "Switching between selection and fusion in combining classifiers: An experiment," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 32, no. 2, pp. 146–156, 2002.

[13] T. Tu, S. Su, H. Shyu, and P. S. Huang, "A new look at IHS-like image fusion methods," *Information Fusion*, vol. 2, no. 3, pp. 177–186, 2001.

[14] K. Yao, B. Pradhan, and M. O. Idrees, "Identification of rocks and their quartz content in gua musang goldfield using advanced spaceborne thermal emission and reflection radiometer imagery," *Journal of Sensors*, vol. 2017, Article ID 6794095, 8 pages, 2017.

[15] M. S. Shifa, M. S. Naweed, M. Omar, M. Zeeshan Jhandir, and T. Ahmed, "Classification of cotton and sugarcane plants on the basis of their spectral behavior," *Pakistan Journal of Botany*, vol. 43, no. 4, pp. 2119–2125, 2011.

[16] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[17] P. M. Szczypiński, M. Strzelecki, A. Materka, and A. Klepaczko, "MaZda—a software package for image texture analysis," *Computer Methods and Programs in Biomedicine*, vol. 94, no. 1, pp. 66–76, 2009.

[18] S. C. Park, J. Pu, and B. Zheng, "Improving performance of computer-aided detection scheme by combining results from two machine learning classifiers," *Academic Radiology*, vol. 16, no. 3, pp. 266–274, 2009.

[19] S. A. M. Rodrigues, *Motivations, Experiences And Potential Impacts of Visitors to a Monastery in New Zealand: A Case Study*, University of Waikato, 2012.

[20] P. Zapotoczny, "Discrimination of wheat grain varieties using image analysis and neural networks. Part I. Single kernel texture," *Journal of Cereal Science*, vol. 54, no. 1, pp. 60–68, 2011.

[21] R. G. Congalton and K. Green, *Assessing the accuracy of remotely sensed data: principles and practices*, CRC press, 2008.

[22] G. M. Foody, "Classification accuracy comparison: hypothesis tests and the use of confidence intervals in evaluations of difference, equivalence and non-inferiority," *Remote Sensing of Environment*, vol. 113, no. 8, pp. 1658–1663, 2009.

[23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[24] N. Alajlan, Y. Bazi, F. Melgani, and R. R. Yager, "Fusion of supervised and unsupervised learning for improved classification of hyperspectral images," *Information Sciences*, vol. 217, pp. 39–55, 2012.

[25] C. Gómez, J. C. White, and M. A. Wulder, "Optical remotely sensed time series data for land cover classification: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 116, pp. 55–72, 2016.

[26] Z. Malenovský, H. Rott, J. Cihlar et al., "Sentinels for science: Potential of Sentinel-1, -2, and -3 missions for scientific observations of ocean, cryosphere, and land," *Remote Sensing of Environment*, vol. 120, pp. 91–101, 2012.

Hindawi
Journal of Sensors
Volume 2017, Article ID 6742615, 12 pages
https://doi.org/10.1155/2017/6742615

Research Article

An Efficient Calibration Method for a Stereo Camera System with Heterogeneous Lenses Using an Embedded Checkerboard Pattern

Pathum Rathnayaka, Seung-Hae Baek, and Soon-Yong Park

School of Computer Science and Engineering, Kyungpook National University, Daegu, Republic of Korea

Correspondence should be addressed to Soon-Yong Park; sypark@knu.ac.kr

Received 28 April 2017; Revised 13 July 2017; Accepted 20 July 2017; Published 5 September 2017

Academic Editor: Oleg Sergiyenko

We present two simple approaches to calibrate a stereo camera setup with heterogeneous lenses: a wide-angle fish-eye lens and a narrow-angle lens in left and right sides, respectively. Instead of using a conventional black-white checkerboard pattern, we design an embedded checkerboard pattern by combining two differently colored patterns. In both approaches, we split the captured stereo images into RGB channels and extract R and inverted G channels from left and right camera images, respectively. In our first approach, we consider the checkerboard pattern as the world coordinate system and calculate left and right transformation matrices corresponding to it. We use these two transformation matrices to estimate the relative pose of the right camera by multiplying the inversed left transformation with the right. In the second approach, we calculate a planar homography transformation to identify common object points in left-right image pairs and treat them with the well-known Zhangs camera calibration method. We analyze the robustness of these two approaches by comparing reprojection errors and image rectification results. Experimental results show that the second method is more accurate than the first one.

1. Introduction

The process of estimating internal-external (also known as intrinsic and extrinsic) camera parameters and knowing the correct relative pose between cameras in a stereo setup has been of the interest in the computer vision field for many years. It is considered as the first and foremost important step in many 2D/3D stereo vision experiments. Much related work have been introduced throughout the past few decades, initially starting in the photogrammetry community [1, 2]. As mentioned in [3], these calibration methods can be divided into two broad categories: photogrammetric calibration and self-calibration. Photogrammetric calibration is performed by observing a calibration object (normally a checkerboard pattern) whose geometry in the 3D space is known for the best precision. In contrary, self-calibration is performed by extracting feature points and processing correspondences between captured images of a static scene. However, one of the constraints in most of these photogrammetric calibration methods is using common or similar field-of-view (FOV) cameras. Correspondingly, many self-calibration methods also follow the same constraint, where a few utilize advantages of using heterogeneous setups. However, extracting rich key points is challenging and sometimes could lead into erroneous approximations. In this paper, we propose two new, yet simplified, calibration approaches for a heterogeneous camera setup. Instead of using the general black-white checkerboard pattern, we design a new color checkerboard pattern, by combining two different patterns. In our first approach, we consider the checkerboard pattern as the world coordinate system and calculate the two transformation relationships between left and right cameras correspondingly. Multiplying the inverted left transformation with the right transformation gives the relative pose of the right camera with respect to the left camera. In our second approach, we use a planar homography transformation method to identify common object points in stereo images. Once these common points are estimated, we apply Zhang's method [3] to calibrate the stereo camera setup. The remainder of this paper is constructed as follows: Section 2 describes some existing stereo calibration methods for heterogeneous

setups. Section 3 describes the preliminaries, including the configuration of our camera setup, the method of designing the color checkerboard pattern, and the method of separating two patterns from each other. Section 4 consists of the core of this paper, brief introductions to two stereo calibration approaches. Section 4.1 describes mono calibration method used to undistort input image sequences. In Section 4.2 we describe the matrix multiplication method and in Section 4.3 the planar homography transformation-based calibration method. Experiments performed to evaluate the accuracy of these two methods are summarized in Section 5. Besides comparing reprojection errors, we perform image rectifications to see how robust our proposed methods are. Finally, the conclusions and further discussions are drawn in Section 6.

2. Related Work

The popularity of wide-angle lenses, such as fish-eye cameras, has started to increase in the field of stereo vision. The wider FOV of such cameras allows users to cover a broad scene area compared to conventional cameras. These cameras have been intensively used in many recent stereo-based experiments, where quite a number of calibration methods have also been tested. Barreto and Daniilidis introduced a factorization approach without performing nonlinear minimization to estimate the relative pose between a conjugated wide-angle camera setup [4, 5] using a minimum of 15 corresponding point matches. Fischler and Bolles proposed a RANdom Sample Consensus (RANSAC) [6] based polynomial eigenvalue method [7] to estimate the relative pose of a noncentral catadioptric camera system [8]. Lhuillier introduced a similar approach [9] in 2008. In this method, he discussed applying a central model to estimate the geometry of the camera and a decoupling orientation translation to identify the transformation relationship. Lim et al. introduced a new stereo calibration method using an antipodal epipolar constraint [10]. In addition, many optical flow estimation approaches have been adopted for pose estimations, as cited in [11]. On the other hand, planar projection (or homography) based approaches have also been studied to estimate relative pose in a stereo camera rig. Chen et al. proposed a calibration method for a high definition stereo camera rig by utilizing the idea of homography transformation [12] using a marker chessboard. In year 2013, they discussed another slightly improved image undistortion and pose estimation method in their technical paper [13].

Even though these existing methods can be used to calibrate heterogeneous stereo camera setups, most of them have certain limitations and drawbacks. Most of these methods depend on geometric invariants of image features, such as projections of straight lines, or the approximations of the fundamental matrix [13]. They require proper extraction/matching of point correspondences between stereo image pairs, which sometimes could be more challenging due to irregular resolutions, different FOVs, and lens distortions of cameras. In addition, the implementation of these methods is limited only for small displacement since the reliability of feature points extraction decreases when there are large FOV differences between images. The method proposed by Barreto and Daniilidis is mostly algebraic, and the linear model requires a minimum of 15 point correspondences. Precise estimation of these correspondences is more ambiguous and less accurate in difficult environments. Similarly, the method proposed by Micusik and Pajdla generalizes Fitzgibbon's technique [14] and requires 9 point correspondences, whereas the method proposed by Lhuillier requires a minimum of 7 point correspondences to calculate the fundamental matrix. The method introduced by Lim et al. imposes the constraints on the distribution of feature points. The planar homography method introduced by Chen et al. in their first research article [12] sometimes failed to detect chessboard corners properly. They proposed a solution for this problem in their second research article [13] by introducing the concept of a robust type homography transformation in which they primarily focused on processing mono video cameras instead of focusing on stereo systems.

In our article, we realized that the above limitations and drawbacks occur mainly because of using point correspondences in-between stereo image pairs. However, the two stereo calibration methods we state in this article do not depend on these sensitive point correspondences and do not show such difficulties. Instead, we use pure mathematical approaches for pose estimations. The embedded checkerboard pattern we introduce is a proper alternative for the traditional black-white checkerboard pattern and can be used in cases where common areas are not visible in images (due to FOV differences).

3. Preliminaries

3.1. Focal Lengths, Field-of-Views, and Wide-Angle Cameras. Focal length is the distance from the center of the lens to the image plane where light converges to a similar point named the focal point. Figure 1 shows how two light rays are converging into this point. The focal length of a camera and its FOV are proportionally interconnected with each other. A longer focal length results in a lower FOV, where a lower focal length results in a higher FOV. This proportional relationship allows for converging or diverging the amount of light entering the camera. This is graphically shown in Figure 2. Using a short focal length is the base idea of wide-angle lenses [15, 16].

The popularity of wide-angle lenses, such as fish-eye lenses, have started to increase because of their ability to cover wider viewing areas. The basement of these wide-angle lenses can be considered as the Double-Gauss lens [17], which is a compound-type lens of a positive and negative meniscus lenses on the object side and the image side, respectively. In general, all these wide-angle lenses can be categorized into two main groups: short focus lenses and retrofocus lenses. Short focus lenses are generally made of multiple glass elements whose shapes are nearly symmetrical in the front and back of the diaphragm. Retrofocus lenses use an inverted telephoto configuration, in which the front element is negative.

3.2. Designing the Special Checkerboard Pattern. The conventional way of capturing stereo images of a black-white

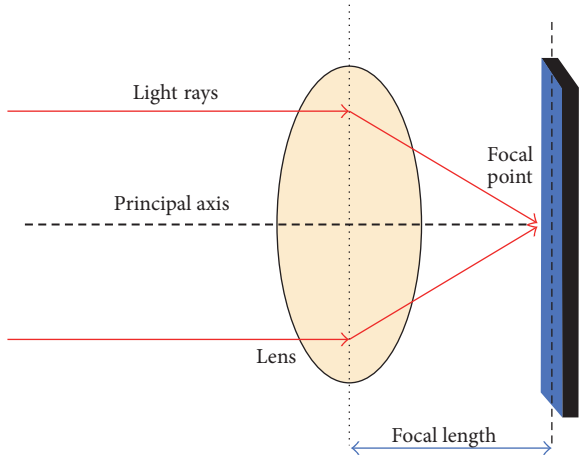


Figure 1: The concept of focal length. The rays are converged into the focal point.

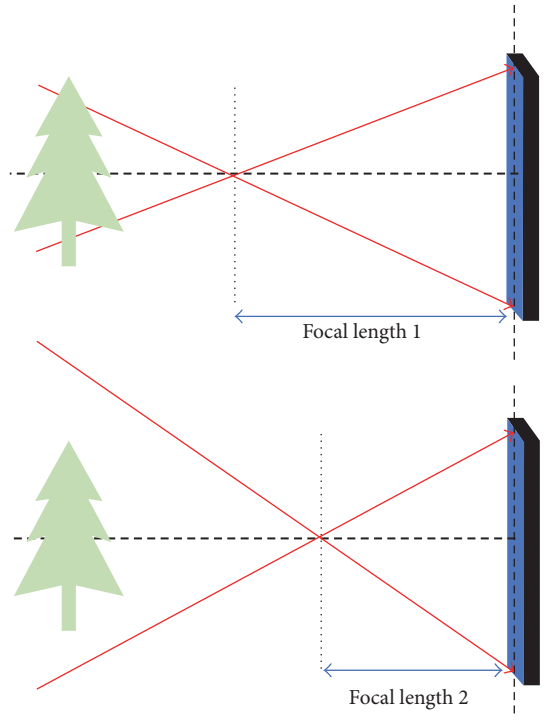


Figure 2: The proportional relationship between the focal length and FOV. When the focal length is longer, the FOV is lower, resulting in only a part of the object light rays to be converged. When the focal length is shorter, the FOV becomes higher, resulting in a wider area of the object to be converged.

checkerboard pattern using narrow-angle cameras has constantly been used in many existing methods. To obtain higher accurate calibration results, the pattern needs to be kept near to cameras. This orientation could sometimes result in limiting the number of poses (even though the minimum number of poses required is six as mentioned in [3]). In some situations, capturing the full area of the checkerboard pattern fails. One possible solution to resolve this occlusion problem would be using wide-angle lenses. In this paper, we have

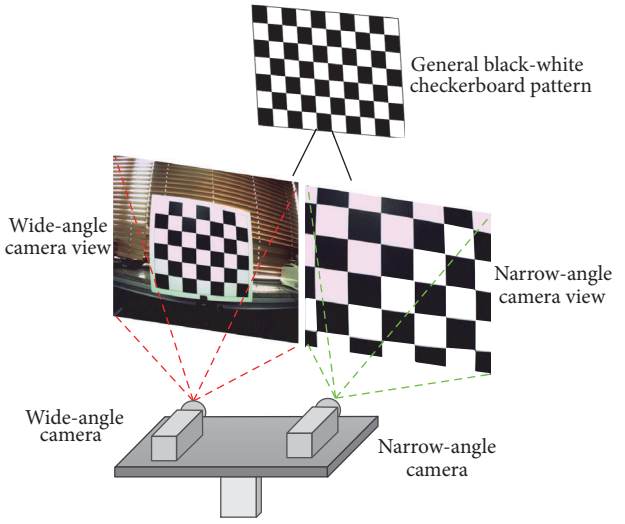


Figure 3: An example showing how the viewing angles are different in wide-angle and narrow-angle cameras. Left side consists of the wide-angle camera, and it covers a larger area of the scene. Right side consists of the narrow-angle camera, and it only covers a smaller part of the scene. The general checkerboard pattern is partially seen by the narrow-angle camera.

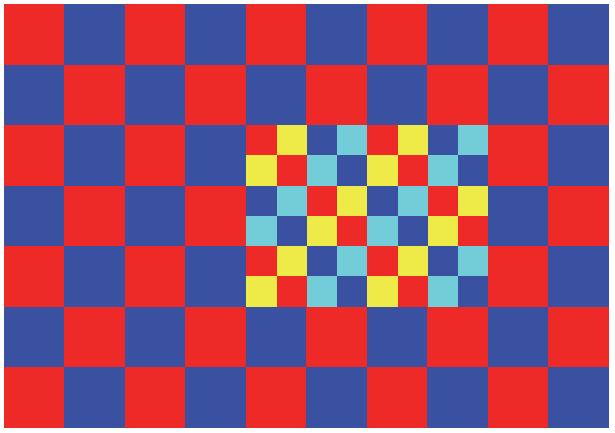


Figure 4: The special colored checkerboard pattern used instead of the conventional black-white checkerboard pattern. The pattern is a combination of two differently colored checkerboard patterns. The outer pattern consists of red-blue checker patterns and the small inner pattern with red-yellow-blue-cyan color patterns. Aspect ratio between two patterns is 2 : 1.

decided to use a single wide-angle lens along with a narrow-angle lens.

However, using a wide-angle lens does not guarantee the stereo setup manages to capture full images of the checkerboard pattern. Since we use a narrow-angle camera in our stereo setup, there is a difficulty to cover the full area of the checkerboard pattern at close distance as it is illustrated in Figure 3. In order to overcome these problems and as a final solution, we have designed a new checkerboard pattern and used it instead of using the conventional black-white pattern. This new checkerboard pattern we used in our proposed methods is graphically shown in Figure 4.

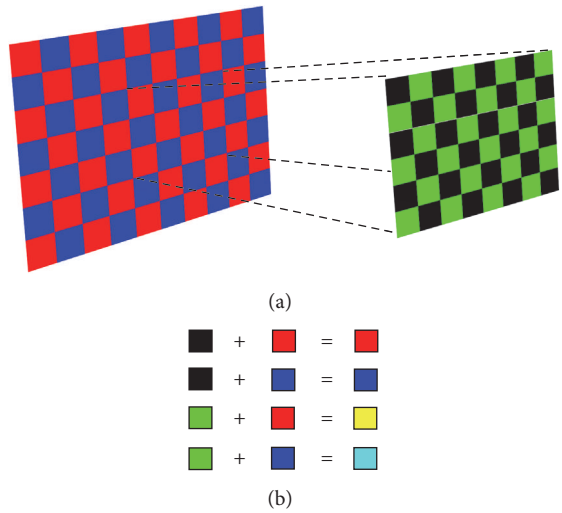


Figure 5: The method of generating the special color checker pattern. (a) The 7×10 red-blue outer pattern is mixed with the 6×8 black-green smaller pattern. (b) The basic colors blend and result in red, yellow, blue, and cyan colors. This mixing results in the inner pattern as shown in Figure 4.

This special checkerboard pattern is made by combining two different color checkerboard patterns: 7×10 larger pattern and a 6×8 smaller pattern. The larger pattern (from now on mentioned as the outer pattern) is designed by red-blue checker patterns, and the smaller pattern is designed by black-green checker patterns. This smaller pattern is embedded into the outer pattern (as in Figure 5(a)), making the basic color blend. Color mixing results in a secondary inner pattern with red-yellow, blue, and cyan colors inside the outer pattern. Therefore, we can think of using two individual checkerboard patterns, instead of using a single pattern. The process of designing this special checkerboard pattern is depicted in Figure 5.

3.3. Capturing Calibration Images of Special Checkerboard Pattern. The heterogeneous stereo camera setup we have used in our experiments is depicted in Figure 6. Two Point Grey Grasshopper cameras are mounted on either side of a horizontal panning bar: left side wide-angle camera (focal length $\cong 3.5$ mm) and right side narrow-angle camera (focal length $\cong 8$ mm). We kept the special checkerboard pattern in front of the cameras in such a way the narrow-angle camera always sees the full area of the inner checkerboard pattern. Since the wide-angle camera has a wider FOV, it fully sees both inner and outer patterns (Figure 7).

In our experiments, we wanted to retain only the outer pattern from wide-angle camera images and the inner pattern from narrow-angle camera images. We performed RGB channel splitting to distinguish two patterns from each other. Once R channel is extracted, we managed to separately identify the outer pattern in wide-angle camera images. Similarly, we first extracted the G channel from narrow-angle camera images and inverted it to identify the inner pattern. Figure 8 shows an instance of how we managed to separately identify two patterns from each other. Figures 8(a) and 8(b) show left wide-angle and right narrow-angle camera images. We can easily identify the outer pattern from the wide-angle image by extracting R channel and the inner pattern from the narrow-angle image by extracting the inverted G channel.

4. Stereo Calibration

4.1. Mono Camera Calibration. One of the problems of using wide-angle cameras is that they suffer from massive barrel distortions. Performing stereo calibrations without correcting distortions could lead into erroneous matrix calculations. Consequently, we start our two stereo calibration methods by first undistorting input wide and narrow-angle images.

We use the same experiment setup mentioned in Section 3.3. We kept the special checkerboard pattern at a short distance and captured left-right wide and narrow-angle camera images separately. After capturing images, we followed the method mentioned in Section 3.3 to retain the outer pattern in wide-angle camera images and the inner pattern in narrow-angle camera images. We then used the well-known Zhang's method [3] to calibrate cameras independently. Figure 9 shows an instance of where wide- and narrow-angle cameras are calibrated separately.

4.2. Stereo Calibration Using Transformation Matrices. The first stereo calibration approach is based on multiplying the two transformation matrices between wide- and narrow-angle cameras. Once two cameras are properly calibrated as mentioned in Section 4.1, we then capture stereo image sequences of the checkerboard pattern from two cameras at the same time. While capturing images, we kept the checkerboard pattern at a short distance to the cameras in such a way the wide-angle camera sees the full area of the pattern and the narrow-angle camera sees the full area of the inner pattern. In this method, we considered the checkerboard pattern as the world coordinate system where the origin lies at the intersection point of first red and blue checker patterns of the outer pattern. Since we consider inner and outer patterns are two different checkerboards, the inner

Figure 6: The stereo camera setup we used in our experiments. Left side consists of the wide-angle camera and the right side consists of the narrow-angle camera. Both cameras are mounted on a horizontal panning bar. The special checkerboard pattern is kept very near to the camera setup.

pattern has its origin at the intersection point of first red and yellow checkers, and we shifted this toward the origin of the outer pattern by simply adding the distance between two origins. This is graphically described in Figure 10.

Taking T_{WL} and T_{WR} representing two transformation matrices between wide-angle and narrow-angle cameras with respect to world coordinate system, we wanted to find the relative pose of the narrow-angle camera with respect to wide-angle camera, T_{LR}. We used the captured stereo image sequences to calibrate two cameras separately (Section 4.1) and estimated two 3×4 camera matrices (or perspective projection matrices).

The general relationship between a 3D point P_{world} in the world coordinate system and its respective 2D point P_{image} in

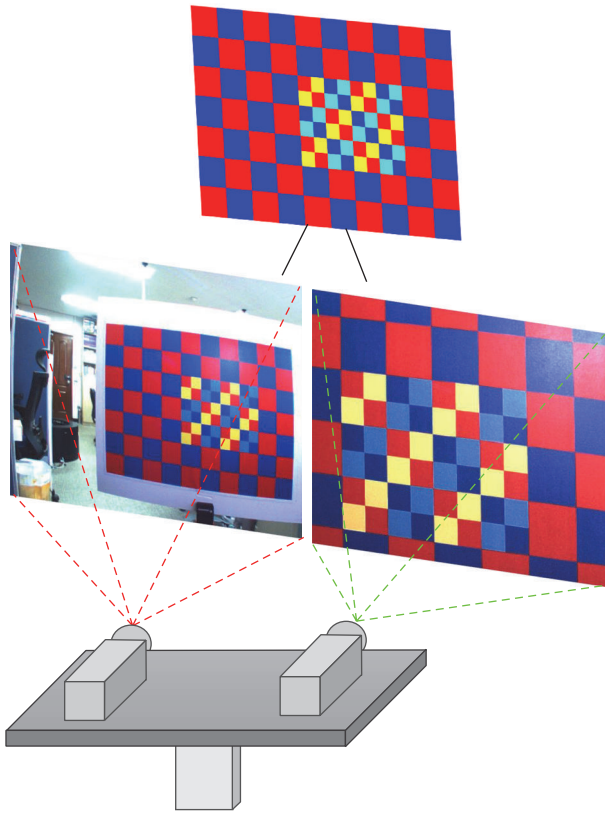
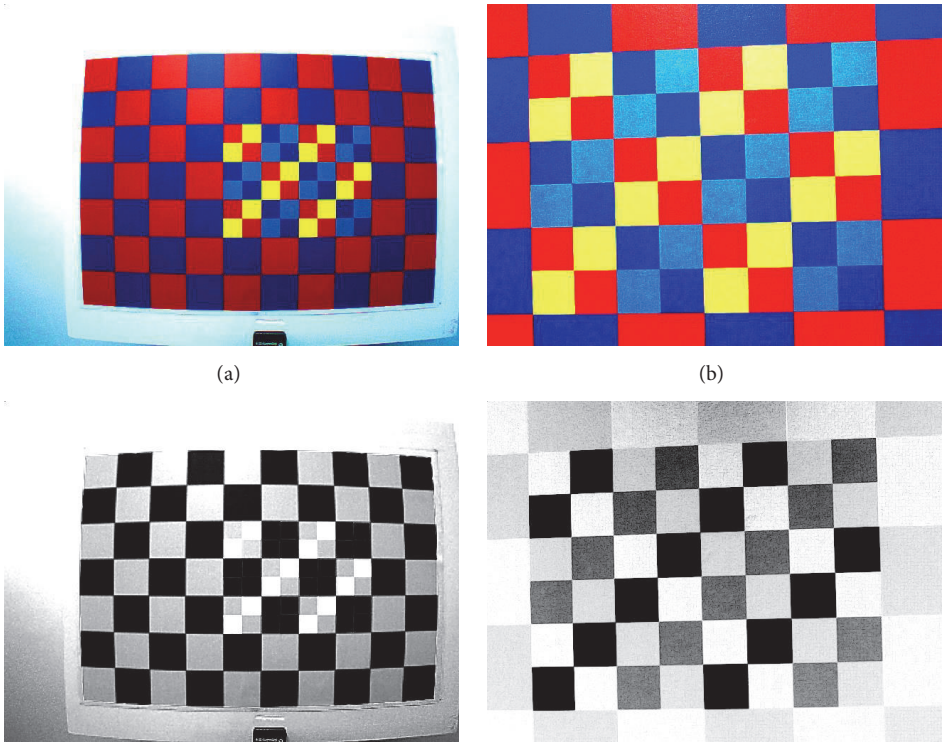


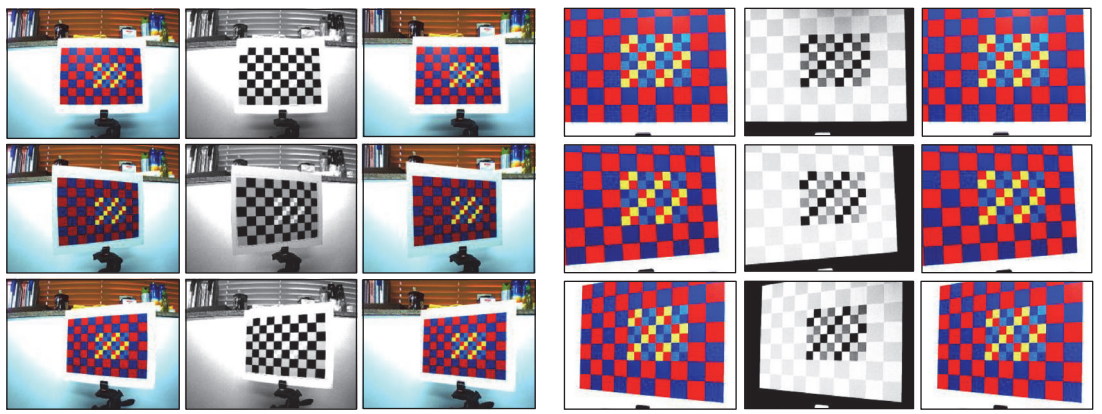
Figure 7: A representation of how the special checkerboard pattern is seen in left wide-angle and right narrow-angle cameras. Checkerboard pattern is kept at a very close distance to the cameras. Wide-angle camera has a higher FOV; thus it sees the whole area of the checker pattern. The FOV of narrow-angle camera is lower; thus it always sees the inner pattern.



(a) (b)

(c) (d)

Figure 8: The method of separately identifying outer and inner patterns from wide-angle and narrow-angle cameras, respectively. Original wide-angle and narrow-angle camera images are shown in (a) and (b). (c) states the extracted R channel of wide-angle camera image, where (d) states the inverted G channel of narrow-angle camera image.



(a) (b)

FIGURE 9: An instance of mono calibrations. First, second and third columns represent original, split channel, and undistorted images. (a) R channel is extracted from wide-angle camera images for calibration. (b) Inverted G channel is extracted from narrow-angle camera images for calibration.

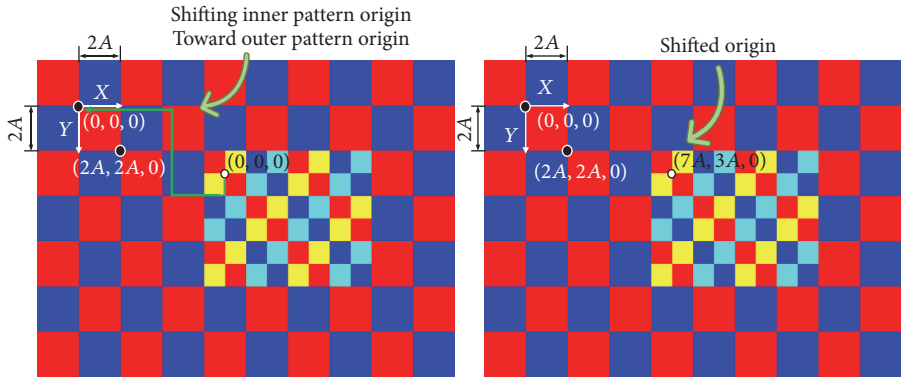


FIGURE 10: Shifting the origin of the inner pattern toward the outer pattern origin.

the image coordinate system can be written as

$$P_{\text{image}} = MP_{\text{world}}, \tag{1}$$

where M depicts the camera matrix. This M matrix can be further decomposed as intrinsic camera matrix K and rigid transformation matrix (or the extrinsic matrix) $[R, t]$ [18]. Thus, (1) can be rewritten as

$$P_{\text{image}} = K\,[R \mid t]\,P_{\text{world}}, \tag{2}$$

where R denotes 3×3 rotation and t denotes 3×1 translation (r_{ij} and t_i in (3), resp.).

$$[R \mid t] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}. \tag{3}$$

These intrinsic and extrinsic entries of M matrix can be easily identified using RQ factorization [19].

We can estimate both T_{WL} and T_{WR} transformation matrices by applying this generalization into wide-angle and narrow-angle cameras separately as follows:

$$\begin{aligned} T_{WL} &= [R \mid t]_{WL}, \\ T_{LR} &= [R \mid t]_{WR}. \end{aligned} \tag{4}$$

The following equation depicts the relationship between transformation matrices shown in Figure 11, which we are interested in estimating T_{LR}.

$$T_{WL} \cdot T_{LR} = T_{WR}. \tag{5}$$

We multiplied the inverse of left transformation matrix with the right transformation matrix to find the relative pose of the narrow-angle camera with respect to wide-angle camera as follows:

$$T_{LR} = {T_{WL}}^{-1} \times T_{WR}. \tag{6}$$

Figure 12 graphically summarizes the whole matrix multiplication-based calibration procedure as a flow chart.

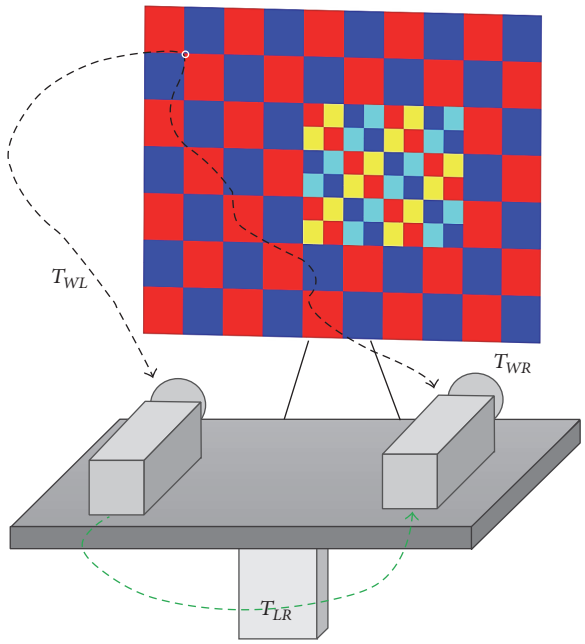


Figure 11: Calculating the relative pose between wide and narrow-angle cameras using two transformation matrices obtained with respect to the world coordinate system.

4.3. Stereo Calibration Using Planar Homography Transformation. Figure 13 summarizes the whole process we followed to find the relative pose of the narrow-angle camera by keeping wide-angle camera as the reference. Similar to the method mentioned in Section 4.2, we first undistorted the images and used them as input data.

This second approach uses Zhang's method to perform stereo calibration, but, to apply Zhang's method, we need to know the correct relationship between point locations in two camera images. Due to the reason that the narrow-angle camera only sees a partial area of the full checkerboard pattern, we could not directly identify this relationship. Therefore, we applied a planar homography transformation on the wide-angle camera images to properly project point locations into the view point of the narrow-angle camera.

To calculate the planar homography matrix H, we need at least four corresponding image points between wide- and narrow-angle images. This means that we need to know at least four sets of 2D image coordinates of the checkerboard pattern. Due to the FOVs of two cameras, wide-angle camera captures both inner and outer patterns, where narrow-angle camera only manages to capture the full area of the inner pattern (with some partial areas of the outer pattern).

Therefore, we decided to retain only the inner pattern in both wide- and narrow-angle images. We followed the same channel splitting method, but this time we only considered extracting the inverted G channel. This results in separately identifying the inner pattern in both camera images. We manually selected four exact common point locations in images to calculate matrix H as shown in Figure 14. According to [18], the homography transformation relationship between two 2D corresponding point locations can be summarized as

$$x' = Hx. \tag{7}$$

H is the 3×3 homography transformation matrix that we are interested in calculating, where x' and x represent known 2D point locations we selected in wide-angle and narrow-angle camera images, respectively. Using the above four point correspondences, we find this H matrix based on singular value decomposition.

After calculating H matrix, we next find chessboard corners of the outer pattern in wide-angle images. We followed the steps mentioned in [20] to find the chess corner locations accurately. We first extract R channel to retain the outer pattern and find 2D point information of all 54 corners. Next we apply H matrix to identify where these corner points projected onto narrow-angle images (Figure 15). Green circles in narrow-angle images represent these projected point locations. We adjusted these points with subpixel accuracy to maximize their cornerness criteria. Once we find 2D coordinates of common object points in both wide- and narrow-angle images, we can treat them with Zhang's method to perform stereo calibration between two cameras.

Table 1: Intrinsic camera parameters.

	Left camera	Right camera
f_X	483.5165	1127.8372
f_Y	483.1779	1125.0019
c_x	343.4943	343.6781
c_Y	257.4602	256.7247
Reprojection Err.	0.3252	0.3500

5. Experiments and Results

We have performed 4 experiments (2 for method 1 and 2 for method 2) to evaluate the robustness of the proposed two methods. We have performed experiments in both indoor and outdoor environments. We have used the same experiment setup mentioned in Figure 6 to perform indoor experiments, where we mounted it on top of the front mirror of a vehicle to do outdoor experiments. We used a similar number of image sequences (30 images) in every experiment. Table 1 summarizes intrinsic camera parameters for both cameras. Parameters f_x and f_y represent the focal lengths expressed in pixel units in X and Y directions. c_x and c_y represent the X and Y components of the principal point. Table 2 summarizes experiment results calculated for both indoor and outdoor environments from method 1 in Section 4.2 and method 2 in Section 4.3. Parameters R_x, R_y, and R_z represent the components of rotation in X, Y, and Z directions, where parameters T_x, T_y, and T_z represent the components of translation in X, Y, and Z directions.

We calculated and compared reprojection error values in both methods, that is, the root mean squared value (RMS) of Euclidean distances between the observed chess corner

Table 2: Stereo calibration results for both methods in indoor and outdoor environments.

	Method 1		Method 2	
	Indoor	Outdoor	Indoor	Outdoor
R_x	−0.008254	−0.007284	−0.007255	−0.007284
R_y	0.015047	0.017057	0.009064	0.01705
R_z	−0.02171	−0.02341	−0.02015	−0.02341
T_x	−106.170024	−106.170057	−106.170006	−106.170057
T_y	4.125034	3.785034	3.525034	3.55035
T_z	−3.964857	−3.961871	−2.962087	−2.961808
Reprojection Err.	1.0701	0.8587	0.6781	0.5702

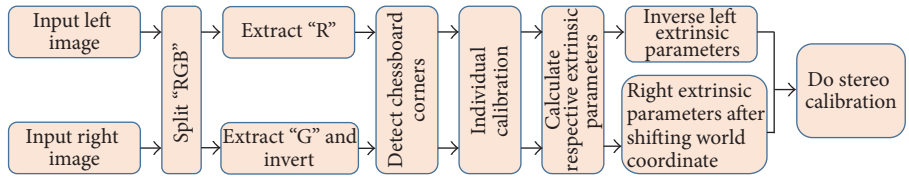


Figure 12: Stereo calibration method by multiplying left-right transformation matrices. Given input images are undistorted using undistortion coefficients calculated in mono camera calibration step.

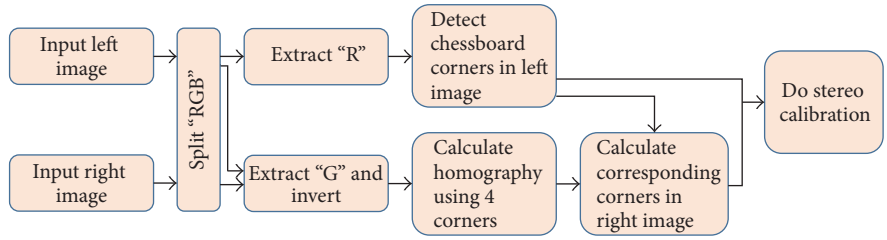


Figure 13: Stereo calibration method by finding a planar homography relationship between wide and narrow-angle camera images. Four points from the inner pattern in both camera images are used to calculate the homography transformation.

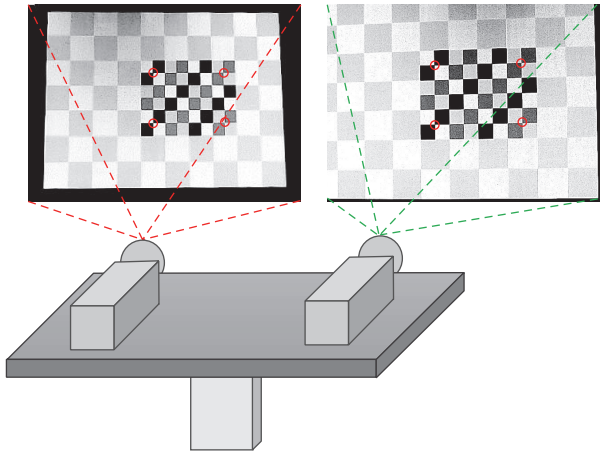


Figure 14: Finding planar homography transformation between wide- and narrow-angle camera images using the inner pattern.

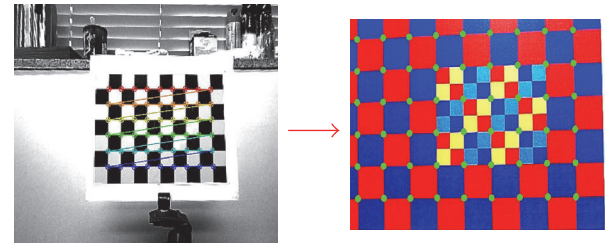


Figure 15: Applying homography to project 54 corner points of wide-angle image to narrow-angle image. Green points depict respective projected corner point.

points in the image coordinate system (in 2D calibration images) and the corresponding projected object points. We referred to [18, 21] to calculate these errors.

Also, we performed image rectifications [22] to see how accurate our calibration methods are. Experiment results affirm homography transformation method is slightly accurate compared to the matrix multiplication method. Some indoor environment rectification results generated from both methods are shown in Figures 16 and 17, where outdoor results are shown in Figures 18 and 19, respectively. There, we drew epilines (green horizontal lines) to represent the rectification error graphically and additionally calculated the absolute Y value differences of the inner pattern's chessboard corner locations to represent it mathematically.

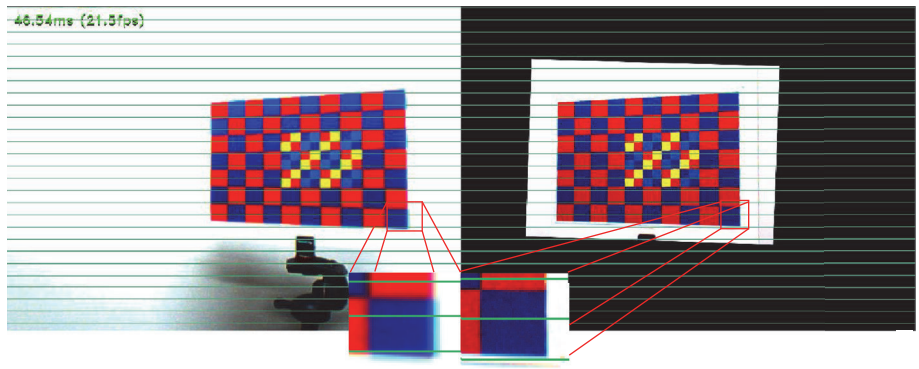


Figure 16: Rectified canvas result for indoor environment from method 1. Green horizontal lines represent rectified Y lines. The bottom edge has some rectification errors.

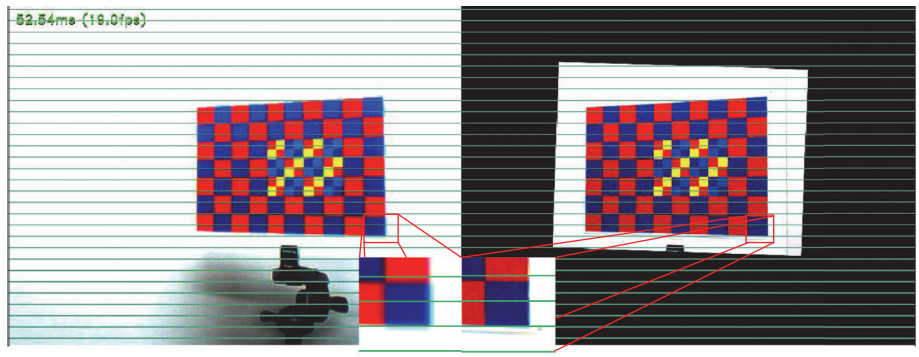


Figure 17: Rectified canvas result for indoor environment from method 2.

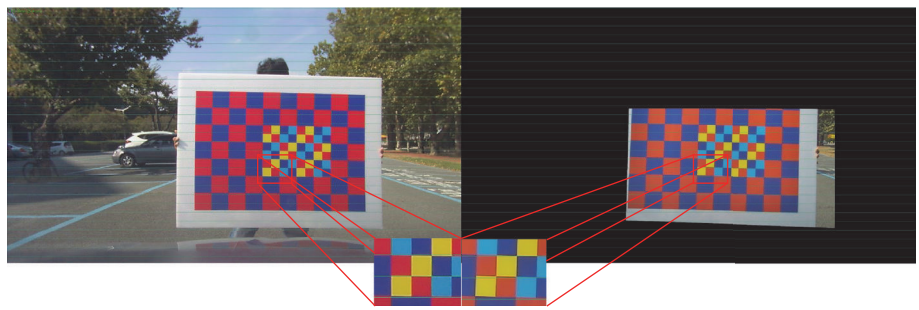


Figure 18: Rectified canvas result for outdoor environment from method 1. Some small rectification errors exist.

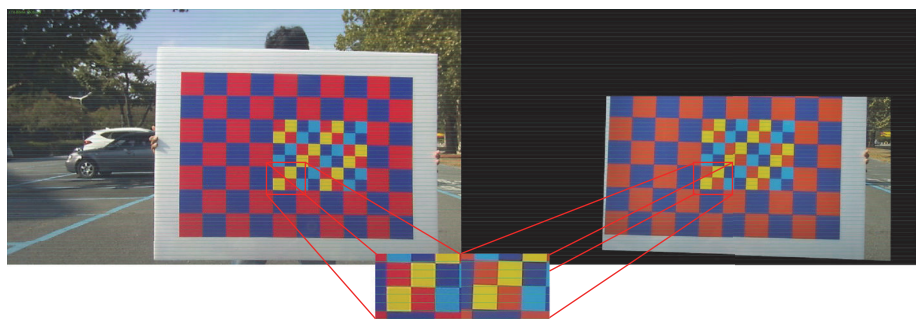
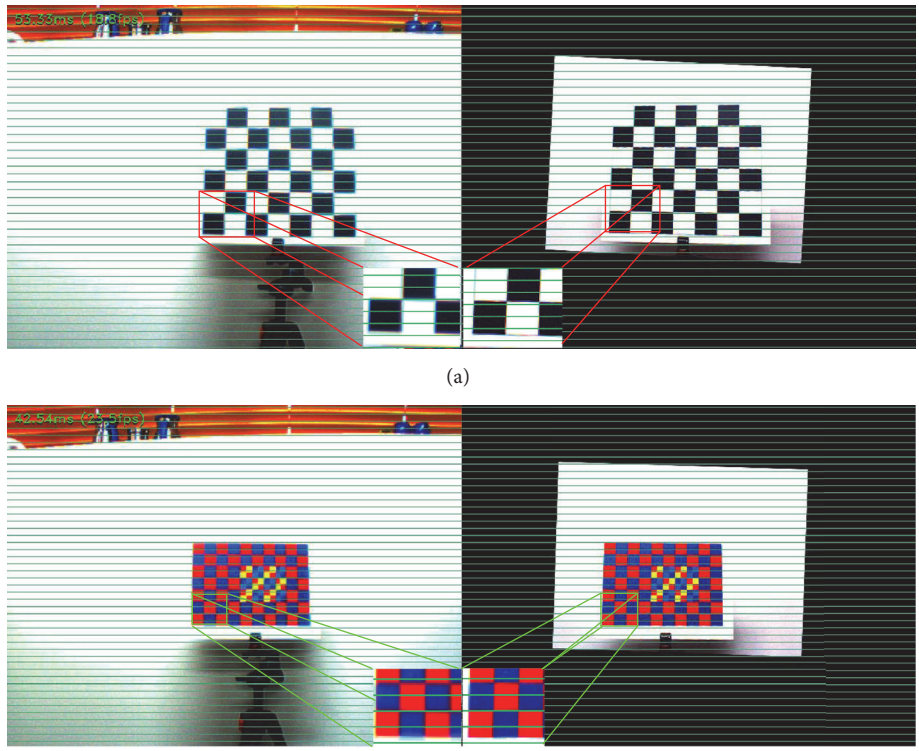


Figure 19: Rectified canvas result for outdoor environment from method 2.



(a)

(b)

Figure 20: Rectification results comparisons when using general pattern and the embedded pattern. (a) depicts the result for general pattern rectification and (b) depicts the result for colored checker pattern.

Table 3: Comparisons of rectification errors for 4 rectified stereo image pairs.

	Method 1 (pixels)	Method 2 (pixels)
Set 1	1.61847	1.49214
Set 2	1.74390	0.98472
Set 3	1.90601	1.32478
Set 4	2.03619	1.57652
Average Err.	1.82614	1.59189

To represent rectification error mathematically, we selected four stereo image pairs from the outdoor environment that are rectified using calibration parameters of the two methods. From each image set, we extracted inner pattern areas, estimated 35 chess corner locations (as mentioned in [20]), and calculated Y value differences (in pixels) between corresponding point locations in wide- and narrow-angle images. We summarized the average difference of each individual image set along with their overall average (term *Average Err.*). Table 3 depicts these results in pixels.

We performed another experiment to evaluate the accuracy of calibration using the embedded checkerboard pattern and a general black-white checkerboard pattern. We kept both patterns at the same position, where both cameras manage to see the full area. We calibrated the images of the black-white pattern according to the general version of Zhang's method. We used our proposed homography transformation method to calibrate the images of the embedded pattern. Similarly, we performed image rectifications and calculated 2D pixel positions to confirm that the combination of our embedded pattern and homography-based method gives better results compared to the general method when using the black-white pattern (Figure 20).

6. Conclusions

In this paper, we proposed two new methods to calibrate a heterogeneous stereo camera setup using a special colored checkerboard pattern. The heterogeneous camera setup consisted of a left wide-angle fish-eye lens camera and a right narrow-angle conventional camera. Because of the viewing angle irregularities, we could not use the conventional black-white checkerboard pattern at a short distance to the cameras. Therefore, we designed a new color checkerboard pattern by combining two different size checkerboard patterns. We embedded the small checkerboard pattern with the larger checkerboard pattern, letting their colors blend. This color blending results in a special checkerboard pattern, which consists of an outer pattern and an inner pattern. This checker pattern is kept at a very close distance to cameras and captured calibration images sequences to improve estimated results. We used RGB channel splitting method to separately identify two patterns from each other.

In our first method, we perform stereo calibration between the cameras by calculating left and right transformation matrices. In our second method, we calculated a

planar homography relationship between two cameras to identify common object point locations of stereo images. We projected chessboard corner locations of the outer pattern into the view point of the narrow-angle camera by treating them with the calculated homography relationship. Zhang's calibration method was applied to calibrate the stereo camera rig afterwards. We created rectification results to evaluate the robustness of our two proposed methods. There, we realized the second method was slightly accurate than the first.

As in future improvements, we are planning to parallelize both calibration approaches in GPU-based Nvidia Jetson TK1 board to speed up calibration by reducing the computation time and to use it in an embedded smart vehicle system for lane detection. In addition, we are planning to enhance the accuracy by updating calibration results using the well-known 5-point algorithm and a parallelized SIFT-GPU based corresponding point extraction.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was jointly supported by the Civil Military Technology Cooperation Center and the Korea Atomic Energy Research Institute (KAERI), the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIP) (no. NRF-2016M2A2A4A04913462), and the BK21-Plus project (SW Human Resource Development Program for Supporting Smart Life) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (21A20131600005).

References

[1] D. C. Brown, "Close-range camera calibration," *Photogrammetric Engineering*, vol. 37, no. 8, pp. 855–866, 1971.

[2] W. Faig, "Calibration of close-range photogrammetric systems: mathematical formulation," *Photogrammetric Engineering and Remote Sensing*, vol. 41, no. 12, pp. 1479–1486, 1975.

[3] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[4] J. Barreto and K. Daniilidis, "Wide area multiple camera calibration and estimation of radial distortion," in *Proceedings of the in Proceedings of the 5th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, vol. 63, p. 64, Prague, Czech Republic, 2004.

[5] J. P. Barreto and K. Daniilidis, "Fundamental matrix for cameras with radial distortion," in *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV '05)*, vol. 1, pp. 625–632, Beijing, China, October 2005.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the Association for Computing Machinery*, vol. 24, no. 6, pp. 381–395, 1981.

[7] B. Micusik and T. Pajdla, "Structure from motion with wide circular field of view cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1135–1149, 2006.

[8] S. Baker and S. K. Nayar, "Single Viewpoint Catadioptric Cameras," in *Panoramic Vision*, Monographs in Computer Science, pp. 39–71, Springer, New York, NY, USA, 2001.

[9] M. Lhuillier, "Automatic scene structure and camera motion using a catadioptric system," *Computer Vision and Image Understanding*, vol. 109, no. 2, pp. 186–203, 2008.

[10] J. Lim, N. Barnes, and H. Li, "Estimating relative camera motion from the antipodal-epipolar constraint," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1907–1914, 2010.

[11] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, pp. 5695–5701, October 2006.

[12] J. Chen, K. Benzeroual, and R. S. Allison, "Calibration for high-definition camera rigs with marker chessboard," in *Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '12)*, pp. 29–36, June 2012.

[13] J. Chen, K. Benzeroual, and R. S. Allison, "Robust homography for real-time image un-distortion," in *Proceedings of the 2013 International Conference on 3D Imaging (IC3D '13)*, pp. 1–8, December 2013.

[14] A. W. Fitzgibbon, "Simultaneous linear estimation of multiple view geometry and lens distortion," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, vol. 1, pp. I125–I132, December 2001.

[15] K. Suzuki, "Wide-angle lens," US Patent 6,016,229, 2000.

[16] W. Rotman and R. Turner, "Wide-angle microwave lens for line source applications," *IEEE Transactions on Antennas and Propagation*, vol. 11, no. 6, pp. 623–632, 1963.

[17] R. P. Jonas and M. D. Thorpe, "Double gauss lens design: a review of some classics," in *Proceedings of the Contract Proceedings 2006, International Society for Optics and Photonics*, Vancouver, Canada, 2006.

[18] R. Szeliski, *Computer Vision Algorithms and Applications*, Springer, London, UK, 2011.

[19] G. H. Golub and C. F. van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, Md, USA, 3rd edition, 1996.

[20] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision in C++ with the OpenCV Library*, O'Reilly Media, Inc., 2nd edition, 2013.

[21] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, New York, NY, USA, 2nd edition, 2003.

[22] D. V. Papadimitriou and T. J. Dennis, "Epipolar line estimation and rectification for stereo image pairs," *IEEE Transactions on Image Processing*, vol. 5, no. 4, pp. 672–676, 1996.

Hindawi
Journal of Sensors
Volume 2017, Article ID 3964376, 7 pages
https://doi.org/10.1155/2017/3964376

Research Article

Aphid Identification and Counting Based on Smartphone and Machine Vision

Suo Xuesong, Liu Zi, Sun Lei, Wang Jiao, and Zhao Yang

Mechanical and Electrical Engineering, Agricultural University of Hebei, Baoding, Hebei 071001, China

Correspondence should be addressed to Suo Xuesong; 13903120861@163.com

Received 27 April 2017; Revised 20 June 2017; Accepted 17 July 2017; Published 29 August 2017

Academic Editor: Oleg Sergiyenko

Exact enumeration of aphids before the aphids outbreak can provide basis for precision spray. This paper designs counting software that can be run on smartphones for real-time enumeration of aphids. As a first step of the method used in this paper, images of the yellow sticky board that is aiming to catch insects are segmented from complex background by using GrabCut method; then the images will be normalized by perspective transformation method. The second step is the pretreatment on the images; images of aphids will be segmented by using OSTU threshold method after the effect of random illumination is eliminated by single image difference method. The last step of the method is aphids' recognition and counting according to area feature of aphids after extracting contours of aphids by contour detection method. At last, the result of the experiment proves that the effect of random illumination can be effectively eliminated by using single image difference method. The counting accuracy in greenhouse is above 95%, while it can reach 92.5% outside. Thus, it can be seen that the counting software designed in this paper can realize exact enumeration of aphids under complicated illumination which can be used widely. The design method proposed in this paper can provide basis for precision spray according to its effective detection insects.

1. Introduction

As the main representative of crop pests affecting wheat field and orchard fruit output [1], aphids with a huge number are widely distributed. By hanging the yellow sticky board on plants, people can take count of the aphids on the board so that they can determine if pesticide is needed. However, this method is imprecise.

Compared with manual counting, machine vision can calculate the number of pests objectively, accurately, and effectively through analyzing the image information. To solve these problems, a lot of research has been carried out.

Cassani et al. approached automated EEG-based Alzheimer's disease diagnosis using relevance vector machines [2], Chakraborty and Newton studied climate change, plant diseases, and food security [3], Pal and Foody used evaluation of SVM, RVM, and SMLR for accurate image classification with limited ground data [4], Patel et al. used improved multiple features based algorithm for fruit detection [5], Luo et al. develop an aphid damage hyperspectral index for detecting aphid (Hemiptera: Aphididae) damage levels in winter wheat and so on [6].

Yue-hua et al. [7] extracted the aphids by using region growing algorithm and OSTU threshold algorithm in the natural environment, and the recognition rate of vector machine training images can reach 90.7% by using k-means; Bai-jing et al. [8] extracted the aphids on the use of the method of G component threshold (each pixel in the aphids and leaves) and carried out counting research on the use of connected area markers. Jingxiao [9] transmitted the image of the pest to the PC side for HSV color model segmentation and then the pest is identified by BP neural network. Vincent et al. [10] proposed a target detector that uses the analytical decision support system to make the detector adapt to changes in the plant to identify aphids on the plant; Tirelli et al. studied the remote monitoring system to prevent pest outbreaks and trigger an alarm when the number of pests is too high [11]; Ukatsu achieved the remote counting function of rice margin bugs through using the wireless network technology.

These methods reflect the following problems: they will be influenced by illumination and the threshold is not unique in threshold extraction method. Extracting aphids by PC remoting method costs too much network traffic with

poor timeliness. Therefore, to obtain images in the complex environment and high impact of the scene, pests can be identified and counted without relying on the feedback of the PC side.

2. Materials and Methods

The acquisition of experimental images was obtained at the West Campus of Agricultural University of Hebei; image acquisition equipment is Samsung Galaxy 4 phone that has a Qualcomm eight-core processor with Android 2.3 system, API 25. It is a common type with satisfactory parameters. The chapter contains two parts of picture distillation; in the first part, the GrabCut algorithm [12] is used to extract yellow sticky board image from complex background and correct it to the same size and position by perspective correction method in order to extract aphid image [13]; in the second part, the method of weighted average gray scale is used to make the yellow sticky board image preprocessing, single image difference method is used to eliminate the influence of illumination, and OSTU threshold segmentation and morphological erosion method is used to extract yellow sticky board image, laying the groundwork for the subsequent counting program.

2.1. Extraction the of Yellow Sticky Board Image. GrabCut is an improved algorithm based on GraphCut. GrabCut is an algorithm that uses the RGB three-channel hybrid Gauss model for color segmentation; segmentation can be completed by using region pixel set which contains background. The algorithm belongs to iterative segmentation, and the segmentation accuracy is higher. Gibbs energy function is a specific vocabulary that means a formula that can calculate the weights and cost.

$$E(\alpha, p, \theta, z) = U(\beta, p, \theta, z) + V(\alpha, z). \tag{1}$$

U is a method when the target pixels are classified as foreground or background models, which are obtained by matrix D iteration of classifying and rewriting pixel:

$$\begin{aligned} U(\alpha, p, \theta, z) &= \sum_n D(\alpha_n, p_n, \theta, z_n), \\ D(\alpha_n, p_n, \theta, z_n) &= \log \pi(\alpha_n, p_n) + \frac{1}{2} \log \det \sum (\alpha_n, p_n) \\ &\quad + \frac{1}{2} \left[z_n - \mu(\alpha_n, p_n)^{-1} \right] \\ &\quad \cdot \sum (\alpha_n, p_n)^{-1} \left[z_n - \mu(\alpha_n, p_n) \right]^{-1}. \end{aligned} \tag{2}$$

V is the boundary energy smoothing term, when the boundary pixels are discontinuous; it will do some boundary pixel classification processing:

$$V(\alpha, z) = \gamma \sum_{(x,y)\in c} \left[\alpha_x, \alpha_y \right] \exp - \beta \left\| z_x - z_y \right\|^2. \tag{3}$$

θ makes the boundary energy more and more little by iterative method; after iteration, the parameters of the Gauss mixture model of target and background are higher.

$$\begin{aligned} \theta = \{ &\pi(\alpha, p), \ \mu(\alpha, p), \ \sum(\alpha, p), \ \alpha = 0, 1, \ k \\ &= 1 \cdots k \}. \end{aligned} \tag{4}$$

2.2. Correction of Yellow Sticky Board. Because the shooting angle is not fixed, the positions of yellow sticky board which has been extracted are not fixed. In order to improve the accuracy of aphid identification and counting, the yellow sticky board must be corrected to the same size and the same position when a target image has been segmented; getting the elevation view, the aphids can be counted accurately.

Transformation from distortion to elevation view can be completed by using perspective correction method [14] which uses the four vertexes of distortion image and elevation view as the correction basis. The top left vertex (x, y) is the key of correction. Firstly, the corrected image matrix $[x_2 \ \ y_2 \ \ z_2]$ is obtained by using the distorted image matrix $[x_1 \ \ y_1 \ \ z_1]$; you can get the top left vertex (x, y) by using the relationship between $[x_1 \ \ y_1 \ \ z_1]$. Secondly, according to top left vertex, the length and width of the rectangle can be used to obtain the information of four vertices.

$$\begin{aligned} D &= [x_2 \ \ y_2 \ \ z_2] = [x_2 \ \ y_2 \ \ z_2] \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, \\ x &= \frac{x_2}{z_2} = \frac{a_{11}x_1 + a_{21}y_1 + a_{31}}{a_{13}x_1 + a_{23}y_1 + a_{33}}, \\ x &= \frac{y_2}{z_2} = \frac{a_{12}x_1 + a_{22}y_1 + a_{32}}{a_{13}x_1 + a_{23}y_1 + a_{33}}. \end{aligned} \tag{5}$$

Figure 1 is the elevation view of yellow sticky board by using GrabCut and perspective correction method. We can see that the picture has a clear boundary and regular shape which will make the aphids extracted easily.

2.3. The Image of the Yellow Sticky Board Preprocessing. The color image change into gray image can reduce amount of computation in the image processing process and reduce the complexity of the calculation [15, 16], the gray scale value obtained by the weighted gray scale method is more in line with the perception of the corresponding gray scale of each pixel, and the brightness is moderate; it is conducive to the next aphid image extraction.

2.4. Aphids Extraction

2.4.1. Image Difference. Due to the influence of light, even if more than two thresholds segmentations are proposed; each of the pictures taken in the natural environment has an uncertain gray value, so the threshold is uncertain too. Aphid color is relatively single, so the impact of light is mainly reflected in the background; an image can be seen to consist of foreground and background; when the background is removed, the influence of light is removed. Regarding the

Figure 1: Perspective correction.

Figure 2: Image difference method for extracting aphids.

process of image difference, firstly the original gray image will be taken for fuzzification as SRC and then subtract the fuzzy background DST. The difference that results, DIFF, is the aphids image.

$$\text{SRC} - \text{DST} = \text{DIFF}. \tag{6}$$

2.4.2. Fuzzy Background. The gray scale image after blurring is compared with the gray scale distribution in the original gray scale graph, and if the local gray scale distribution is equal, it can be used as the background image. Compared with the fuzzy effect of the three filters, the Gaussian weight filter has the smallest noise, which is consistent with the original gray scale distribution. When the kernel value is (51, 51), the background blurring is the best.

2.4.3. Threshold Selection. Image difference is only gray image, and the gray image is not the result of image segmentation and needs the threshold segmentation [17]. A large number of aphids are missing counted by using binary threshold method, while there are too many contour break points by using self-adaption threshold method. So this paper uses the OSTU threshold segmentation method which has a clear and integrated contour as the binarization method.

2.4.4. Morphological Operation. In the natural environment, the aphids on the yellow sticky board have a high base and a high density, and some of the aphids' adhesion does not affect the overall counting result and requires some corrosion of the partially adhered aphids to reduce the error. This paper uses the elliptical nucleus with a nuclear value of (4, 4) for corrosion operation. This method has a good effect on elimination of adhesions.

$$E(x) = \{a \mid B_a \in X\} = X \Theta B. \tag{7}$$

Figure 2 shows the use of image difference algorithm to obtain the aphid images; aphid images are clear and complete.

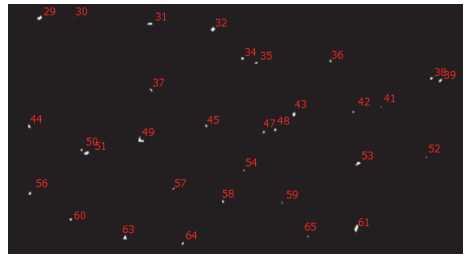


Figure 3: A digital mark of a binary image.

2.5. Aphids Count

2.5.1. Feature Extraction. In order to count accurately, it is necessary to identify whether there are aphids. With access to relevant information we can see that the aphid's length is between 1.5 mm and 4.9 mm. Most aphids are 2 mm in length, and there are differences in the lengths of other insects in the orchard; for example, the apple leaf mites are about 0.3 mm. So distinguish between aphids and nonaphids according to area characteristics. First, all the contours in the binary image are identified as digital marks, and the sum of the pixels in each connection domain is calculated to find the relationship between the actual length of the aphids and the pixel values, as shown in Figure 3.

According to Table 1, based on the relationship between the pixel values in each connected domain and the actual lengths of aphids, we can draw a conclusion that when the pixel value is less than 20, the insect's body length increases by 0.1 mm and the pixel value is increased by 1 px~2 px; when the pixel value is higher than 20, the insect's body length increases by 0.1 mm and the pixel value is increased by 3 px~4 px. In this paper, the aphids involved are mostly 1.5 mm to 2.8 mm, so when the aphid area characteristics range from 15 px to 48 px, nonaphids range from 1 px to 10 px is the most suitable. According to the range of aphid area characteristics derived from the above study to identify aphid images on the yellow sticky board [18], in the obtained result graph, the aphid

Table 1: Part of the connection domain number, pixel values, and the relationship between the lengths of pests.

Pest category	The pixel value within the contour (contour number, pixel value, and aphid length/mm)		
Aphid	(29, 41, 2.6)	(31, 28, 2.2)	(32, 35, 2.5)
	(34, 21, 2)	(35, 19, 1.9)	(36, 19, 1.9)
	(37, 20, 1.9)	(38, 21, 1.9)	(39, 27, 2.2)
	(43, 32, 2.2)	(44, 29, 2.2)	(45, 19, 1.8)
	(47, 20, 1.9)	(48, 20, 1.9)	(49, 42, 2.6)
	(50, 18, 1.9)	(51, 40, 2.6)	(53, 35, 2.5)
	(56, 25, 2.1)	(58, 19, 1.8)	(60, 23, 2.1)
	(61, 48, 2.8)	(63, 32, 2.4)	(64, 19, 1.9)
Nonaphids	(30, 2, 0.2)	(41, 4, 0.4)	(52, 6, 0.6)
	(42, 9, 0.9)	(54, 8, 0.9)	(57, 10, 1)
	(59, 9, 0.9)	(65, 10, 1)	

image is marked with white and nonaphid images are not marked with white, consistent with the human eye.

2.5.2. Aphids Counting Method. Using findContours() contour extraction method to find all the contours of the image. Firstly, the outline which can be detected in the image will be converted to vector form to iterate, and then the aphids section will be screened out by using "for-cycle" according to their area features [19].

2.6. Design and Implementation Based on the Android Platform

2.6.1. Software Overall Flow Chart. In order to meet the needs of the public for the convenience of equipment, Android system as a mobile platform mainstream system [20] in the mobile side of the software development plays a guiding role; based on the Android platform aphid count APP will greatly facilitate the majority of farmers and agricultural science and technology workers. The aphid counting software in this paper uses the Java programming language in the Android studio platform with OpenCV computer vision library development for Android 4.0.3 and above. The software implements the counting of aphids; the whole is divided into three modules: image acquisition module, image processing module, and image counting module; as shown in Figure 4, now we design and implement the part of the content of the software.

(i) Image Acquisition Module. The software obtains the picture in two ways: invoking the local camera to shoot in real time or retrieving the existing photo in the local album. After the photo is taken to ensure the quality of the picture, the appropriate image compression can not only reduce the computational complexity of the image processing process but also avoid the overflow caused by the image being too large to cause the program to crash. First create an intent object and set the action of the object as the string of MediaStore, ACTION_IMAGE_CAPTURE. Then use the string of startActivityForResult to start Intent program. After setting the length and width, the image information will be loaded into Imageview by using decodeFile method. At last the image information will return and be displayed on the next interface.

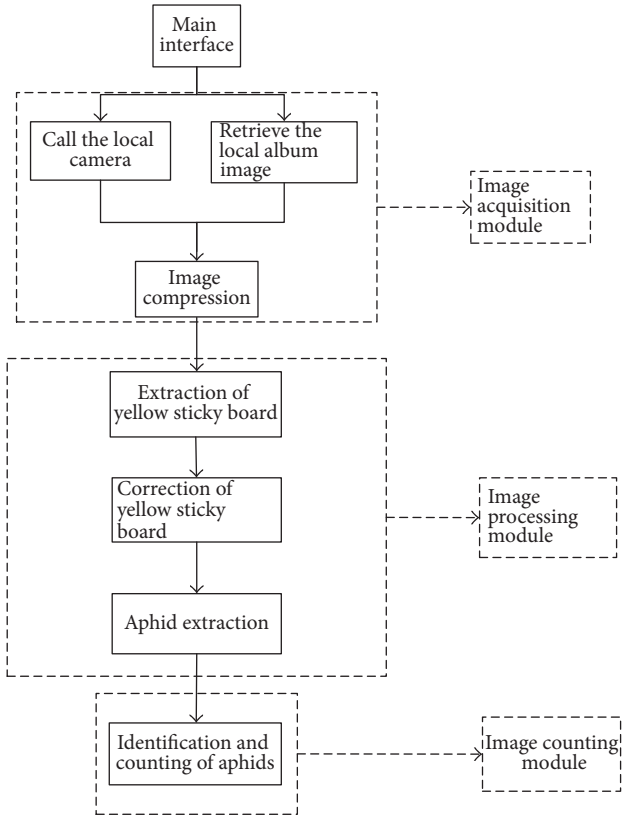
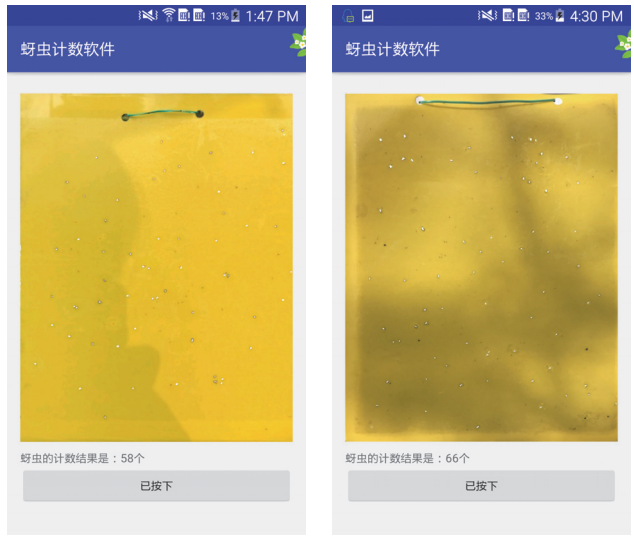


Figure 4: Software flow chart.

(ii) Image Processing Module. In the image processing module, we set "yellow sticky board segmentation" "yellow sticky board correction" "aphids identification" to extract the aphids.

Using GrabCut algorithm to extract yellow sticky board, first create a Rect object to create a rectangular box to select the possible prospects in the image and use the canvas.drawRect () method to draw the rectangle and pass into the mask matrix. Create a mask object as the display position of the segmented yellow sticky board image, which is the image of the CvType.CV_8UC1 channel that needs to be of the same size as the original image. Black is used as the color of scallar which means the background of output image is black as well. Finally the segmentation can be completed by calling the arguments.

Use the perspectiveTransform () method to correct the yellow sticky board. Use the "for-cycle" to traverse the four edge vertexes of the two intersecting lines in the segmented image by using GrabCut. Then compare the coordinates of the four vertexes with the midpoint of the yellow sticky board. For example, in the coordinates (x, y) of the horizontal axis, the value of the vertical axis is less than the corresponding value of the midpoint; then the point is the top left vertex. According to the upper left, upper right, lower left, and lower right of the order of the four vertex coordinates in the variable



(a) In greenhouse (b) Orchard

Figure 5: Practical application of aphid counting software.

called startM, the perspective transformation method can be corrected.

Use single image difference method to extract aphids. Through the Imgproc class cvtColor, color image will be converted to the output image grayMat gray image. Set the parameters of nuclear value Size (w, h) as Size (51, 51); then use the Imgproc.GaussianBlur method to complete the process of fuzzy background. Use Core.absdiff() for differential operation. Use the Imgproc.THRESH_OTSU for threshold operation after difference background image. The differential images need to undergo morphological corrosion to remove the effects of the aphids. Set the morphological action as MORPH_ERODE and set the shape of structure element kernel as MORPH_ELLIPSE with a new Size (5, 5) by using getStructuringElement(). Elimination of adhesions will be completed.

(iii) Image Counting Module. The image counting module filters the contours based on the total pixel range in each connected domain and stores the count on the counter. The output value is the number of aphids.

The counting process can be seen as a traversal of the "funnel"; then we should define two containers: the first container is stored in all the contour data, whether or not meeting the characteristics of the requirements, and the second container is stored after the screening of the contour information. Traverse the contour type to Imgproc.RETR_CCOMP and output it as Imgproc.CHAIN_APPROX_SIMPLE. The "for-cycle" is the key of selecting. Set 15 px to 18 px as the shape of aphids image in the connected domain and set scallar (255, 255, 255) to show aphids as white.

3. Results and Discussions

Using the aphid counting software to count the images in greenhouse and the images in orchards, Figure 5(a) shows the counting results of the aphids in the greenhouse, and Figure 5(b) shows the counting results of the aphids in the orchard.

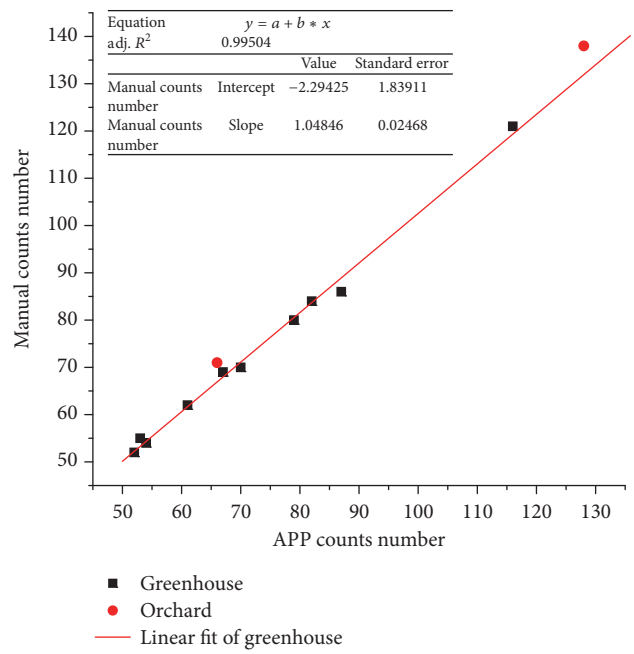


Figure 6: The comparison of accuracy between the manual count and the APP count.

Based on the least squares method, a linear regression analysis has been carried out by taking 10 sets of data from Table 2 in the greenhouse to get the fitting curve. x-axis is the APP counts number and y-axis is the manual counts number. As shown in Figure 6, the relationship of the fitting between the mobile APP count and the manual count is $y = 1.04846x - 2.29425$; the fit degree R^2 factor is up to

Table 2: Error analysis of manual count and APP count.

Image source	Image number	Number of manual counts/month	APP counts number/month	Relative error/%	Accuracy/%
Greenhouse	(1)	52	52	0	100
	(2)	80	79	1.25	98.75
	(3)	62	61	1.61	98.38
	(4)	84	82	2.38	97.61
	(5)	86	87	1.16	98.83
	(6)	55	53	3.7	96.22
	(7)	54	54	0	100
	(8)	69	67	2.9	97.01
	(9)	70	70	0	100
	(10)	121	116	4.13	95.86
Orchard	(11)	71	66	7.04	92.95
	(12)	138	128	7.24	92.75

0.99504, indicating that the aphid recognition and mobile APP counting can be used to achieve functions of recognition and counting. Adding two sets of orchard data from Table 2 into the coordinate system, we can find that the two points are in the vicinity of the curve, indicating that the method is basically effective. Because the orchard environment is more complex, the accuracy rate of the method is slightly worse than the greenhouse; as shown in Table 2, the greenhouse accuracy rate is up to 95% and orchard accuracy rate is up to 92%. In general, the aphids identification and counting method is highly reliable.

4. Conclusion

In the past, aphid identification focuses on the separation of a single kind of aphids, few on different kinds of pests. The count number is less, and the aphids' antennae, foot, and wings need to be extracted precisely, because the operation complexity is high; most of the application software that implements the aphid recognition and counting function is concentrated on the PC side. The traditional counting APP collects the photos through the Internet to transfer data by the PC processing, and then the results return back to the mobile App, the process Overreliance on the PC side and the Internet; its real time performance is poor. Collection of aphids requires special equipment; this professional equipment is not convenient and also very expensive. The device that collects the aphids is not universal.

This paper uses yellow sticky board in the natural environment to attracts aphids and some other pests, and collects images in the real growth environment of plants. Aphid counting software overcomes the complex natural environment and the impact of light caused by image processing difficulties; it does not rely on the PC side and the network feedback; it can quickly identify and count aphids, which are the representative of the pest group; it has very practical value. The accuracy of the aphid counting software is compared with the result of manual counting; the accuracy rate of the aphid counting software is more than 95% in the greenhouse environment; the accuracy rate of the aphid counting software is more than 92%. The operation is smooth and convenient. The method proposed in this paper can extract and correct the image of yellow sticky board, eliminate influence of illumination on the image, and count aphids accurately.

Due to the limitations of the season and the study site, the aphid counting software needs to design a memory storage module to achieve the same number of yellow sticky insect boards to make trend prediction function.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the new facilities of agricultural machinery research and development (17227206D).

References

[1] X. Zhan-li, L. Xiao-xia, Z. Qing-wen et al., "Characteristics of aphid age identification of wheat long pipe," *Acta Entomologica Sinica*, vol. 01, pp. 81–87, 2014.

[2] R. Cassani, T. H. Falk, F. J. Fraga, P. A. Kanda, and R. Anghinah, "Towards automated EEG-Based Alzheimer's disease diagnosis using relevance vector machines," in *Proceedings of the 5th ISSNIP - IEEE Biosignals and Biorobotics Conference (BRC '14)*, pp. 1–6, IEEE, May 2014.

[3] S. Chakraborty and A. C. Newton, "Climate change, plant diseases and food security: an overview," *Plant Pathology*, vol. 60, no. 1, pp. 2–14, 2011.

[4] M. Pal and G. M. Foody, "Evaluation of SVM, RVM and SMLR for accurate image classification with limited ground data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 5, pp. 1344–1355, 2012.

[5] H. N. Patel, R. K. Jain, and M. V. Joshi, "Fruit detection using improved multiple features based algorithm," *International Journal of Computer Applications*, vol. 13, no. 2, pp. 1–5, 2011.

[6] J. Luo, D. Wang, Y. Dong, W. Huang, and J. Wang, "Developing an aphid damage hyperspectral index for detecting aphid (Hemiptera: Aphididae) damage levels in winter wheat," in *Proceedings of the 2011 IEEE International Geoscience and*

Remote Sensing Symposium (IGARSS '11), pp. 1744–1747, IEEE, July 2011.

[7] C. Yue-hua, H. U. Xiao-guang, and Z. Chang-li, "Study on wheat pest segmentation algorithm based on machine vision," *Journal of Agricultural Engineering*, vol. 12, pp. 187–191, 2007.

[8] Q. Bai-jing, W. Tian-bo, L. Juan-juan et al., "Image recognition and counting method of cucumber aphids," *Journal of Agricultural Mechanization*, vol. 08, pp. 151–155, 2010.

[9] R. Jingxiao, *Image-based pest automatic counting and recognition system*, Zhejiang University, 2014.

[10] M. Vincent et al., "Towards a Video Camera Network for Early Pest Detection in Greenhouses," 2008, http://www-sop.inria.fr/pulsar/projects/bioserre/file/endure.pdf.

[11] P. Tirelli, N. A. Borghese, F. Pedersini, G. Galassi, and R. Oberti, "Automatic monitoring of pest insects traps by Zigbee-based wireless networking of image sensors," in *Proceedings of the 2011 IEEE International Instrumentation and Measurement Technology Conference (I2MTC '11)*, pp. 1–5, IEEE, May 2011.

[12] J. Wang, L. Ji, A. Liang, and D. Yuan, "The identification of butterfly families using content-based image retrieval," *Biosystems Engineering*, vol. 111, no. 1, pp. 24–32, 2012.

[13] H. Ran, *Image Segmentation Algorithm Based on Visual Attention Mechanism and Its Application*, Beijing Jiaotong University, 2016.

[14] D. Qin, W. Yanjie, and H. Han Guangliang, "Perspective correction based on improved Hough transform and perspective transformation," *Journal of Liquid Crystals and Display*, vol. 04, pp. 552–556, 2012.

[15] J. Faria, T. Martins, M. Ferreira, and C. Santos, "A computer vision system for color grading wood boards using Fuzzy Logic," in *Proceedings of the 2008 IEEE International Symposium on Industrial Electronics (ISIE '08)*, pp. 1082–1087, July 2008.

[16] Z. Jian-wei, W. Yong-mo, and S. Zu-ruo, "Study on automatic counting of aphids in wheat field," *Chinese Journal of Agricultural Engineering*, vol. 09, pp. 159–162, 2006.

[17] W. Zhi-bin, W. Kai-yi, Z. Shui-fa, and M. Cui-xia, "Class counting algorithm of whitefly based on K-means clustering and elliptic fitting method," *Journal of Agricultural Engineering*, vol. 01, pp. 105–112, 2014.

[18] B. Kui, J. Chao, L. Lei, and W. Cheng, "Non-nondestructive measurement of wheat morphological characteristics based on morphological image processing," *Journal of Agricultural Engineering*, vol. 12, pp. 212–216, 2010.

[19] N. Gonçalves, V. Carvalho, M. Belsley, R. M. Vasconcelos, F. O. Soares, and J. Machado, "Yarn features extraction using image processing and computer vision—a study with cotton and polyester yarns," *Measurement*, vol. 68, pp. 1–15, 2015.

[20] W. Huaxu, "Development and design of software framework for android platform image processing software," *Software*, vol. 02, pp. 46-47, 2014.

Hindawi
Journal of Sensors
Volume 2017, Article ID 3296874, 13 pages
https://doi.org/10.1155/2017/3296874

Review Article

A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles

Adrian Carrio, Carlos Sampedro, Alejandro Rodriguez-Ramos, and Pascual Campoy

Computer Vision and Aerial Robotics Group, Centre for Automation and Robotics (CAR) UPM-CSIC, Universidad Politécnica de Madrid, Calle José Gutiérrez Abascal 2, 28006 Madrid, Spain

Correspondence should be addressed to Adrian Carrio; adrian.carrio@upm.es

Received 28 April 2017; Accepted 18 June 2017; Published 14 August 2017

Academic Editor: Vera Tyrsa

Deep learning is recently showing outstanding results for solving a wide variety of robotic tasks in the areas of perception, planning, localization, and control. Its excellent capabilities for learning representations from the complex data acquired in real environments make it extremely suitable for many kinds of autonomous robotic applications. In parallel, Unmanned Aerial Vehicles (UAVs) are currently being extensively applied for several types of civilian tasks in applications going from security, surveillance, and disaster rescue to parcel delivery or warehouse management. In this paper, a thorough review has been performed on recent reported uses and applications of deep learning for UAVs, including the most relevant developments as well as their performances and limitations. In addition, a detailed explanation of the main deep learning techniques is provided. We conclude with a description of the main challenges for the application of deep learning for UAV-based solutions.

1. Introduction

Recent successes of deep learning techniques in solving many complex tasks by learning from raw sensor data have created a lot of excitement in the research community. However, deep learning is not a recent technology. It started being used back in 1971, when Ivakhnenko [1] trained an 8-layer neural network using the Group Method of Data Handling (GMDH) algorithm. The term deep learning began to be used during the 2000s, when Convolutional Neural Networks (CNNs), a computational original model from the 80s [2] but trained efficiently in the 90s [3], were able to provide decent results in visual object recognition tasks. At the time, datasets were small and computers were not powerful enough, so the performance was often similar to or worse than that of classical Computer Vision algorithms. The development of CUDA for Nvidia GPUs which enabled over 1000 GFLOPS per second and the publication of the ImageNet dataset, with 1.2 million images classified in 1000 categories [4], were important facts for the popularization of CNNs with several layers (10^9 to 10^{10} connections and 10^7 to 10^9 parameters). These deep models show great performance not only in Computer Vision tasks but also in other tasks such as speech recognition, signal processing, and natural language processing [5]. More details about recent advances in deep learning can be found in [6, 7].

An evidence of the suitability of deep learning for many kinds of autonomous robotic applications is the increasing trend in *deep learning robot* related scientific publications over the past decades, which is expected to continue growing [8].

Due to the versatility, automation capabilities, and low cost of Unmanned Aerial Vehicles (UAVs), civilian applications in diverse fields have experienced a drastic increase during the last years. Some examples include power line inspection [9], wildlife conservation [10], building inspection [11], and precision agriculture [12]. However, UAVs have limitations in the size, weight, and power consumption of the payload and limited range and endurance. These limitations cannot be overlooked and are particularly relevant when deep learning algorithms are required to run on board a UAV.

In this survey, we have grouped publications according to the taxonomy proposed in Aerostack [13], which is aerial robotics architecture consistent with the usual components

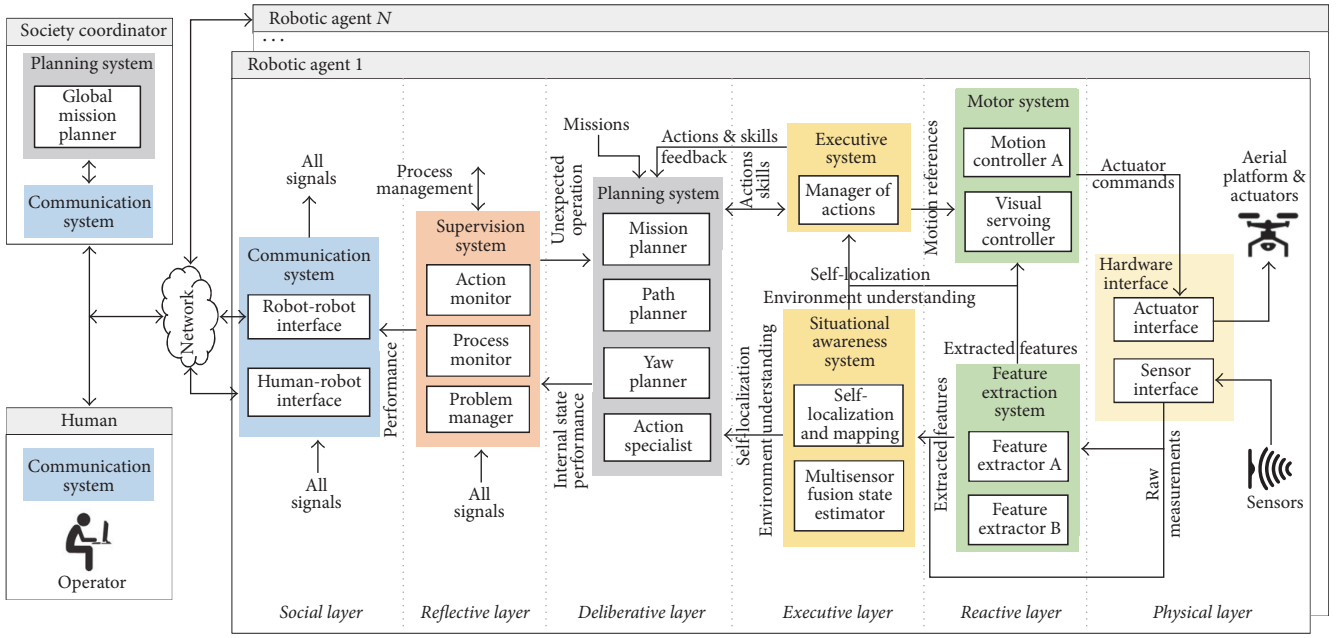


Figure 1: Aerostack architecture, consisting of a layered structure, corresponding to the different abstraction levels in an unmanned aerial robotic system. The architecture has been applied here to systematically classify deep learning-based algorithms available in the state of the art which have been deployed for applications with Unmanned Aerial Vehicles.

related to perception, guidance, navigation, and control of unmanned rotorcraft systems. The purpose of referring to this architecture, depicted in Figure 1, is to achieve a better understanding about the nature of the components to the aerial robotic systems analyzed. Using this taxonomy also helps identify the components in which deep learning has not been applied yet. According to Aerostack, the components constituting an unmanned aerial robotic system can be classified into the following systems and interfaces:

(i) *Hardware interfaces*: this category includes interfaces with both sensors and actuators

(ii) *Motor system*: the components of a motor system are motion controllers, which typically receive commands of desired values for a variable (position, orientation, or speed). These desired values are translated into low-level commands that are sent to actuators

(iii) *Feature extraction system*: feature extraction here refers to the extraction of useful features or representations from sensor data. The task of most deep learning algorithms is to learn data representations, so feature extraction systems are somewhat inherent to deep learning algorithms

(iv) *Situational awareness system*: this system includes components that compile sensor information into state variables regarding the robot and its environment, pursuing environment understanding. An example component within the situational awareness system is SLAM algorithms

(v) *Executive system*: this system receives high-level symbolic actions and generates detailed behaviour sequences

(vi) *Planning system*: this type of system generates global solutions to complex tasks by means of planning (e.g., path planning and mission planning)

(vii) *Supervision system*: components in the supervision system simulate self-awareness in the sense of ability to supervise other integrated systems. We can exemplify this type of component with an algorithm that checks whether the robot is actually making progress towards its goal and reacts in the presence of problems (unexpected obstacles, faults, etc.) with recovery actions

(viii) *Communication system*: the components in the communication system are responsible for establishing an adequate communication with human operators and/or other robots

The remainder of this paper is as follows: firstly, Section 2 covers a description of the currently relevant and prominent deep learning algorithms. For the sake of completeness, deep learning algorithms have been included regardless of their direct use in UAV applications. Section 3 presents the state of the art in deep learning for feature extraction in UAV applications. Section 4 surveys UAV applications of deep learning for the development of components of planning and situation awareness systems. Reported applications of deep learning for motion control in UAVs are presented in Section 5. Finally, a discussion of the main challenges for the application of deep learning for UAVs is covered in Section 6.

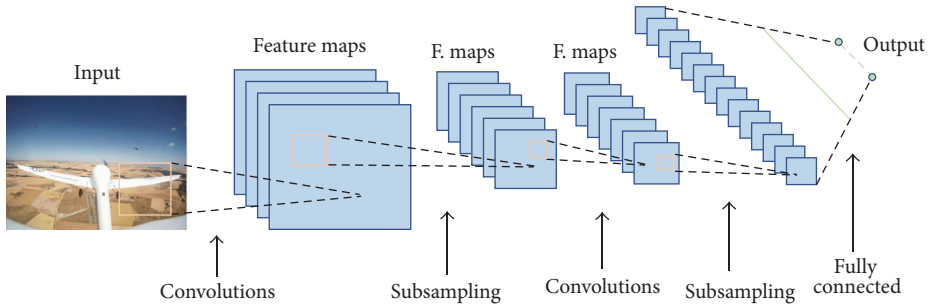


Figure 2: A generic example of a Convolutional Neural Network model. The usual architecture alternates convolution and subsampling layers. Fully connected neurons are used in the last layers.

2. Deep Learning in the Context of Machine Learning

Machine Learning is a capability enabling Artificial Intelligence (AI) systems to learn from data. A good definition for what learning involves is the following: "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E" [15]. The nature of this experience E is typically considered for classifying Machine Learning algorithms into the following three categories: supervised, unsupervised, and reinforcement learning:

(i) In supervised learning, algorithms are presented with a dataset containing a collection of features. Additionally, labels or target values are provided for each sample. This mapping of features to labels of target values is where the knowledge is encoded. Once it has learned, the algorithm is expected to find the mapping from the features of unseen samples to their correct labels or target values.

(ii) The purpose in unsupervised learning is to extract meaningful representations and explain key features of the data. No labels or target values are necessary in this case in order to learn from the data.

(iii) In reinforcement learning algorithms, an AI agent interacts with a real or simulated environment. This interaction provides feedback between the learning system and the interaction experience which is useful to improve performance in the task being learned.

Deep learning algorithms are a subset of Machine Learning algorithms that typically involve learning representations at different hierarchy levels to enable building complex concepts out of simpler ones. The following paragraphs cover the most relevant deep learning technologies currently available in supervised, unsupervised, and reinforcement learning.

2.1. Supervised Learning. Supervised learning algorithms learn how to associate an input with some output, given a training set of examples of inputs and outputs [16]. The following paragraphs cover the most relevant algorithms nowadays in supervised learning: Feedforward Neural Networks, a popular variation of these called Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and a variation of RNNs called Long Short-Term Memory (LSTM) models.

Feedforward Neural Networks, also known as Multilayer Perceptrons (MLPs), are the most common supervised learning models. Their purpose is to work as function approximators: given a sample vector $\mathbf{x}$ with n features, a trained algorithm is expected to produce an output value or classification category $\mathbf{y}$ that is consistent with the mapping of inputs and outputs provided in the training set. The approximated function is usually built by stacking together several hidden layers that are activated in chain to obtain the desired output. The number of hidden layers is usually referred to as the depth of the model, which explains the origin of the term deep learning: learning using models with several layers. These layers are made up of neurons or units whose activation given an input vector $x \in \mathbb{R}^n$ is given by the following equation:

$$a_\theta(x) = g\left(\theta^T x\right), \tag{1}$$

where θ is a vector of n weights and g is an activation function that is usually chosen to be nonlinear. The activation of unit k in layer m given its n inputs (outputs of the previous layer $m-1$) is given by the following equation:

$$a_k^m = g\left(\Theta_{k0}^{m-1} a_0^{m-1} + \Theta_{k1}^{m-1} a_1^{m-1} + \cdots + \Theta_{kn}^{m-1} a_n^{m-1}\right). \tag{2}$$

During the process of learning, the weights in each unit are updated using backpropagation in order to optimize a cost function, which generally indicates the similarity between the desired outputs and the actual ones.

Convolutional Neural Networks (CNNs), depicted in Figure 2, are a specific type of models conceived to accept 2-dimensional input data, such as images or time series data. These models take their name from the mathematical linear operation of convolution which is always present in at least one of the layers of the network. The most typical convolution

operation used in deep learning is 2D convolution of a 2-dimensional image I with a 2-dimensional kernel K, given by the following equation:

$$C(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n) K(i-m, j-n). \quad (3)$$

The output of the convolution operation is usually run through a nonlinear activation function and then further modified by means of a pooling function, which replaces the output in a certain location with a value obtained from nearby outputs. This pooling function helps make the representation learned invariant to small translations of the input and performs subsampling of the input data. The most common pooling function is max pooling, which replaces the output with the maximum activation within a rectangular neighborhood. Convolution and pooling layers are stacked together to achieve feature learning in a hierarchical way. For example, when learning from images, layers closer to the input learn low-level feature representations (i.e., edges and corners) and those closer to the output learn higher level representations (i.e., contours and parts of objects). Once the features of interest have been learned, their activations are used in final layers, which are usually made up of fully connected neurons, to classify the input or perform value regression with it.

In contrast to MLPs, Recurrent Neural Networks (RNNs) are models in which the output is a function of not only the current inputs but also of the previous outputs, which are encoded into a hidden state h. This means that RNNs have memory of the previous outputs and therefore can encode the information present in the sequence itself, something that MLPs cannot do. As a consequence, this type of model can be very useful to learn from sequential data. The memory is encoded into an internal state and updated as indicated in the following equation:

$$h_t = g[Wx_t + Uh_{t-1}], \quad (4)$$

where h_t represents the hidden state at time step t. The weight matrices W (input-to-hidden) and U (hidden-to-hidden) determine the importance given to the current input and to the previous state, respectively. The activation is computed with a third weight matrix V (hidden-to-output) as indicated by the following equation:

$$a_t = Vh_t. \quad (5)$$

RNNs are usually trained using Backpropagation Through Time (BPTT), an extension of backpropagation which takes into account temporality in order to compute the gradients. Using this method with long temporal sequences can lead to several issues. Gradients accumulated over a long sequence can become immeasurably large or extremely small. These problems are referred to as exploding gradients and vanishing gradients, respectively. Exploding gradients are easier to solve, as they can be truncated or squashed, whereas vanishing gradients can become too small for networks to learn from and for the resolution of a computer to enable its representation.

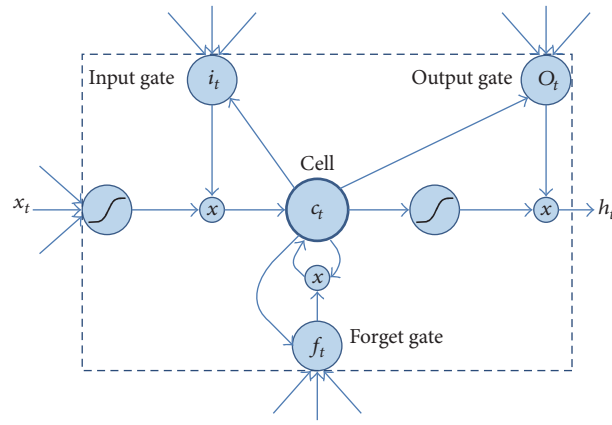


Figure 3: A long-short term memory model, adapted from the original figure in [14]. Learned weights control how data enter and leave and are deleted through the use of gates.

Long Short-Term Memory (LSTM) models are a type of RNN architecture proposed in 1997 by Hochreiter and Schmidhuber [17] which successfully overcomes the problem of vanishing gradients by maintaining a more constant error through the use of gated cells, which effectively allow for continuous learning over a larger number of time steps. A typical LSTM cell is depicted in Figure 3. The input, output, and forget gate vector activations in a standard LSTM are given as follows:

$$i_t = g(W_i x_t + U_i h_{t-1}),$$
$$o_t = g(W_o x_t + U_o h_{t-1}), \quad (6)$$
$$f_t = g(W_f x_t + U_f h_{t-1}).$$

The cell state vector activation is given by the following equation:

$$c_t = f_t \circ c_{t-1} + i_t \circ g(W_c x_t + U_c h_{t-1}), \quad (7)$$

where $\circ$ represents the Hadamard product. Finally, the output gate vector activation is given by the following equation:

$$h_t = o_t \circ g(c_t). \quad (8)$$

As it has been already stated, LSTM gated cells in RNNs have internal recurrence, besides the outer recurrence of RNNs. Cells store an internal state, which can be written to and read from them. There are gates controlling how data enter and leave and are deleted from this cell state. Those gates act on the signals they receive, and, similar to a standard neural network, they block or pass on information based on its strength and importance using their own sets of weights. Those weights, as the weights that modulate input and hidden states, are adjusted via the recurrent network's learning process. The cells learn when to allow data to enter and leave or be deleted through the iterative process of making guesses,

backpropagating error, and adjusting weights via gradient descent. This type of model architecture allows successful learning from long sequences, helping to capture diverse time scales and remote dependencies. Practical aspects on the use of LSTMs and other deep learning architectures can be found in [18].

2.2. Unsupervised Learning. Unsupervised learning aims towards the development of models that are capable of extracting meaningful and high-level representations from high-dimensional sensory unlabeled data. This functionality is inspired by the visual cortex which requires very small amount of labeled data.

Deep Generative Models such as Deep Belief Networks (DBNs) [19, 20] allow the learning of several layers of nonlinear features in an unsupervised manner. DBNs are built by stacking several Restricted Boltzmann Machines (RBMs) [21, 22], resulting in a hybrid model in which the top two layers form a RBM and the bottom layers act as a directed graph constituting a Sigmoid Belief Network (SBN). The learning algorithm proposed in [19] is supposed to be one of the first efficient ways of learning DBNs by introducing a greedy layer-by-layer training in order to obtain a deep hierarchical model. In this greedy learning procedure, the hidden activity patterns obtained in the current layer are used as the "visible" data for training the RBM of the next layer. Once the stacked RBMs have been learned and combined to form a DBN, a fine-tuning procedure using a contrastive version of the wake-sleep algorithm [23] is applied.

For a better understanding, the theoretical details of RBMs are provided in the following equations. The energy of a joint configuration $\{\mathbf{v}, \mathbf{h}\}$ can be calculated as follows:

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i \in \text{vis}} v_i b_i - \sum_{j \in \text{hid}} h_j a_j - \sum_{i,j} W_{ij} v_i h_j, \tag{9}$$

where $\theta = \{W, b, a\}$ represent the model parameters. $\mathbf{v} \in \{0, 1\}$ are the "visible" stochastic binary units, which are connected to the "hidden" stochastic binary units $\mathbf{h} \in \{0, 1\}$. The bias terms are denoted by b_i for the visible units and a_j for the hidden units.

The probability of a joint configuration over both visible and hidden units depends on the energy of that joint configuration and is given by (10), where $Z(\theta)$ represents the partition function (see (11)):

$$P(\mathbf{v}, \mathbf{h}; \theta) = \frac{1}{Z(\theta)} \exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right), \tag{10}$$

$$Z(\theta) = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \left(\exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right)\right). \tag{11}$$

The probability assigned by the model to a visible vector $\mathbf{v}$ can be computed as expressed in the following equation:

$$P(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{h} \exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right). \tag{12}$$

The conditional distributions over hidden variables $\mathbf{h}$ and visible variables $\mathbf{v}$ can be extracted using (13). Once a training sample is presented to the model, the binary states of the hidden variables are set to 1 with probability given by (14). Analogously, once the binary states of the hidden variables are computed, the binary states of the visible units are set to 1 with a probability given by (15).

$$\begin{aligned} P(\mathbf{h} \mid \mathbf{v}; \theta) &= \prod_j p\left(h_j \mid v\right), \\ P(\mathbf{v} \mid \mathbf{h}; \theta) &= \prod_i p\left(v_i \mid h\right), \end{aligned} \tag{13}$$

$$p\left(h_j = 1 \mid v\right) = \sigma\left(\sum_i W_{ij} v_i + a_j\right), \tag{14}$$

$$p\left(v_i = 1 \mid h\right) = \sigma\left(\sum_j W_{ij} h_j + b_i\right), \tag{15}$$

where $\sigma(z) = 1/1 + \exp(-z)$ is the logistic function.

For training the RBM model, the learning is conducted by applying the Contrastive Divergence algorithm [22], in which the update rule applied to the model parameters is given by the following equation:

$$\Delta W_{ij} = \epsilon\left(\left\langle v_i h_j \right\rangle_{\text{data}} - \left\langle v_i h_j \right\rangle_{\text{recons}}\right), \tag{16}$$

where ϵ is the learning rate, $\langle v_i h_j \rangle_{\text{data}}$ represents the expected value of the product of visible and hidden states at thermal equilibrium, when training data is presented to the model, and $\langle v_i h_j \rangle_{\text{recons}}$ is the expected value of the product of visible and hidden states after running a Gibbs chain.

Deep neural networks can also be utilized for dimensionality reduction of the input data. For this purpose, deep "autoencoders" [24, 25] have been shown to provide successful results in a wide variety of applications such as document retrieval [26] and image retrieval [27]. An autoencoder (see Figure 4) is an unsupervised neural network in which the target values are set to be equal to the inputs. Autoencoders are mainly composed of an "encoder" network, which transforms the input data into a low-dimensional code, and a "decoder" network, which reconstructs the data from the code. Training these deep models involves minimizing the error between the original data and its reconstruction. In this process, the weights initialization is critical to avoid reaching a bad local optimum; thus some authors have proposed a pretrained stage based on stacked RBMs and a fine-tuning stage using backpropagation [24, 27]. In addition, the encoder part of the autoencoder can serve as a good unsupervised nonlinear feature extractor. In this field, the use of Stacked Denoising Autoencoders (SDAE) [25] has been proven to be an effective unsupervised feature extractor in different classification problems. The experiments presented in [25] showed that training denoising autoencoders with higher noise levels forced the model to extract more distinctive and less local features.

2.3. Deep Reinforcement Learning. In reinforcement learning, an agent is defined to interact with an environment, seeking to find the best action for each state at any step in time (see

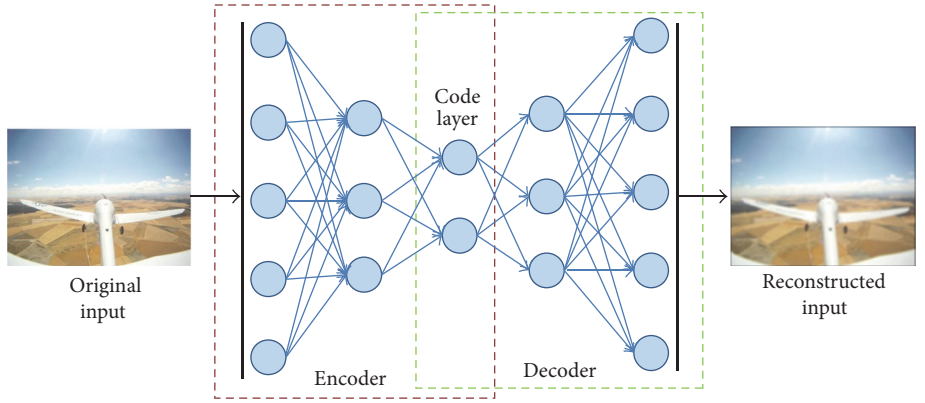


Figure 4: Deep autoencoder. An autoencoder consists of an encoder network, which transforms the original input data into a low-dimensional code, and a decoder network, which reconstructs the data from the code.

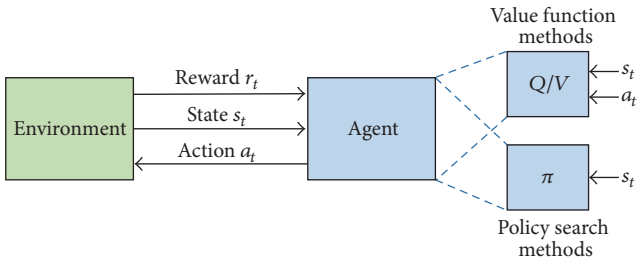


Figure 5: Generic structure of a reinforcement learning problem. The optimization methods to solve the reinforcement learning problem are mainly categorized into value function and policy search methods.

Figure 5). The agent must balance exploration and exploitation of the state space in order to find the optimal policy that maximizes the accumulated reward from the interaction with the environment. In this context, an agent modifies its behaviour or policy with the awareness of the states, actions taken, and rewards for every time step. Reinforcement learning composes an optimization process throughout the whole state space in order to maximize the accumulated reward. Robotic problems are often task-based with temporal structure. These types of problems are suitable to be solved by means of a reinforcement learning framework [28].

The standard reinforcement learning theory states that an agent is able to obtain a policy, which maps every state $s \in \mathbb{S}$ to an action $a \in \mathbb{A}$, where $\mathbb{S}$ is the state space (possible states of the agent in the environment) and $\mathbb{A}$ is the finite action space. The inner dynamics of the agent are represented by the transition probability model $p(s_{t+1} \mid s_t, a_t)$ at time t. The policy can be stochastic $\pi(a \mid s)$, with a probability associated with each possible action, or deterministic $\pi(s)$. In each time step, the policy determines the action to be chosen and the reward $r(s_t, a_t)$ is observed from the environment. The goal of the agent is to maximize the accumulated discounted reward $R_t = \sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i)$ from a state at time t to time T ($T = \infty$ for infinite horizon problems) [29]. The discount factor γ is defined to allocate different weights for the future rewards.

For a specific policy π, the value function V^π in (17) is a representation of the expectation of the accumulated discounted reward R_t for each state $s \in \mathbb{S}$ (assuming a deterministic policy $\pi(s_t)$):

$$V^\pi(s_t) = \mathbb{E}\left[R_t \mid s_t,\ a_t = \pi(s_t)\right]. \tag{17}$$

An equivalent of the value function is represented by the action-value function Q^π in (18) for every action-state pair (s_t, a_t):

$$Q^\pi(s_t, a_t) = r(s_t, a_t) + \gamma \sum_{s_{t+1}} p(s_{t+1} \mid s_t, a_t)\, V^\pi(s_{t+1}). \tag{18}$$

The optimal policy π^* shall be the one that maximizes the value function (or equivalently the action-value function), as in the following equation:

$$\pi^* = \arg\max_\pi V^\pi(s_t). \tag{19}$$

A general problem in real robotic applications is that the state and action spaces are often continuous spaces. A continuous state and/or action space can make the optimization problem intractable, due to the overwhelming set of different states and/or actions. As a general framework for representation, reinforcement learning methods are enhanced through deep learning to aid the design for feature representation, which is known as deep reinforcement learning. Reinforcement learning and optimal control aim at finding the optimal policy π^* by means of several methods. The optimal solution can be searched in this original primal problem, or the dual formulation V^*, Q^* can be the optimization objective. In this review, deep reinforcement learning methods are divided into two main categories: value function and policy search methods.

2.3.1. Value Function Methods. These methods seek to find optimal V^*, Q^*, from which the optimal policy π^* in (20) is directly derived. Q-learning approaches are based on the

optimization of the action-value function Q, based on the *Bellman Optimality Equation* [29] for Q (see (21)):

$$\pi^* = \arg\max_{a_t} Q^*(s_t, a_t), \tag{20}$$

$$Q^*(s_t, a_t) = \mathbb{E}\left[r(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})\right]. \tag{21}$$

Deep Q-Network (DQN) [30, 31] method estimates the action-value function (see (22)) by means of a CNN model with a set of weights θ as $Q^*(s, a) \approx Q(s, a; \theta)$:

$$\begin{aligned} Q_i^*(s_t, a_t) &= y_i \\ &= \mathbb{E}\left[r(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta_{i-1}) \mid s_t, a_t\right]. \end{aligned} \tag{22}$$

The CNN can be trained by minimizing a sequence of loss functions $L_i(\theta_i)$ which are optimized in each iteration i as shown in the following equation:

$$L_i(\theta_i) = \mathbb{E}\left[(y_i - Q(s_t, a_t; \theta_i))^2\right]. \tag{23}$$

The state s of the DQN algorithm is the raw image and it has been widely tested with Atari games [31]. DQN is not designed for continuous tasks; thus this method may find difficulties approaching some robotics problems previously solved by continuous control. Continuous Q-learning with Normalized Advantage Functions (NAF) overcomes this issue by the use of a neural network that separately outputs a value function $V(x)$ and an advantage term $A(x, u)$, which is parametrized as a quadratic function of nonlinear features [32]. These two functions compose final $Q(x, u \mid \theta^Q)$, given by the following equation:

$$Q(x, u \mid \theta^Q) = A(x, u \mid \theta^A) + V(x \mid \theta^V) \tag{24}$$

with x being the state, u being the action, and θ^Q, θ^A, and θ^V being the sets of weights of Q, A, and V functions, respectively. This representation allows simplifying more standard actor-critic style algorithms, while preserving the benefits of nonlinear value function approximation [32]. NAF is valid for continuous control tasks and takes advantage of trained models to approximate the standard model-free value function.

2.3.2. Policy Search Methods. Policy-based reinforcement learning methods aim towards directly searching for the optimal policy π^*, which provides a feasible framework for continuous control. Deep Deterministic Policy Gradient (DDPG) [33] is based on the actor-critic paradigm [29], with two neural networks to approximate a greedy deterministic policy (actor) and Q function (critic). The actor network is updated by applying the chain rule to the expected return from the start distribution J with respect to the actor parameters (see (25)):

$$\nabla_{\theta^\mu} J \approx \mathbb{E}_{s_t \sim \rho^\beta}\left[\nabla_{\theta^\mu} Q(s, a \mid \theta^Q)\big|_{s=s_t, a=\mu(s_t \mid \theta^\mu)}\right]. \tag{25}$$

DDPG method learns with an average factor of 20 times fewer experience steps than DQN [33]. Both DDPG and DQN require large samples datasets, since they are model-free algorithms. Regarding DNN-based Guided Policy Search (DNN-based GPS) [34] method, it learns to map from the tuple raw visual information and joint states directly to joint torques. Compared to the previous works, it managed to perform high-dimensional control, even from imperfect sensor data. DNN-based GPS has been widely applied to robotic control, from manipulation to navigation tasks [35, 36].

3. Deep Learning for Feature Extraction

The main objective of feature extraction systems is to extract representative features from the raw measurements provided by sensors on board a UAV.

3.1. With Image Sensors. Deep learning techniques for feature extraction using image sensors have been applied over a wide range of applications using different imaging technologies (e.g., monocular RGB camera, RGB-D sensors, infrared, etc.). Despite the wide variety of sensors utilized for image processing, main deep learning feature extractors are based on CNNs [67]. As explained in Section 2.1, CNN models consist of several stacked convolution and pooling layers. The convolution layers are responsible for extracting features from the data by convolving the input image with learned filters, while pooling layers provide a dimensionality reduction over previous convolution layers.

In the robotics field, feature extraction systems based on CNN models have been mainly applied for object recognition [42–48] and scene classification [51–54]. Concerning the object recognition task, recent advances have integrated object detection solutions by means of bounding box regression and object classification capabilities within the same CNN model [42–44]. Unsupervised feature learning for object recognition was applied in [68], making fewer requirements on manually labeled training data, the obtainment of which can be an extremely time-consuming and costly process. Regarding the scene classification problem, recent advances have focused on learning efficient and global image representations from the convolutional and fully connected layers from pretrained CNNs in order to obtain representative image features [53]. In [52], it was also shown that the learned features obtained from pretrained CNN models were able to generalize properly even in substantially different domains for those in which they were trained, such as the classification of aerial images. Scene classification on board a Parrot AR.Drone quadrotor was also presented in [40], where a 10-layered CNN was utilized for classifying the input image of a forest trail into three classes, each of which represented the action to be taken in order to maintain the aerial robot on the trail (turn left, go straight, and turn right).

Nowadays, object recognition and scene classification from aerial imagery using deep learning techniques have also acquired a relevant role in agriculture applications. In these kinds of applications, UAVs provide a low-cost platform for aerial image acquisition, while deep learned features

are mainly utilized for plant counting and identification. Several applications have used deep learning techniques for this purpose [12, 49, 50, 55, 56], providing robust systems for monitoring the state of the crops in order to maximize their productivity. In [55], a sparse autoencoder was utilized for unsupervised feature learning in order to perform weed classification from images taken by a multirotor UAV. In [56], a hybrid neural network for crop classification amongst 23 classes was proposed. The hybrid network consisted of the combination of a Feedforward Neural Network for histogram information management and a CNN. In [49], the well-known AlexNet CNN architecture proposed in [69] was utilized in combination with a sliding window object proposal technique for palm tree detection and counting. Other similar approaches have focused on weed scouting using a CNN model for weed specifies classification [12].

Deep learning techniques applied on images taken from UAVs have also gained a lot of importance in monitoring and search and rescue applications, such as jellyfish monitoring [70], road traffic monitoring from UAVs [71], assisting avalanche search and rescue operations with UAV imagery [72], and terrorist identification [73]. In [72, 73], the use of pretrained CNN models for feature extraction is worth noting again. In both cases, the well-known Inception model [74] was used. In [72], the Inception model was utilized with a Support Vector Machine (SVM) classifier for detecting possible survivors, while in [73], a transfer-learning technique was used to fine-tune the Inception network in order to detect possible terrorists.

Most of the presented approaches, especially in the field of object recognition, require the use of GPUs for dealing with real-time constraints. In this sense, the state-of-the-art object recognition systems are based on the approaches presented in [46, 47], in which the object recognizer is able to run at rates from 40 to 90 frames per second on an Nvidia GeForce GTX Titan X.

Despite the good results provided by the aforementioned systems, UAV constraints such as endurance, weight, and payload require the development of specific hardware and software solutions for being embedded on board a UAV. Taking these limitations into account, only few systems in the literature have embedded feature extraction algorithms using deep learning processed by GPU technology on board a UAV. In [75], the problem of automatic detection, localization, and classification (ADLC) of plywood targets was addressed. The solution consisted of a cascade of classifiers based on CNN models trained on an Nvidia Titan X and applied over 24 M-pixel RGB images processed by an Nvidia Jetson TK1 mounted on board a fixed-wing UAV. The ADLC algorithm was processed by combining the CPU cores for the detection stage, allowing the GPU to focus on the classification tasks.

3.2. With Other Sensors. Most of the presented workload using deep learning in the literature has been applied to data capture by image sensors due to the consolidated results obtained using CNN models. However, deep learning techniques cover a wide range of applications and can be used in conjunction with sensors other than cameras, such as acoustic, radar, and laser sensors.

Deep learning techniques for UAVs have been utilized for acoustic data recognition [64, 65]. In [64], a Partially Shared Deep Neural Network (PS-DNN) was proposed to deal with the problem of sound source separation and identification using partially annotated data. For this purpose, the PS-DNN is composed of two partially overlapped subnetworks: one regression network for sound source separation and one classification network responsible for the sound identification. The objective of the regression network for sound source separation is to improve the network training for sound source classification by providing a cleaner sound signal. Results showed that PS-DNN model worked reasonably well for people's voice identification in disastrous situations. The data was collected using a microphone array on board a Parrot Bebop UAV.

In [65], the problem of UAVs identification based on their specific sound was addressed by using a bidirectional LSTM-RNN with 3 layers and 300 LSTM blocks. This model exhibited the best performance amongst other 2 preselected models, namely, Gaussian Mixture Models (GMM) and CNN.

Concerning the radar technology and despite the fact that radar data has not been widely addressed using deep learning techniques for UAVs in the literature, the recent advances presented in [62] are worth mentioning. In this paper, the spectral correlation function (SCF) was captured using a 2.4 GHz Doppler radar sensor that was utilized in order to detect and classify micro-UAVs amongst 3 predefined classes. The model utilized for this purpose was based on a semisupervised DBN trained with the SCF data.

Regarding laser technology, in [66], a novel strategy for detecting safe landing areas based on the point clouds captured from a LIDAR sensor mounted on a helicopter was proposed. In this paper, subvolumes of $1\,m^3$ from a volumetric density map constructed from the original point cloud were used as input to a 3D CNN which was trained to predict the probability of the evaluated area as being a safe landing zone. Several CNN models consisting of one or two convolutional layers were evaluated over synthetic and semisynthetic datasets, showing in both cases good results when using a 3D CNN model with two convolutional layers.

4. Deep Learning for Planning and Situational Awareness

Several deep learning developments have been reported for tasks related to UAV planning and situational awareness. Planning tasks refer to the generation of solutions for complex problems without having to hand-code the environment model or the robot's skills or strategies into a reactive controller. Planning is required in the presence of unstructured, dynamic environments or when there is diversity in the scope and/or the robot's tasks. Typical tasks include path, motion, navigation, or manipulation planning. Situational awareness tasks allow robots to have knowledge about their own state and their environment's state. Some examples of this kind of tasks are robot state estimation, self-localization, and mapping.

4.1. Planning. Path planning for collaborative search and rescue missions with deep learning-based exploration is presented in [57]. This work, where a UAV explores and maps the environment trying to find a traversable path for a ground robot, focuses on minimizing overall deployment time (i.e., both exploration and path traversal). In order to map the terrain and find a traversable path, a CNN is proposed for terrain classification. Instead of using a pretrained CNN, training is done on the spot, allowing training the classifier on demand with the terrain present at the disaster site [58]. However, the model takes around 15 minutes to train.

4.2. Situational Awareness. Cross-view localization of images is achieved with the help of deep learning in [59]. Although the work is presented as a solution for UAV localization, no UAVs were used for image collection and the experiments were based on ground-level images only. The approach is based on mining a library of raw image data to find nearest neighbor visual features (i.e., landmarks) which are then matched with the features extracted from an input query image. A pretrained CNN is used to extract features for matching verification purposes, and although the approach is said to have low computational complexity, authors do not provide details about retrieval time.

Ground-level query images are matched to a reference database of aerial images in [60]. Deep learning is applied here to reduce the wide baseline and appearance variations between both ground-level and aerial images. A pair-based network structure is proposed to learn deep representations from data for distinguishing matched and unmatched cross-view image pairs. Even though the training procedure in the reported experiments took 4 days, the use of fast algorithms such as locality-sensitive hashing allowed for real-time cross-view matching at city scale. The main limitation of their approach is the need to estimate scale, orientation, and dominant depth at test time for ground-level queries.

In [61], a CNN is proposed to generate control actions (the permitted turns for a UAV) given an image captured on board and a global motion plan. This global motion plan indicates the actions to take given a position on the map by means of a potential function. The purpose of the CNN is to learn the mapping from images to position-dependent actions. The process would be equivalent to perform image registration and then generate the control actions given the global motion plan but this behaviour is here learnt to be efficiently encoded in a CNN, demonstrating superior results to classical image registration techniques. However, no tests on real UAV were carried out and no information is provided about execution time, which might complicate the deployment for a real UAV application.

As seen from the presented works, developments in planning and situational awareness with deep learning for UAVs are still quite rudimentary. The path planning approach presented is limited to small-scale disaster sites and the different localization and mapping approaches are still slow and have little accuracy for real UAV applications.

5. Deep Learning for Motion Control

Deep learning techniques for motion control have been recently involved in several scientific researches. Classic control has solved diverse robotic control problems in a precise and analytic manner, allowing robots to perform complex maneuvers. Nevertheless, standard control theory only solves the problem for a specific case and for an approximated robot model, not being able to easily adapt to changes in the robot model and/or to hostile environments (e.g., a propeller on a UAV gets damaged, wind gusts, and rain). In this context, learning from experience is a matter of importance which can overcome numerous stated limitations.

As a key advantage, deep learning methods are able to properly generalize with certain sets of labelled input data. Deep learning allows inferring a pattern from raw inputs, such as images and LIDAR sensor data which can lead to proper behaviour even in unknown situations. Concerning the UAV indoor navigation task, recent advances have led to a successful application of CNNs in order to map images to high-level behaviour directives (e.g., turn left, turn right, rotate left, and rotate right) [38, 39]. In [38], Q function is estimated through a CNN, which is trained in simulation and successfully tested in real experiments. In [39], actions are directly mapped from raw images. In all stated methods, the learned model is run off board, usually taking advantage of a GPU in an external laptop.

With regard to UAV navigation in unstructured environments, some studies have focused on cluttered natural scenarios, such as dense forests or trails [40]. In [40], a DNN model was trained to map image to action probabilities (turn left, go straight, or turn right) with a final *softmax* layer and tested on board by means of an ODROID-U3 processor. The performance of two automated methods, SVM and the method proposed in [76], is latterly compared to that of two human observers.

In [37], navigable areas are predicted from a disparity image in the form of up to three bounding boxes. The center of the biggest bounding box found is selected as the next waypoint. Using this strategy, UAV flights are successfully performed. The main drawback is the requirement to send the disparity images to a host device where all computations are made. The whole pipeline for the UAV horizontal translation, disparity map generation, and waypoint selection takes about 1.3 seconds which makes navigation still quite slow for real applications. On the other hand, low-level motion control is challenging, since tackling with continuous and multi-variable action spaces can become an intractable problem. Nevertheless, recent works have proposed novel methods to learn low-level control policies from imperfect sensor data in simulation [41, 63]. In [63], a Model Predictive Controller (MPC) was used to generate data at training time in order to train a DNN policy, which was allowed to access only raw observations from the UAV onboard sensors. In testing time, the UAV was able to follow an obstacle-free trajectory even in unknown situations. In [41], the well-known Inception v3 model (pretrained CNN) was adapted in order to enable the final layer to provide six action nodes (three transitions and

Table 1: Deep learning-based UAV applications grouped by learning algorithms and application fields.

Learning type	Algorithm	Tasks	Field of application	References
Supervised	CNN	Outdoor navigation	Navigation	[37–39]
		Indoor navigation		[40, 41]
		Object recognition	Generic	[42–45]
				[46–48]
		Object recognition	Agriculture	[49, 50]
		Scene classification	Generic	[51–54]
		Scene classification	Agriculture	[55, 56]
		Path planning	Search & rescue	[57, 58]
		Image registration	Localization Navigation	[59–61]
Unsupervised	Autoencoder	Feature extraction	Agriculture	[55]
	DBN	Feature extraction	UAV identification	[62]
Reinforcement	DQN	—	—	—
	DDPG	—	—	—
	NAF	—	—	—
	GPS	Indoor navigation	Navigation	[63]

three orientations). After retraining, the UAV managed to cross a room filled with a few obstacles in random locations.

Deep learning techniques for robotic motion control can provide increasing benefits in order to infer complex behaviours from raw observation data. Deep learning approaches have the potential of generalization, with the limitations of current methods which have to overcome the difficulties of continuous state and action spaces, as well as issues related to the samples efficiency. Furthermore, novel deep learning models require the usage of GPUs in order to work in real time. In this context, onboard GPUs, Field Programmable Gate Arrays (FPGAs), or Application-Specific Integrated Circuits (ASICs) are a matter of importance which hardware manufacturers shall take into consideration.

6. Discussion

Deep learning has arisen as a promising set of technologies to the current demands for highly autonomous UAV operations, due to its excellent capabilities for learning high-level representations from raw sensor data. Multiple success cases have been reported (Tables 1 and 2) in a wide variety of applications.

A straightforward conclusion from the surveyed articles is that images acquired from UAVs are currently the prevailing type of information being exploited by deep learning, mainly due to the low cost, low weight, and low power consumption of image sensors. This noticeable fact explains the dominance of CNNs among the deep learning algorithms used in UAV applications, given the excellent capabilities of CNNs in extracting useful information from images.

However, deep learning techniques, UAV technology, and the combined use of both still present several challenges, which are preventing faster and further advances in this field.

Challenges in Deep Learning. Deep learning techniques are still facing several challenges, beginning with their own theoretical understanding. An example of this is the lack of knowledge about the geometry of the objective function in deep neural networks or why certain architectures work better than others. Furthermore, a lot of effort is currently being put in finding efficient ways to do unsupervised learning, since collecting large amounts of unlabeled data is nowadays becoming economically and technologically less expensive. Success in this objective will allow algorithms to learn how the world works by simply observing it, as we humans do.

Additionally, as mentioned in Section 2.3, real-world problems that usually involve high-dimensional continuous state spaces (large number of states and/or actions) can turn the problem intractable with current approaches, severely limiting the development of real applications. An efficient way for coping with these types of problems remains as an unsolved challenge.

Challenges in UAV Autonomy. UAV autonomous operations, enabling safe navigation with little or no human supervision, are currently key for the development of several civilian and military applications. However, UAV platforms still have important flight endurance limitations, restricting size, weight, and power consumption of the payload. These limitations arise mainly from the current state of sensor and battery technology and limit the required capabilities for autonomous operations. Undoubtedly, we will see developments in these areas in the forthcoming years.

Furthermore, onboard processing is desired for many UAV operations, especially those where communications can compromise performance, such as when large amounts of data have to be transmitted and/or when there is limited bandwidth available. Today, the design of powerful miniaturized computing devices with low-power consumption,

Table 2: Deep learning-based UAV applications grouped by the type of system within an unmanned aerial systems architecture, the sensor technologies, and the type of learning algorithms: supervised (*S*), unsupervised (*U*), and reinforcement (*R*).

Aerial robot systems	Sensing technologies	Learning algorithms	References
Feature extraction	Image	*S*	[42–45] [46–48] [51–54]
	Image	*S*, *U*	[55]
	Acoustic	*S*	[64, 65]
	Radar	*S*, *U*	[62]
	LIDAR	*S*	[66]
Planning	Image	*S*	[57, 58]
Situational awareness	Image	*S*	[59–61]
Motion control	Image	*S*	[38–41]
	LIDAR	*R*	[63]

particularly GPUs, is an active working field for embedded hardware developers.

Challenges in Deep Learning-Based UAV Applications. This review reveals that, within the architecture of an unmanned aerial system, feature extraction systems are the type of systems in which deep learning algorithms have been more widely applied. This is reasonable given the excellent abilities of deep learning to learn data representations from raw sensor data. Systems regarding higher-level abstractions, such as UAV supervision and planning systems, have so far obtained little regard from the research community. These systems implement complex behaviours that have to be learned and where the application of supervised learning (e.g., the generation of labelled datasets) is complex.

Nevertheless, systems operating at lower levels of abstraction, such as feature extraction systems, still demand great computational resources. These resources are still hard to integrate on board UAVs, requiring powerful communication capabilities and off-board processing. Furthermore, available computational resources are in most cases not compatible with online processing, limiting the applications where reactive behaviours are necessary. This again imposes the aforementioned challenge of developing embedded hardware technology advances but should also encourage researchers to design more efficient deep learning architectures.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Spanish Ministry of Science (Project DPI2014-60139-R). The LAL UPM and the MONCLOA Campus of International Excellence are also acknowledged for funding the predoctoral contract of one of the authors.

References

[1] A. G. Ivakhnenko, "Polynomial theory of complex systems," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 1, no. 4, pp. 364–378, 1971.

[2] K. Fukushima, "Neocognitron: a hierarchical neural network capable of visual pattern recognition," *Neural Networks*, vol. 1, no. 2, pp. 119–130, 1988.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[4] J. Deng, W. Dong, R. Socher et al., "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, Fla, USA, June 2009.

[5] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[6] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[7] J. Gu, Z. Wang, J. Kuen et al., *Recent Advances in Convolutional Neural Networks*, https://arxiv.org/abs/1512.07108.

[8] L. Tai and M. Liu, "Deep-learning in mobile robotics - from perception to control systems: a survey on why and why not," CoRR abs/1612.07139. http://arxiv.org/abs/1612.07139.

[9] C. Martinez, C. Sampedro, A. Chauhan, and P. Campoy, "Towards autonomous detection and tracking of electric towers for aerial power line inspection," in *Proceedings of the 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014*, pp. 284–295, May 2014.

[10] M. A. Olivares-Mendez, C. Fu, P. Ludivig et al., "Towards an autonomous vision-based unmanned aerial system against wildlife poachers," *Sensors*, vol. 15, no. 12, pp. 31362–31391, 2015.

[11] A. Carrio, J. Pestana, J.-L. Sanchez-Lopez et al. et al., "Ubristes: uav-based building rehabilitation with visible and thermal infrared remote sensing," in *Proceedings of the Robot 2015: Second Iberian Robotics Conference*, pp. 245–256, Springer International Publishing, 2016.

[12] L. Li, Y. Fan, X. Huang, and L. Tian, "Real-time uav weed scout for selective weed control by adaptive robust control and

machine learning algorithm," in *Proceedings of the 2016 ASABE Annual International Meeting, American Society of Agricultural and Biological Engineers*, p. 1, 2016.

[13] J. L. Sanchez-Lopez, M. Molina, H. Bavle et al., "A multi-layered component-based approach for the development of aerial robotic systems: The aerostack framework," *Journal of Intelligent & Robotic Systems*, pp. 1–27, 2017.

[14] A. Graves, "Generating sequences with recurrent neural networks," arXiv preprint https://arxiv.org/abs/1308.0850.

[15] T. M. Mitchell, *Machine Learning*, vol. 45 (37), McGraw Hill, Burr Ridge, Ill, USA, 1997.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, Mass, USA, 2016.

[17] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Proceedings of the 10th Annual Conference on Neural Information Processing Systems, NIPS 1996*, pp. 473–479, December 1996.

[18] A. Gibson and J. Patterson, *Deep Learning*, O'Reilly, 2016.

[19] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[20] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle et al., "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems*, vol. 19, pp. 153–160, 2007.

[21] P. Smolensky, "Information processing in dynamical systems: foundations of harmony theory," Tech. Rep., DTIC Document, 1986.

[22] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[23] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal, "The "wake-sleep" algorithm for unsupervised neural networks," *Science*, vol. 268, no. 5214, pp. 1158–1161, 1995.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *American Association for the Advancement of Science. Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] P. Vincent, H. Larochelle, I. Lajoie, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[26] R. Salakhutdinov and G. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[27] A. Krizhevsky and G. E. Hinton, "Using very deep autoencoders for content-based image retrieval," in *Proceedings of the 19th European Symposium on Artificial Neural Networks (ESANN '11)*, Bruges, Belgium, April 2011.

[28] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[29] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1, MIT Press, Cambridge, UK, 1998.

[30] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing atari with deep reinforcement learning," arXiv preprint https://arxiv.org/abs/1312.5602.

[31] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[32] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep q-learning with model-based acceleration," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, vol. 48, pp. 2829–2838, New York, NY, USA, June 2016, preprint https://arxiv.org/abs/1603.00748.

[33] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," preprint https://arxiv.org/abs/1509.02971.

[34] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016, preprint https://arxiv.org/abs/1504.00702.

[35] M. Zhang, Z. McCarthy, C. Finn, S. Levine, and P. Abbeel, "Learning deep neural network policies with continuous memory states," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation, ICRA 2016*, pp. 520–527, May 2016.

[36] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation, ICRA 2016*, pp. 528–535, May 2016.

[37] U. Shah, R. Khawad, and K. M. Krishna, "Deepfly: Towards complete autonomous navigation of mavs with monocular camera," in *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 16*, pp. 59:1–59:8, New York, NY, USA, 2016.

[38] F. Sadeghi and S. Levine, "Real single-image flight without a single real image," preprint https://arxiv.org/pdf/1611.04201.pdf.

[39] D. K. Kim and T. Chen, "Deep neural network for real-time autonomous indoor navigation," preprint https://arxiv.org/abs/1511.04668.

[40] A. Giusti, J. Guzzi, D. C. Ciresan et al., "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, 2016.

[41] K. Kelchtermans and T. Tuytelaars, "How hard is it to cross the room? – training (recurrent) neural networks to steer a uav," preprint https://arxiv.org/abs/1702.07600.

[42] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 580–587, Columbus, Ohio, USA, June 2014.

[43] R. Girshick, "Fast R-CNN," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1440–1448, December 2015.

[44] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, vol. 28, pp. 91–99, 2015.

[45] J. Lee, J. Wang, D. Crandall, S. Šabanovic, and G. Fox, "Real-time, cloud-based object detection for unmanned aerial vehicles," in *Proceedings of the 1st IEEE International Conference on Robotic Computing (IRC)*, pp. 36–43, Taichung, Taiwan, April 2017.

[46] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016, preprint https://arxiv.org/abs/1506.02640

[47] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," preprint https://arxiv.org/abs/1612.08242.

[48] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: Single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, pp. 21–37, Springer, 2016.

[49] W. Li, H. Fu, L. Yu, and A. Cracknell, "Deep learning based oil palm tree detection and counting for high-resolution remote sensing images," *Remote Sensing*, vol. 9, no. 1, p. 22, 2017.

[50] S. W. Chen, S. S. Shivakumar, S. Dcunha et al., "Counting apples and oranges with deep learning: a data-driven approach," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 781–788, 2017.

[51] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proceedings of the 28th Annual Conference on Neural Information Processing Systems 2014, NIPS 2014*, pp. 487–495, December 2014.

[52] O. A. B. Penatti, K. Nogueira, and J. A. Dos Santos, "Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW 2015*, pp. 44–51, June 2015.

[53] F. Hu, G.-S. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14680–14707, 2015.

[54] A. Gangopadhyay, S. M. Tripathi, I. Jindal, and S. Raman, "Sacnn: dynamic scene classification using convolutional neural networks," preprint https://arxiv.org/abs/1502.05243.

[55] C. Hung, Z. Xu, and S. Sukkarieh, "Feature learning based approach for weed classification using high resolution aerial images from a digital camera mounted on a UAV," *Remote Sensing*, vol. 6, no. 12, pp. 12037–12054, 2014.

[56] J. Rebetez, H. F. Satizábal, M. Mota et al., "Augmenting a convolutional neural network with local histograms-a case study in crop classification from high-resolution uav imagery," in *Proceedings of the European Symposium on Artificial Neural Networks*, 2016.

[57] J. Delmerico, E. Mueggler, J. Nitsch, and D. Scaramuzza, "Active autonomous aerial exploration for ground robot path planning," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 664–671, 2017.

[58] J. Delmerico, A. Giusti, E. Mueggler, L. M. Gambardella, and D. Scaramuzza, ""on-the-spot training" for terrain classification in autonomous air-ground collaborative teams," in *Proceedings of the International Symposium on Experimental Robotics (ISER)*, EPFL-CONF-221506, 2016.

[59] T. Taisho, L. Enfu, T. Kanji, and S. Naotoshi, "Mining visual experience for fast cross-view UAV localization," in *Proceedings of the 8th Annual IEEE/SICE International Symposium on System Integration, SII 2015*, pp. 375–380, December 2015.

[60] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 5007–5015, June 2015.

[61] F. Aznar, M. Pujol, and R. Rizo, "Visual Navigation for UAV with Map References Using ConvNets," in *Advances in Artificial Intelligence*, vol. 9868 of *Lecture Notes in Computer Science*, pp. 13–22, Springer, 2016.

[62] G. J. Mendis, T. Randeny, J. Wei, and A. Madanayake, "Deep learning based doppler radar for micro UAS detection and classification," in *Proceedings of the MILCOM 2016 - 2016 IEEE Military Communications Conference (MILCOM)*, pp. 924–929, Baltimore, Md, USA, November 2016.

[63] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 528–535, Stockholm, Sweden, May 2016.

[64] T. Morito, O. Sugiyama, R. Kojima, and K. Nakadai, "Partially shared deep neural network in sound source separation and identification using a uav-embedded microphone array," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016*, pp. 1299–1304, October 2016.

[65] S. Jeon, J.-W. Shin, Y.-J. Lee, W.-H. Kim, Y. Kwon, and H.-Y. Yang, "Empirical study of drone sound detection in real-life environment with deep neural networks," preprint https://arxiv.org/abs/1701.05779.

[66] D. Maturana and S. Scherer, "3D convolutional neural networks for landing zone detection from LiDAR," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '15)*, pp. 3471–3478, IEEE, Washington, DC, USA, May 2015.

[67] Y. LeCun, B. E. Boser, J. S. Denker et al., "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, Ed., vol. 2, pp. 396–404, 1990.

[68] A. Ghaderi and V. Athitsos, "Selective unsupervised feature learning with convolutional neural network (S-CNN)," in *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2486–2490, December 2016.

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[70] H. Kim, D. Kim, S. Jung, J. Koo, J.-U. Shin, and H. Myung, "Development of a UAV-type jellyfish monitoring system using deep learning," in *Proceedings of the 12th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2015*, pp. 495–497, October 2015.

[71] N. V. Kim and M. A. Chervonenkis, "Situation control of unmanned aerial vehicles for road traffic monitoring," *Modern Applied Science*, vol. 9, no. 5, pp. 1–13, 2015.

[72] M. Bejiga, A. Zeggada, A. Nouffidj, and F. Melgani, "A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery," *Remote Sensing*, vol. 9, no. 2, p. 100, 2017.

[73] A. Sawarkar, V. Chaudhari, R. Chavan, V. Zope, A. Budale, and F. Kazi, "HMD vision-based teleoperating UGV and UAV for hostile environment using deep learning," CoRR abs/1609.04147. URL http://arxiv.org/abs/1609.04147.

[74] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1–9, Boston, Mass, USA, June 2015.

[75] The Technion – Israel Institute of Technology, "Technion aerial systems 2016," in *Journal Paper for AUVSI Student UAS Competition*, 2016.

[76] P. Santana, L. Correia, R. Mendonça, N. Alves, and J. Barata, "Tracking natural trails with swarm-based visual saliency," *Journal of Field Robotics*, vol. 30, no. 1, pp. 64–86, 2013.

Hindawi
Journal of Sensors
Volume 2017, Article ID 9374870, 12 pages
https://doi.org/10.1155/2017/9374870

Research Article

A Nonlocal Method with Modified Initial Cost and Multiple Weight for Stereo Matching

Shenyong Gao,[1,2] Haohao Ge,[3] Hua Zhang,[3] and Ying Zhang[2]

[1]*College of Electrical Engineering, Zhejiang University, Hangzhou 310027, China*
[2]*School of Information Engineering, Zhejiang University of Water Resources and Electric Power, Hangzhou 310018, China*
[3]*School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China*

Correspondence should be addressed to Shenyong Gao; gaosy@hdu.edu.cn

Received 7 April 2017; Revised 5 June 2017; Accepted 20 June 2017; Published 6 August 2017

Academic Editor: Wendy Flores-Fuentes

This paper presents a new nonlocal cost aggregation method for stereo matching. The minimum spanning tree (MST) employs color difference as the sole component to build the weight function, which often leads to failure in achieving satisfactory results in some boundary regions with similar color distributions. In this paper, a modified initial cost is used. The erroneous pixels are often caused by two pixels from object and background, which have similar color distribution. And then inner color correlation is employed as a new component of the weight function, which is determined to effectively eliminate them. Besides, the segmentation method of the tree structure is also improved. Thus, a more robust and reasonable tree structure is developed. The proposed method was tested on Middlebury datasets. As can be expected, experimental results show that the proposed method outperforms the classical nonlocal methods.

1. Introduction

Dense two-frame stereo matching is one of the most extensively researched topics in machine vision. Finding corresponding points in two or more images is the most important progress. After their disparities are computed, the results are used to distinguish the objects and background. Moreover, the depth information arises from the obtained disparity map. Scharstein and Szeliski [1] performed the following four steps:

(1) Cost computation
(2) Cost aggregation
(3) Disparity computation
(4) Disparity refinement

Additionally, they separated stereo matching algorithms into local methods and global methods. On the one hand, in local methods, they require cost aggregation, which ensures that the disparity between pixels is more accurate and specific than making the calculation with only one pixel. Therefore, in local methods, the support windows of cost aggregation for each pixel are significant. On the other hand, global methods construct a global energy function, and then the matching problem can be replaced by optimization. In these methods, a global energy function always consists of data and a smoothness item. The former measures the matching degree of the guidance image and the disparity function. However, the latter is capable of embodying the constraint of the definition model. An important problem for these methods, however, is to find the balance. It is different to obtain the perfect matching result between both measures. A number of global methods have been developed such as dynamic programming [2], graph cut [3], and belief propagation [4].

The semiglobal matching (SGM) algorithm by Hirschmüller [5] plays a good trade between matching accuracy and speed. SGM performs energy minimization along several 1D paths across the image and, thus, approximates the otherwise two-dimensional NP-complete energy minimization problem. However, high computational complexity and memory demand are a challenge for fast implementations. SGM can be implemented relatively efficiently by parallelization schemes. Real-time designs are possible and have been reported for

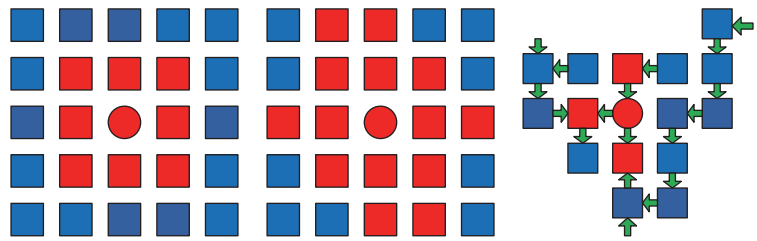


Figure 1: The support regions of cost aggregation. The red circle denotes the central pixel, and the red squares denote the pixels in the support region. The blue pixels are irrelevant. (a) Fixed support window; (b) cross-based support window; and (c) tree structure.

CPU and GPU systems [6]. There also exist some real-time embedded system designs, for example, on FPGA [7]. Schumacher and Greiner designed a higher data throughput FPGA architecture for SGM [8].

As for local methods, the problem in finding the correspondence of pixel p and pixel p' can be concluded as a similarity comparison of the two local patches, which exist around p and p', respectively [9]. Hence, the problem of finding the correspondence of two pixels is how to compute the cost value about two patches surrounded. Since then, it requires gathering the cost of each pixel during the cost aggregation procedure. Yoon and Kweon [10] proposed an adaptive support weight (ASW) method, which has higher matching accuracy but low efficiency. They use large support windows for robust cost aggregation which causes a huge computational burden [11] and fails to obtain satisfactory results on large planar surfaces.

For this reason, to obtain accurate results, the matching windows with an appropriate size and shape should be selected. However, the fixed windows method (shown in Figure 1(a)) is restrictive. It may result in incorrect matching in low-texture areas if the support windows are not large enough, and the windows break the boundaries between the object and background to influence the validity of the depth discontinuity regions [12].

To this end, many methods to construct matching windows have been proposed recently. For instance, Qu et al. [13] presented an algorithm that filters the inapposite pixels around the matching point by using the color similarity of the pixels around a central matching point. This algorithm finally acquires the appropriate pixels that construct the adaptive support windows, which are helpful to the matching point. Zhang et al. [14] also proposed a cross-based structure (Figure 1(b)) and constructed it in the form of adaptive support windows by comparing the color similarity around the adjacent pixels. Both methods calculate the disparity of pixels with the assistance of adaptive support windows, which make the operations more specific and suitable than the approaches using a predefined fixed-size window. These computations, however, are dependent on the construction of each support window. And the time consumption caused by cost aggregation still does not satisfy the real-time requirement. Therefore, Mei et al. [11] designed an accurate stereo matching system by using an accelerated CUDA implementation on the basis of the previous proposed methods, which significantly improved the efficiency of the algorithm under the help of hardware.

Recently, Yang [15] proposed a nonlocal cost aggregation (NLCA) method and then relied on it to perform tree-based filtering [16]. The NLCA algorithm is a novel cost aggregation method on a tree structure instead of using support windows. It also has been demonstrated to outperform the tradition of cost aggregation methods on support windows in terms of both speed and accuracy. In the NLCA algorithm, the nodes of the tree are all the image pixels, and the edges are all the edges between the nearest neighboring pixels. The similarity between any two pixels is decided by their shortest distance on the tree. All the pixels are connected to make a tree as shown in Figure 1(c), each node is aggregated only with its parents and children directly, and then every node on the tree makes a contribution to the final results. Hence, both the accuracy and the efficiency have been improved in this method. Nevertheless, this method does not perform well when the scene is composed of boundaries between object and background areas with similar color distribution because it considers color correlation as the only component of the weight function.

Mei et al. [17] proposed segment-tree cost aggregation (STCA) that segments the guidance image into several independent trees and then independent segment graphs are linked to form the segment-tree structure. In addition, they selected initial depth as a new component when computing the weight function. This method involves a new process; it leads to consistent scene segmentation; and only one judgement condition is adopted during the three-step image segmentation process. More recently, a cross-scale framework which unified aggregated based algorithms was also proposed [18]. With the proposed color-depth weight, Peng et al. [19] further iteratively rebuilt the tree to improve the matching efficiency in textureless regions. Besides, based on a minimum spanning tree, Pham et al. [20] proposed a robust nonlocal stereo matching algorithm that improves the performance of nonlocal approaches for outdoor driving images.

In this paper, we propose an improved nonlocal cost aggregation algorithm that modifies the original algorithm in both computational cost and aggregation. The additional vertical gradient will be used as one of the components

to calculate the initial cost of each pixel. We also employ a known function named *Gemen – McClure* [21] to deal with outliers. Furthermore, we add the inner correlations and mix them with color correlation. And then we compute the weight function with a mixture of both correlations together. Moreover, when segmenting the guidance image more reasonably is under consideration, we also try to provide a new segmentation method with brand.

We evaluate our proposed method on standard and extra Middlebury datasets and compare our method with ST and MST. Experimental results show that our method can achieve acceptable results when it is in the process of computing the accuracy of disparity, especially in some representative regions. The average number of erroneous pixels around discontinuous regions can be reduced efficiently while the disparities of flat regions become more stable. Compared with NLCA and STCA, a performance evaluation on Middlebury datasets shows that the proposed method has higher correct matching rate. In our method, the percentage of matching error declined to between 5% and 15%. Additionally, the computational cost of the new segmentation method can be ignored usually, while only the cost from the inner color correlation which was employed in our cost aggregation procedure also has a weak impact on the computational complexity. In this method, the computational complexity is the same as color correlation in terms of magnitude. Therefore, the total computational complexity retains the same magnitude as the STCA algorithm but slightly improves the result.

The main contribution of this paper is to improve the original nonlocal cost aggregation method with the following advantages:

(1) It has higher accuracy by adding the vertical gradient as one of the components in the process of cost computation. It is proved to be better in some discontinuous areas. Its initial value is more stable with the *Gemen – McClure* function.

(2) Inner color correlation is employed in the computation of the weight function to make constructing a tree structure more robust and reasonable.

(3) The segmentation method of STCA is improved and it achieves a better result. Moreover, irrelevant pixels contribute less to each other.

The rest of this paper is organized as follows. In Section 2, we briefly introduce related work on local methods. Then, our proposed improved method is described in Section 3. Section 4 describes and analyzes the experimental results, and Section 5 discusses setting the parameters. Finally, we provide conclusion in Section 6.

2. Related Work

Cost aggregation, which consists of constructing support regions and aggregating the disparity for each pixel within those support regions, is one of the important processes in stereo matching. The efficiency and effectiveness rely on the used aggregation method; therefore, they are different from each other. In this section, we review the related work on cost aggregation, especially on the traditional local methods and nonlocal cost aggregation methods based on tree structure.

2.1. The Traditional Local Methods. The stationary support windows with a stationary weight for each pixel are used by the simplest local method of cost aggregation. However, note that this method fails in many specific regions, including occlusion regions and low-textured areas. Furthermore, this method is unable to achieve decent robustness and its matching accuracy falls well short of the ideal result. To resolve this dilemma, there are usually two approaches: (1) make the fixed support window alterable using shiftable windows, multiple windows [22], or variable windows [23, 24] or (2) concentrate on varying the weights to achieve excellent matching accuracy.

The algorithms based on adaptive weight consider every pixel in the support windows as a unique unit and calculate weight for the central point by themselves. The pixel will have a dramatic effect on the final result only if there is a cost value which is similar to the central point. Hence, every pixel is able to receive proper contributions from all the other neighboring pixels. This approach blurs the boundaries between local methods and global methods due to its remarkable accuracy and the obvious increase of computational cost.

Yoon and Kweon [10] first proposed an adaptive weight method and Gu et al. [25] further enhanced their method by introducing rank transform and disparity refinement. Tombari et al. [26] obtained the cost value after using the Meanshift [27] algorithm to segment the image, which revises ASW algorithm performance calamitously in repetitive texture regions and discontinuous regions. Hosni et al. [28] performed connectivity by using the geodesic distance transform; nevertheless, the computational efficiency of their strategy still has similar efficiency to others.

2.2. Nonlocal Cost Aggregation Based on Tree Structure. Even though great progress has been made in local algorithms, they still aggregate pixels into local regions. As mentioned above, a nonlocal cost aggregation (NLCA) method has been proposed that breaks through the boundaries of local and global methods. This method transforms the guidance images into a graph and constructs a tree structure so that all the image pixels become the nodes of the tree. Before aggregating, a minimum spanning tree (MST) must be constructed. The nodes attached to edges with the lowest weights (calculated by differences in color distribution process) are connected to one another until all the pixels are finally included in the tree. It is an important step, that is, to convert the guidance image into a cost tree after all the pixels have been connected. Then, the whole process is separated into three steps:

(1) Traversing the cost tree

(2) Assigning an appropriate value to each node

(3) Calculating each node's disparity level with its relatives

After constructing the tree structure, the aggregation costs can be efficiently computed by executing a tree filter,

which traces the MST from the leaf nodes to the root nodes and from the root nodes to the leaf nodes. Hence, the aggregation is complete after only two trees traverse, and then any pixel receives proper contributions from every node in the constructed tree (more or less). Based on the tree structure, some effective disparity refinement methods are proposed as follows.

Chen et al. [29] improved the NLCA by adding depth information in the weight function, which enhances the effect of regions around the border. Mei et al. [17] proposed a new segment-tree (ST) method that divides the construction of the tree structure into two rounds. In the first round, it combines subtrees in the homogeneous regions, and it also keeps those subtrees that belong to different regions separate from each other if they break the predefined equation. In the second round, to ensure that the different regions have little impact on each other, it combines the remaining subtrees with a penalty value. However, the segmentation performance is not robust because the segmentation equation is extremely ordinary. Therefore, the performance of this method falls short of expectations.

3. Our Proposed Method

Our work is directly motivated by the above two nonlocal cost aggregation methods. We further improve these methods during cost computation and tree construction process, respectively. We include the vertical gradient as a new component in the cost computation. On the other hand, due to its stability and versatility, inner color correlation is employed instead of using a single color component. Moreover, we modify the structure of the segment tree, which improves its validity and robustness. In this section, we divide our methods into five parts as follows:

(1) Cost computation

(2) Tree construction

(3) Cost aggregation

(4) Disparity computation and refinement

(5) Computation complexity

More details can be found in the following subsections.

3.1. Cost Computation. Traditional nonlocal methods are considered to employ the truncated absolute difference of the color and the horizontal gradient as the initial cost. However, the performance of this cost measurement is unstable in marginal areas. Hence, we decided to employ the vertical gradient to make the cost measurement reveal more detailed description of the reference images. We compute the individual cost values $C_{AD}(p,d)$, $C_{GD_x}(p,d)$, and $C_{GD_y}(p,d)$ primarily for a pixel $p = (x,y)$ in the guidance image with a disparity level d. Let I_i denote RGB color component. $C_{AD}(p,d)$ is defined as the average absolute difference of p and its relevant pixel pd in the *RGB* channel (as shown in (1)):

$$C_{AD}(p,d) = \frac{1}{3}\sum_{i=R,G,B}\left|I_i^{\text{Left}}(p) - I_i^{\text{Right}}(pd)\right|. \quad (1)$$

Then, we compute the gradient cost values $C_{GD_x}(p,d)$ and $C_{GD_y}(p,d)$ using (2) and (3), respectively. The equations can be designed as follows:

$$C_{GD_x}(p,d) = \left|\nabla_x I^{\text{Left}}(p) - \nabla_x I^{\text{Right}}(pd)\right|, \quad (2)$$

$$C_{GD_y}(p,d) = \left|\nabla_y I^{\text{Left}}(p) - \nabla_y I^{\text{Right}}(pd)\right|. \quad (3)$$

In addition, our proposed method works pretty well when truncated values are used for discarding the extremum of the initial cost. However, the improvement this method yields is not obvious. Therefore, we employ the *Gemen – McClure* function to handle the exception values as shown in

$$C_{\text{tran}_c} = \frac{C_{\text{init}_c}^2}{C_{\text{init}_c}^2 + \varepsilon_c^2},$$
$$C_{\text{tran}_g} = \frac{C_{\text{init}_g}^2}{C_{\text{init}_g}^2 + \varepsilon_g^2}, \quad (4)$$

where C_{tran_c} and C_{init_c} denote the final and initial cost values of the color, respectively. And then let C_{tran_g} and C_{init_g} denote the final and initial cost values of the gradient, respectively. In addition, ε_c and ε_g are user-specified parameters for adjustment. The former is related to the color adjustment and the latter is related to adjustments on behalf of the gradient. ε_c is set to 7, and ε_g is set to 2 in our experiments. The effect of this function declines smoothly when the initial cost reaches a certain value and the final cost value converges to 1 under the control of ε. So, by using three cost components as mentioned above together, the final initial cost value can be expressed as the following equation:

$$C(p,d) = \alpha \cdot C_{AD}(p,d) + \beta \cdot C_{GD_y}(p,d) + (1-\alpha-\beta)\cdot C_{GD_x}(p,d), \quad (5)$$

where α and β are the weights for each component. Figure 2 shows a comparison between the traditional cost computation and our method, which demonstrates the improvement after adding the discontinuous regions.

3.2. Tree Construction. According to Yang's contribution [15], we treat the guidance image I as a graph $G = (V,E)$ in this paper, where each node denotes the corresponding pixel in I and each edge represents the weight that connects two neighboring nodes. Accordingly, a flow chart shows how to construct our tree structure in Figure 3.

The weight W_e of an edge e is determined with its conjoint nodes p and q; this process can be described as follows:

$$W_e = \theta_{\text{In}} \cdot \left|I_{\text{In}}(p) - I_{\text{In}}(q)\right| + (1-\theta_{\text{In}}) \cdot \left|I(p) - I(q)\right|, \quad (6)$$

where θ_{In} is the predefined weight and is set to 0.2 in this paper. I_{In} denotes the inner color correlation, which is shown in

$$I_{\text{In}} = \left[I_{RtG}, I_{GtB}, I_{BtR}\right]. \quad (7)$$

(a) (b) (c) (d)

Figure 2: Cost measure comparison. (a) The input image; the black boxes express the target areas. (b) Insets of target area; (c) and (d) denote the results of the traditional cost measure and our method, respectively.

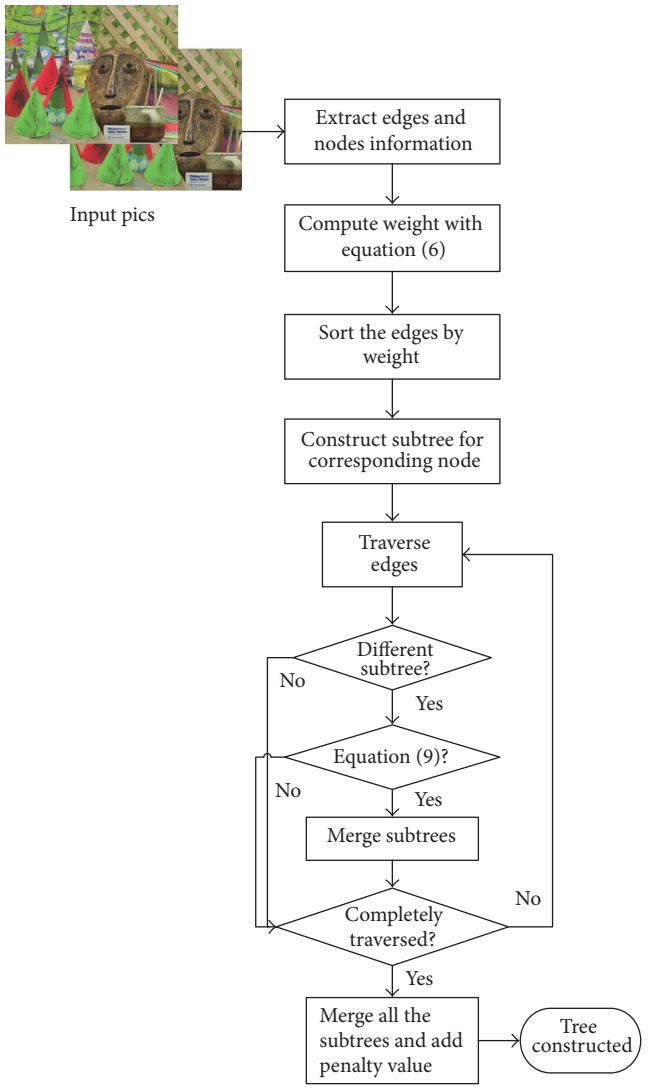


Figure 3: A flow chart showing the tree construction steps.

The components with a pixel $p(x, y)$ of I_{In} are specifically expressed as follows:

$$\begin{aligned} I_{RtG} &= I_R(x, y) - I_G(x, y), \\ I_{GtB} &= I_G(x, y) - I_B(x, y), \\ I_{BtR} &= I_B(x, y) - I_R(x, y). \end{aligned} \tag{8}$$

Then, the edges in E are sorted in an ascending order according to their weights. And then the subtrees are created for each node in V. Every node p has one subtree T_p. Finally, we traverse the sequence of edges, and then the subtrees T_p and T_q are merged into bigger groups only if the edge weight should satisfy

$$\begin{aligned} w_{e_i} \le \min &\left(\left(\max\left(w_{e_{T_p}}\right) + \frac{\tau}{|T_p|}\right),\right. \\ &\left.\left(\max\left(w_{e_{T_q}}\right) + \frac{\tau}{|T_q|}\right)\right), \quad w_{e_i} < w_{e_{\text{Avg}}}, \\ w_{e_i} \le \min &\left(\left(w_{e_{\text{Avg}}} + \frac{\tau}{|T_p|}\right), \left(w_{e_{\text{Avg}}} + \frac{\tau}{|T_q|}\right)\right), \\ &w_{e_i} \ge w_{e_{\text{Avg}}}, \end{aligned} \tag{9}$$

where w_{e_i} denotes the weight of edge e_i that connects the two nodes p and q. $w_{e_{T_p}}$ and $w_{e_{T_q}}$ denote the weight sequence of edges in subtrees T_p and T_q, respectively. $w_{e_{\text{Avg}}}$ denotes the average weight of all the edges. τ is a predefined parameter. We employ $w_{e_{\text{Avg}}}$ and divide the equation into two cases, which guarantees that the constraint condition will not be lost in those boundary regions with high weights and makes the segmentation of the tree more precise and robust.

After traversing all the edges, a large number of subtrees are merged with each other and changed into some new

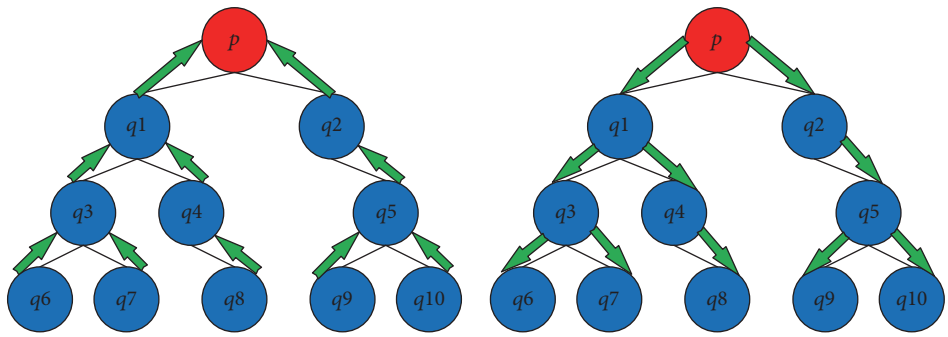


Figure 4: The tree filter for cost aggregation; p denotes the matching pixel. (a) Leaves to root pass; (b) root to leaves pass.

subtrees that have a bigger structure but are small in quantity. Note that the integrated graph I has been segmented into several smaller pieces. We then traverse the edges once again and merge the rest of the subtrees. Meanwhile, we add a penalty value to the weight of edges to ensure that boundary regions do not interact with each other. Finally, all the nodes are constructed into a segment tree T, and there is only one path between any two nodes in T. The segment tree T is used in aggregating the final cost value.

3.3. Cost Aggregation. The nonlocal cost aggregation method is a linear-time method in which the computational complexity is extremely low. We employ a weighting function $S(p, q)$ to compute the contribution from pixel q to p; its function is decided as follows:

$$S(p,q) = \exp\left(-\frac{D(p,q)}{\sigma}\right), \tag{10}$$

where $D(p, q)$ denotes the distance from p to q in the tree structure that relates to (6) and σ is a predefined parameter for adjustment. Because of the otherness of our initial matching cost, σ is set to 0.08 in our experiments, and the setting of σ will be discussed in Section 5. Let $C_d(p)$ denote the cost value for pixel p at disparity level d; the aggregated cost value $C_d^A(p)$ is computed as follows:

$$C_d^A(p) = \sum_{q \in I} S(p,q) \cdot C_d(q), \tag{11}$$

where I denotes the whole graph and therefore $C_d^A(p)$ is aggregated with all the nodes in the graph I. Yang employs a tree filter to compute the cost aggregation that traverses the tree structure from leaves to root and root to leaves [15], as shown in Figure 4. A node is affected by all the other nodes in the segment tree T but aggregates with only its children and parents. For a pixel p, the aggregated value is calculated as follows:

$$C_d^{A\uparrow}(p) = \sum_{q \in \text{Child}(p)} S(p,q) \cdot C_d^{A\uparrow}(q), \tag{12}$$

where the set Child(p) contains the children of node p, and the computation for the node will be complete only if its child nodes have already been computed. Therefore, all the nodes have been aggregated by their low-grade nodes. Then, the tree structure is traversed from root to leaves, and the final aggregated cost value of pixel p is computed as follows:

$$\begin{aligned} C_d^A(p) = {} & S(\text{Parent}(p), p) \cdot C_d^A(\text{Parent}(p)) \\ & + \left(1 - S^2(\text{Parent}(p), p) \cdot C_d^{A\uparrow}(p)\right), \end{aligned} \tag{13}$$

where Parent(p) denotes the parent node of pixel p. After that, all the pixels eventually obtain a reliable aggregated cost. The complexity of computation is $O(n \cdot d)$, where n denotes the number of pixels in the guidance image and d denotes the disparity level.

3.4. Disparity Computation and Refinement. This subsection describes the universal winner-takes-all strategy, which is employed to seek the appropriate disparity level. And it carries the lowest matching cost, as shown in

$$D(p) = \arg \min_{d \in \text{dislevel}} \left(C_d^A(p)\right), \tag{14}$$

where set dislevel denotes the disparity level.

We employ a tree structure to refine the coarse disparity map. First, we use the left and right images as guidance images, respectively. And the tree filter is executed twice, receiving two corresponding disparity maps. Then, we employ left and right consistency checks to mark the mismatched pixels and store them in set P_{mis}. For the left disparity map D, the cost value $C_{\text{new}}(p, d)$ for each pixel p at each disparity d is recalculated as follows:

$$C_{\text{new}}(p,d) = \begin{cases} 0, & p \in P_{\text{mis}} \\ |d - D(p)|, & \text{else}, \end{cases} \tag{15}$$

where $D(p)$ denotes the initial disparity of pixel p. This method uses the tree structure mentioned above to execute the tree filter, and the process of creating a new mathematical model has no extra computation cost. The total running time is taken by recalculating the cost value and executing the tree filter. Furthermore, all the pixels with unstable disparity are marked as mismatch pixels, and the cost value of each

Table 1: Comparison of computational complexity.

Process	Complexity of computation	
	Tree construction	Cost aggregation
MST	$O(e + n + 2e \cdot \log_2 n)$	$O(n \cdot d)$
ST-1	$O(2 \cdot (e + e\alpha(e)))$	$O(n \cdot d)$
ST-2	$O(4 \cdot (e + e\alpha(e)))$	$O(2n \cdot d)$
Our proposed method	$O(2 \cdot (2e + e\alpha(e)))$	$O(n \cdot d)$

disparity level is set to zero. Only pixels with stable and precise disparity participate in aggregating the new cost value. The mismatched pixels achieve their final disparity value through the propagation of stable pixels afterwards.

This postprocessing technique has two advantages. A great advantage is that it is a nonlocal method and the whole stable and precise pixels contribute to the mismatched pixels. Another great advantage is that the tree structure is ready-made and the additional computational cost is negligible. The computation of the tree filter has an extremely low cost as well.

Moreover, we can further refine the disparity by means of (9) as mentioned above. Here, this equation can be regarded as a standard method for image segmentation. By comparing the boundaries of the disparity map with those of other segmented maps to mark the blurry regions, we can execute the tree filter again to obtain a disparity map with higher precision and more elaborate boundaries.

3.5. Complexity of Computation. We mainly analyze the computational complexity of tree construction and the cost aggregation in this section. Let n denote the number of pixels in image I and e denote the number of edges. The computation of tree construction in MST concentrates on the calculation of edge weights and node connections. The calculation of edge weight is $O(e)$. The pixels connections are divided into *find* and *connect* operations. The *find* operation requires $O(2e \cdot \log_2 n)$, and the complexity of the *connect* operation is determined only by n, so the total computation of tree construction in MST is $O(e + n + 2e \cdot \log_2 n)$.

As shown in Table 1, compared with MST, ST-1 must execute more *find* operations due to the constraint condition. So, the complexity of tree construction in ST-1 is $O(2 \cdot (e + e\alpha(e)))$, but in ST-2, it is $O(4 \cdot (e + e\alpha(e)))$ according to [17]. Therefore, the computational complexity of tree construction in our proposed method is $O(2 \cdot (2e + e\alpha(e)))$, which is slightly larger than ST-1 due to the multiple components of weight function. As for cost aggregation, let d denote the disparity level. Therefore, it is ordinary to deduce the computational complexity of aggregation. The cost aggregation computation complexity of MST, ST-1, and our proposed method is $O(n \cdot d)$ while ST-2 is 2 times slower. Our proposed method requires more computations than some nonlocal cost aggregation methods but only on an extremely small scale.

4. Experimental Results

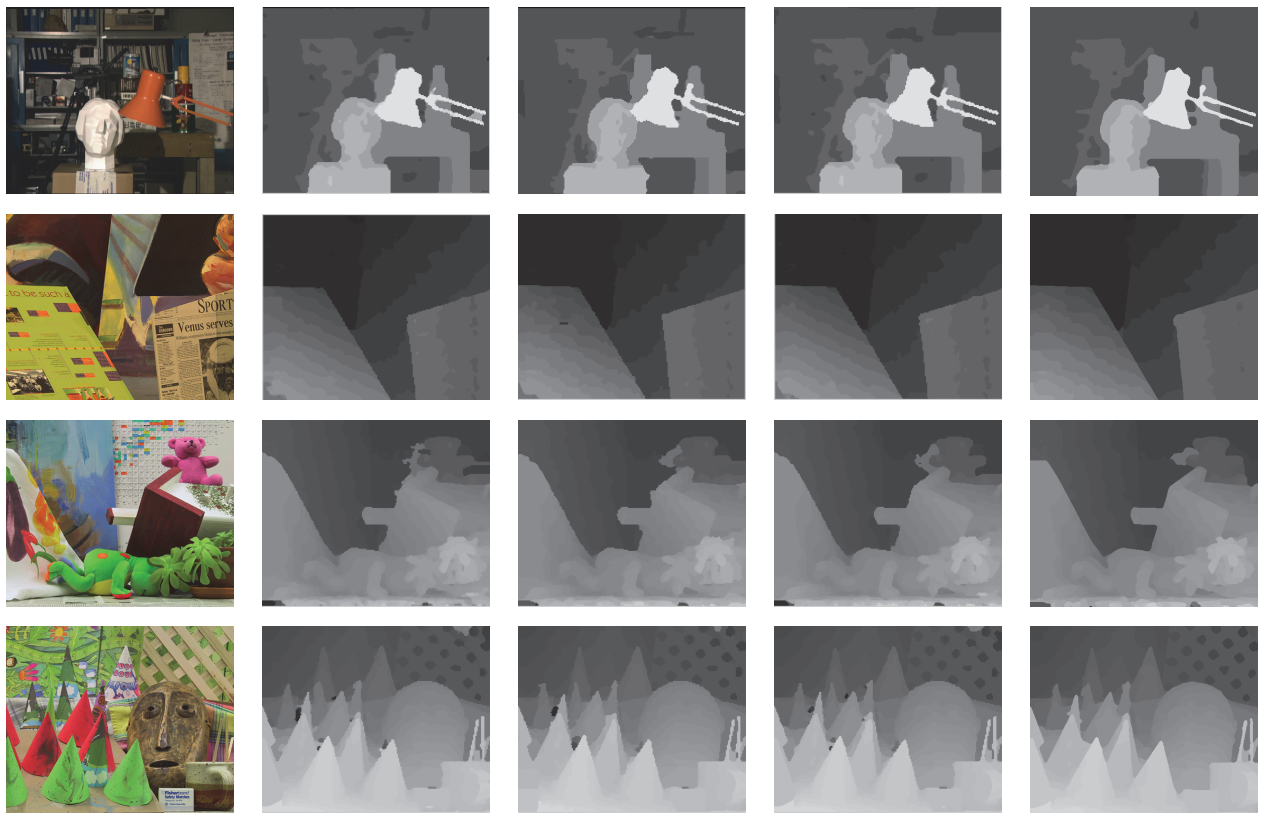
This section compares three mature nonlocal cost aggregation methods (MST [15], ST-1, and ST-2 [17]) with our proposed method. We tested our method using four standard Middlebury datasets [30] (Tsukuba, Venus, Teddy, and Cones). The MST and ST methods use an AD-Gradient measure [31] as the matching cost, while our proposed method employs the improved AD-Gradient method mentioned in Section 3. Moreover, the initial disparity for all the methods is computed by a WTA strategy. Finally, the postprocessing for each method involves nonlocal disparity refinement using their own tree structures. The parameters for our proposed method are defined as follows: $\alpha = \beta = 0.2$, $\varepsilon_c = 7$, $\varepsilon_g = 2$, $\theta_{\text{In}} = 0.2$, $\sigma = 0.08$, and $\tau = 1200$, and the parameters of MST and ST methods follow the relevant cited papers. The performance is tested on a PC with a 3.40 GHz CPU and 4 GB of memory.

Figure 5 shows the results of the four standard Middlebury datasets with these methods described above. The performance of ST-2 is better than that of ST-1 and MST in most typical regions when the boundaries of ST-2 are quite expressive. Our proposed methods' performance on the areas around the eaves near Teddy (the occluded regions) is particularly excellent. On Tsukuba, the angle of the table, where the foreground objects and the background have similar color contributions, is resolved faultlessly. In addition, the results of our proposed method are more satisfactory than the results of ST-2; the boundaries of the disparity maps are extremely smooth and precise. The typically tough regions such as the discontinuity regions and low-texture areas both achieve a good performance. However, our proposed method also fails in some regions, especially in the areas around the cones in the Cones datasets. The inner pixels of the cones contribute too much to the mismatch of the pixels outside, and the areas between any two cones do not achieve desirable results. The regions between the lamp and the table in Tsukuba are affected by various regions and, finally, obtain incorrect results.

More intuitive results are shown in Table 2. ST-1 is slightly better than MST, while the performance of ST-2 is better than both. Moreover, our proposed method obtains the best performance among these four algorithms. Compared with three classical methods, the number of erroneous pixels is reduced efficiently to between 5% and 15%.

We further tested 16 extra Middlebury datasets. The quantitative evaluation results are shown in Table 3. Only nonoccluded regions are evaluated in this table. First, ST-1 has the worst average rank. However, the average ranks are nearly equal between ST-2 and MST. Nevertheless, the average percentages of erroneous pixels in the three nonlocal methods are extremely close to one another. Besides, our proposed method achieves a tremendous advance, whether to compare the average percentage of erroneous pixels or the average rank. The percentages of erroneous pixels decline distinctly in *Baby*3, *Lampshade*1, and *Laundry*. However, the performance of some images (*Baby*1, *Baby*2, *Books*, *clothes*2, and *Moebius*) exhibits negative growth.

We selected four representative images from the extra 16 datasets (*Baby*3, *Flowerpots*, *Lampshade*1, and *Wood*1) to show the superiority of our proposed method through a visual comparison. The results are shown in Figure 6. Compared to the other nonlocal methods, our proposed



(a) Guidance image (b) MST (c) ST-1 (d) ST-2 (e) Proposed

Figure 5: The final disparity maps of the four most common images in the standard Middlebury datasets. (a) denotes the guidance images. From top to bottom, these are Tsukuba, Venus, Teddy, and Cones. The subfigures (b) to (e) show the disparity maps computed by different nonlocal methods. (b) shows the results of MST [15]; (c) and (d) show the results of the two segment-tree cost aggregations [17], respectively, and (e) shows the results of our proposed method.

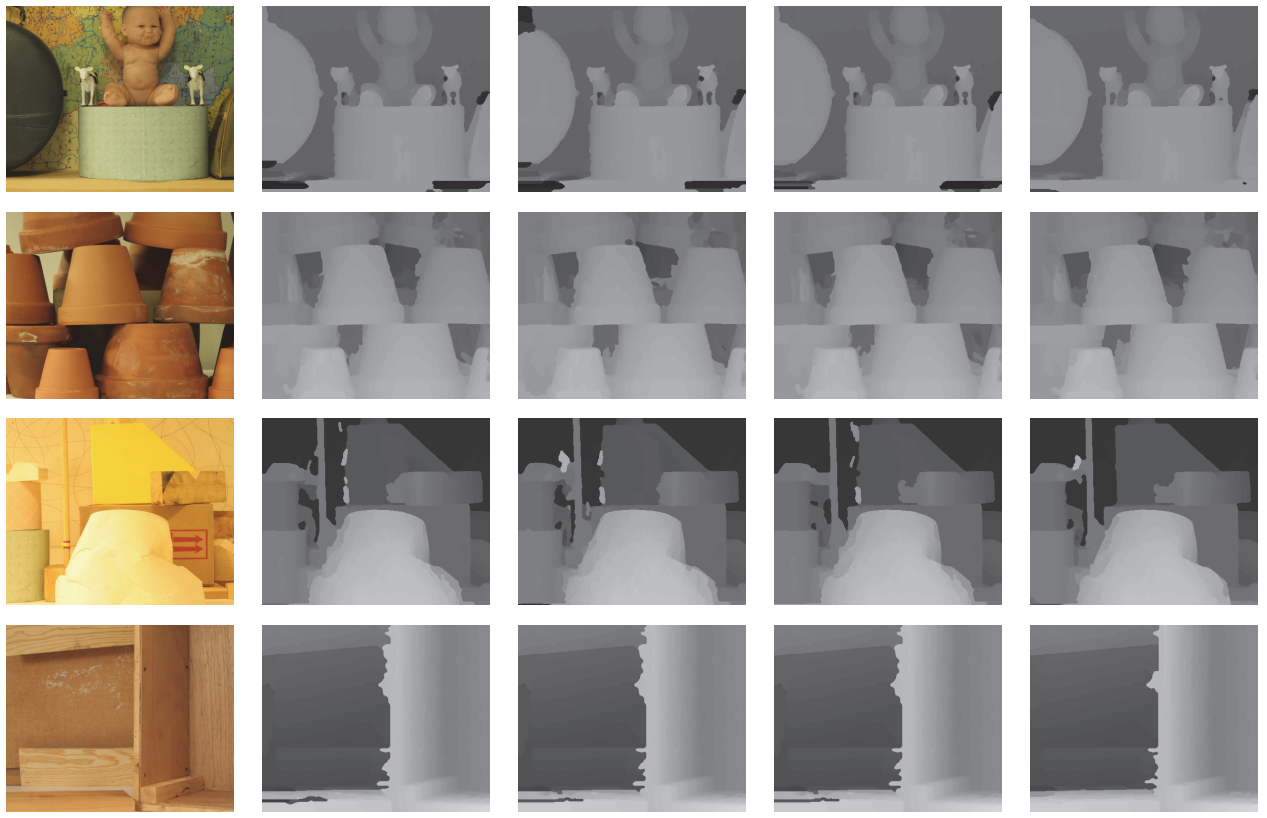
Table 2: Comparison of the four nonlocal algorithms (MST [15], ST-1 [17], ST-2 [17], and the proposed method) with Middlebury datasets and the standard of benchmark. The error threshold is set to 1 and three regions (nonocc, all, and disc) are used to evaluate the performance of the methods. Our proposed method exhibits the best accuracy in every region.

Algorithm	Avg.error	Tsukuba			Venus			Teddy			Cones		
		nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc
MST	5.73	1.50	2.18	8.02	0.42	0.85	5.02	5.95	10.89	14.15	3.14	8.68	7.94
ST-1	5.66	1.73	2.52	9.22	0.47	0.71	4.56	6.11	10.88	14.53	2.47	8.28	7.11
ST-2	5.18	1.35	2.00	7.29	0.42	0.69	5.27	5.17	9.95	12.95	2.49	7.90	6.62
Proposed	4.92	1.34	1.77	7.14	0.44	0.64	4.49	5.03	9.57	12.86	2.12	7.24	6.29

method achieves superior results, resulting in a more accurate disparity map and more reliable boundaries.

In *Lampshade*1, the results are adversely affected by illumination. Although other methods fail to detect the authentic boundaries, our method produces a better result. For example, the boundaries of the yellow trapezoid block are extremely close to the ground-truth map. As for *Wood*1, nearly the entire image contributes a similar color intensity. Therefore, it is crucial to calculate a rational result from the discontinuous regions. Unfortunately, all the other methods fail to detect clear boundaries on these datasets. However, the percentage of erroneous pixels declined to 2.49% by using our proposed method, which improves on the other nonlocal methods.

We mentioned the computational complexity in Section 3.5. In this section, we test 4 datasets and the average time consumption of each nonlocal method. The results are listed in Table 4. Most of the time is consumed during tree construction and tree filter requires only a slight amount of time. Moreover, MST is the shortest among the four methods, while our proposed method is a bit shorter than ST-2. The superiority of the proposed improved method over MST, ST-1, and ST-2 methods is demonstrated on experimental results (Tables 2 and 3, Figures 5 and 6). Moreover, in contrast to



(a) Guidance image (b) MST (c) ST-1 (d) ST-2 (e) Proposed

Figure 6: The final disparity maps of the extra Middlebury datasets. Four representative images were selected to show the superiority of our proposed method. (a) denotes the guidance images. From top to bottom, these are Baby3, Flowerpots, Lampshade1, and Laundry. Subfigures (b) to (e) show the disparity maps computed by different nonlocal methods. (b) shows the results of MST [15]; (c) and (d) show the results of the two segment-tree cost aggregations [17], respectively, and (e) shows the results of our proposed method.

Table 3: The comparison of the four nonlocal algorithms (MST [15], ST-1, ST-2 [17], and the proposed method) with 16 extra Middlebury datasets. The error threshold is set to 1 and only nonoccluded regions are used to evaluate the performance of the methods.

Data	MST	ST-1	ST-2	Proposed
Aloe	8.41_3	9.50_4	8.37_2	7.42_1
Art	13.96_2	14.75_4	13.99_3	12.83_1
Baby1	4.67_3	4.98_4	4.52_1	4.58_2
Baby2	7.93_2	9.1_4	7.50_1	8.53_3
Baby3	6.44_2	6.82_4	6.52_3	3.93_1
Books	6.10_1	6.29_3	6.19_2	6.82_4
Cloth2	2.64_3	2.84_4	2.58_1	2.59_2
Cloth3	1.69_3	2.22_4	1.65_2	1.28_1
Dolls	5.11_3	5.07_2	5.46_4	4.64_1
Flowerpots	9.96_4	9.81_2	9.86_3	9.17_1
Lampshade1	8.56_3	8.43_2	8.82_4	6.80_1
Laundry	16.54_2	16.63_3	16.77_4	14.18_1
Midd1	28.03_4	23.41_2	24.46_3	22.05_1
Moebius	8.66_2	9.34_4	8.35_1	8.91_3
Reindeer	8.99_3	9.15_4	8.68_2	7.53_1
Wood1	4.05_3	4.75_4	3.91_2	2.49_1
Avg.Error	8.88	8.94	8.60	7.74
Avg.Rank	2.69	3.38	2.38	1.56

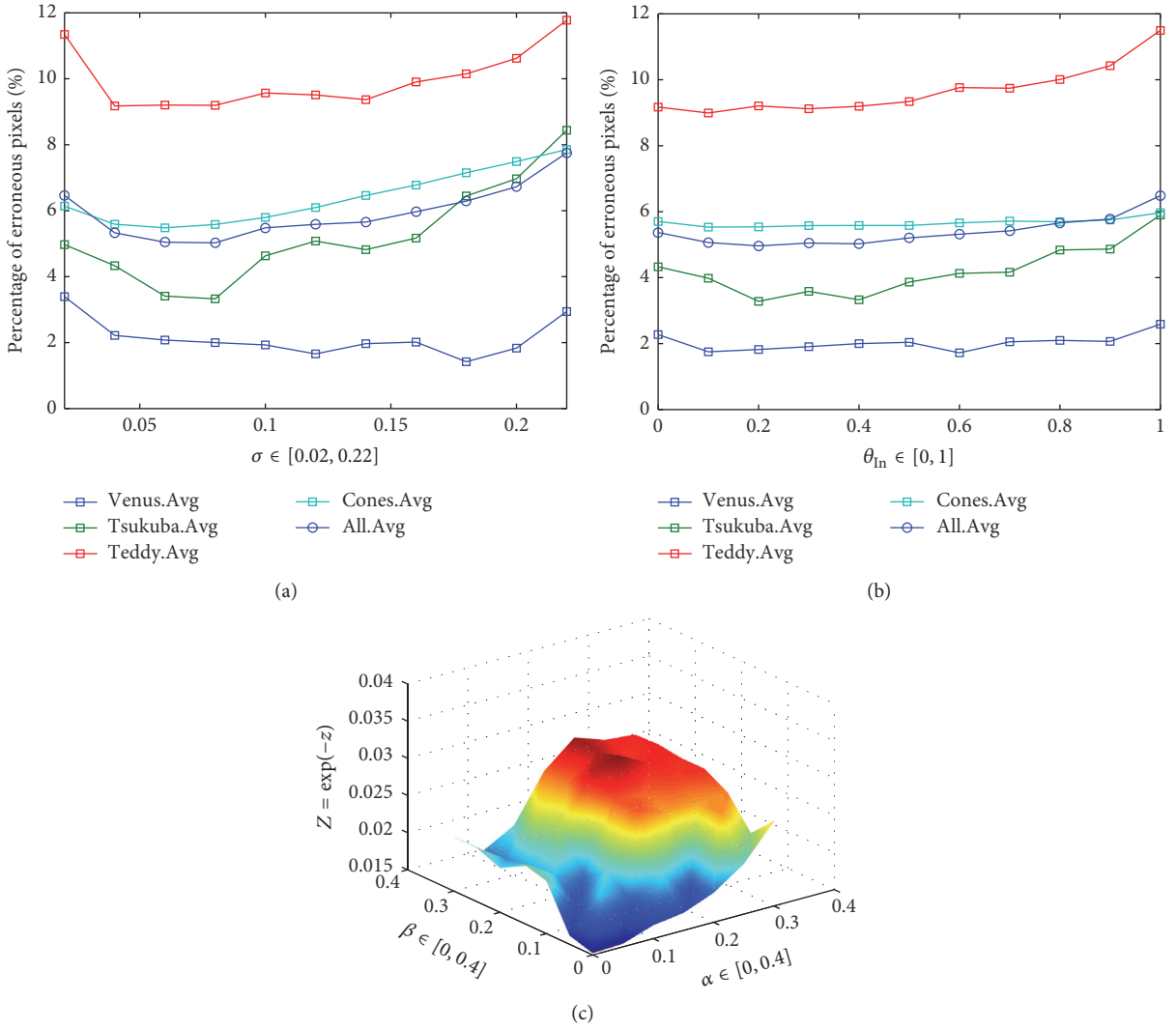


Figure 7: Parameter sensitivity analysis of our experiments. Four standard datasets (Tsukuba, Venus, Teddy, and Cones) were used in this experiment. (a) is a line chart representing the percentage of error pixels as parameter σ increases from [0.02 to 0.22], whereas (b) is a line chart representing the changes as parameter θ_{In} increases from [0 to 1]. Avg denotes the average percentage of erroneous pixels for three evaluation regions (nonocc, all, and disc), and All denotes the four standard datasets. (c) represents the average number of erroneous pixels from the four standard datasets using different weights for the initial components; an exponential function is employed to make the results more intuitive; $\alpha \in [0, 0.4]$ denotes the weight of color cost and $\beta \in [0, 0.4]$ denotes the weight of the vertical gradient cost.

MST and ST-1, the overall runtime cost of our proposed method does not increase obviously and is even shorter than ST-2. In contrast to the color-gradient based matching cost computation method proposed by Rhemann et al. [31], our method also has higher accuracy.

5. Parameter Setting

Several parameters are used in our proposed method. ε_c and ε_g are user-specified parameters used for adjustment in (4). They follow the truncated value in [31] while the predefined parameter $\tau = 1200$ in the tree construction follows the settings of the segment-tree [17] method. In this section, we discuss the rationale and sensitivity of the remaining four parameters, the weights for each component (α and β) in the initial computation, the predefined weight of inner color correlation (θ_{In}) in tree construction, and the adjustment value (σ) of the weight function.

Table 4: Average time consumption for each nonlocal method with 4 Middlebury datasets.

Process	Overall runtime (seconds)			
	MST	ST-1	ST-2	Proposed
Tree construction	0.870	0.894	1.740	1.360
Cost aggregation	0.108	0.110	0.218	0.108
Whole process	0.978	1.004	1.958	1.468

First, we test the adjustment value (σ) of (10). The results are shown in Figure 7(a). When $\sigma \in [0.04, 0.14]$, the

experimental results from most of the images are extremely low and vary slightly. In contrast, the erroneous pixels decline to a minimum when $\sigma \in [0.06, 0.08]$, which is due to the variation in the initial cost value. We employ the *Geman – McClure* function to protect the initial cost value from the encroachment of extremum, and the initial cost value converges to 1. With the adjustment of the initial cost value, a parameter σ is required to be adjusted accordingly, or disparity boundaries will be unclear and foreground objects will be confused with background.

As for the weight of the inner color correlation θ_{In}, the parameter range of this experiment is 0 to 1. More details are shown in Figure 7(b). The percentage of erroneous pixels increases significantly when the parameter $\theta_{\text{In}} \in [0.2, 0.4]$. The experimental results show that employing inner color correlation is obviously reasonable but the parameter θ_{In} should be confined to 0.5 or below.

Figure 7(c) evaluates the sensitivity of the initial component weights α and β with four original Middlebury datasets, to clarify that the final results (percentage of erroneous pixels) are processed by an exponential function. The figure shows that the algorithm achieves its best performance when the parameters α and $\beta \in [0.15, 0.3]$. The range of the parameters that achieve dramatic performance is much larger than the original nonlocal methods. And Figure 7 further demonstrates that employing the *Geman – McClure* function helps to resolve the errors caused by outliers more effectively and robustly than the methods described above which use truncated values.

6. Conclusion

In this paper, our work is directly motivated by two original algorithms [15, 17]. We propose an improved nonlocal cost aggregation algorithm based on them. The proposed method is developed with modified initial cost and multiple weight for stereo matching, which modifies the original algorithm in both computational cost and aggregation. Our method has some advantages. First, it has higher accuracy by adding the vertical gradient as one of the components in the process of cost computation. Particularly, the performance near some discontinuous areas is much better than that of other methods. Second, due to its stability and versatility, inner color correlation is employed instead of using a single color component. Thus, it makes constructing a tree structure more robust and reasonable. Besides, we modify the structure of the segment tree.

The performance was tested on a PC with a 3.40 GHz CPU and 4 GB of memory. The proposed method was evaluated on Middlebury datasets. The experimental results verified that our proposed method could achieve better accuracy with a minor cost of increased execution time. In the near future, we would like to focus on more novel tree structures. And we will continue to study nonlocal methods and image segmentation, proposing new ideas to resolve the issues mentioned above.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61471150), the International Cooperation and Exchange of the National Natural Science Foundation of China (Grant no. 2014DFA12040), and Zhejiang Provincial Natural Science Foundation of China (Grant no. LY13F020033).

References

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1–3, pp. 7–42, 2002.

[2] C. Lei, J. Selzer, and Y.-H. Yang, "Region-tree based stereo using dynamic programming optimization," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2006*, pp. 2378–2385, June 2006.

[3] L. Hong and G. Chen, "Segment-based stereo matching using graph cuts," in *Proceedings of the IEEE Computer Society Conference on Computer Vision & Pattern Recognition*, pp. 74–81, IEEE, 2004.

[4] Q. Yang, L. Wang, and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 1458–1465, San Francisco, Calif, USA, June 2010.

[5] H. Hirschmüller, "Accurate and efficient stereo processing by Semi-Global Matching and Mutual Information," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, pp. 807–814, June 2005.

[6] S. K. Gehrig and C. Rabe, "Real-time semi-global matching on the CPU," in *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, CVPRW 2010*, pp. 85–92, San Francisco, CA, USA, June 2010.

[7] S. Gehrig, F. Eberli, and T. Meyer, "A real-time low-power stereo vision engine using semi-global matching," in *Proceedings of the the 7th International Conference on Computer Vision Systems*, M. Fritz, B. Schiele, and P. H. Justus, Eds., vol. 5815 of *LNCS*, pp. 134–143, Springer, Berlin, Germany, 2009.

[8] F. Schumacher and T. Greiner, "Matching cost computation algorithm and high speed FPGA architecture for high quality real-time Semi global matching stereo vision for road scenes," in *Proceedings of the 2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, pp. 3064–3069, Quingdao, China, October 2014.

[9] F. Cheng, H. Zhang, M. Sun, and D. Yuan, "Cross-trees, edge and superpixel priors-based cost aggregation for stereo matching," *Pattern Recognition*, vol. 48, no. 7, pp. 2269–2278, 2015.

[10] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 650–656, 2006.

[11] X. Mei, X. Sun, M.-C. Zhou, S.-H. Jiao, H. Wang, and X. Zhang, "On building an accurate stereo matching system on graphics hardware," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV '11)*, pp. 467–474, Barcelona, Spain, November 2011.

[12] L. Zhou, G. Xu, K. Li, B. Wang, Y. Tian, and X. Chen, "Stereo matching algorithm based on census transform and modified

adaptive windows," *Acta Aeronautica et Astronautica Sinica*, vol. 33, no. 5, pp. 886–892, 2012.

[13] Y. Qu, J. Jiang, X. Deng, and Y. Zheng, "Robust local stereo matching under varying radiometric conditions," *IET Computer Vision*, vol. 8, no. 4, pp. 263–276, 2014.

[14] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 1073–1079, 2009.

[15] Q. Yang, "A non-local cost aggregation method for stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 1402–1409, Providence, RI, USA, June 2012.

[16] Q. Yang, "Stereo matching using tree filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 4, pp. 834–846, 2015.

[17] X. Mei, X. Sun, W. Dong, H. Wang, and X. Zhang, "Segment-tree based cost aggregation for stereo matching," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013*, pp. 313–320, June 2013.

[18] K. Zhang, Y. Fang, D. Min et al., "Cross-scale cost aggregation for stereo matching," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 407–414, June 2014.

[19] Y. Peng, Z. Hua, X. Yanbing et al., "Iterative color-depth MST cost aggregation for stereo matching," in *Proceedings of the 2016 IEEE International Conference on Multimedia and Expo, ICME 2016*, pp. 1–6, July 2016.

[20] C. C. Pham, V. Q. Dinh, and J. W. Jeon, "Robust non-local stereo matching for outdoor driving images using segment-simple-tree," *Signal Processing: Image Communication*, vol. 39, pp. 173–184, 2015.

[21] M. Gerrits and P. Bekaert, "Local stereo matching with segmentation-based outlier rejection," in *Proceedings of the 3rd Canadian Conference on Computer and Robot Vision, CRV 2006*, pp. 66–72, June 2006.

[22] J. Chen, C. Cai, and C. Li, "A multi-window stereo matching algorithm in rank tranform domain," in *Proceedings of the 2012 11th International Conference on Signal Processing, ICSP 2012*, pp. 997–1000, October 2012.

[23] V. Q. Dinh, D. D. Nguyen, V. Dinh Nguyen, and J. W. Jeon, "Local stereo matching using an variable window, census transform and an edge-preserving filter," in *Proceedings of the 2012 12th International Conference on Control, Automation and Systems, ICCAS 2012*, pp. 523–528, October 2012.

[24] G.-B. Kim and S.-C. Chung, "An accurate and robust stereo matching algorithm with variable windows for 3D measurements," *Mechatronics*, vol. 14, no. 6, pp. 715–735, 2004.

[25] Z. Gu, X. Su, Y. Liu, and Q. Zhang, "Local stereo matching with adaptive support-weight, rank transform and disparity calibration," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1230–1235, 2008.

[26] F. Tombari, S. Mattoccia, and L. Di Stefano, "Segmentation-based adaptive support for accurate stereo correspondence," in *Advances in Image and Video Technology*, vol. 4872 of *Lecture Notes in Computer Science*, pp. 427–438, Springer, Berlin, Germany, 2007.

[27] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[28] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, "Local stereo matching using geodesic support weights," in *Proceedings of the 16th IEEE International Conference on Image Processing, ICIP 2009*, pp. 2069–2072, November 2009.

[29] D. Chen, M. Ardabilian, X. Wang, and L. Chen, "An improved Non-Local Cost Aggregation method for stereo matching based on color and boundary cue," in *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo, ICME 2013*, pp. 1–6, July 2013.

[30] D. Scjarstein and R. Szeliski, Middlebury stereo evaluation, 2012 http://vision.middlebury.edu/stereo/eval/.

[31] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 3017–3024, June 2011.

Hindawi
Journal of Sensors
Volume 2017, Article ID 7051931, 8 pages
https://doi.org/10.1155/2017/7051931

Research Article

Automated Recognition of a Wall between Windows from a Single Image

Yaowen Zhang, Linsheng Huo, and Hongnan Li

State Key Laboratory of Coastal and Offshore Engineering, Dalian University of Technology, Dalian 116024, China

Correspondence should be addressed to Linsheng Huo; lshuo@dlut.edu.cn

Received 12 January 2017; Revised 8 April 2017; Accepted 12 April 2017; Published 3 May 2017

Academic Editor: Julio Rodriguez-Quiñonez

To avoid the time-consuming, costly, and expert-dependent traditional assessment of earthquake damaged structures, image-based automatic methods have been developed recently. Since automated recognition of structure elements is the basis by which these methods achieve automatic detection, this study proposes a method to recognize the wall between windows from a single image automatically. It begins from detection of line segments with further selection and linking to obtain longer line segments. The color features of the two sides of each long line segment are employed to pick out line segments as candidate window edges and then label them. Finally, the images are segmented into several subimages, window regions are located, and then the wall between the windows is located. Real images are tested to verify the method. The results indicate that walls between windows can be successfully recognized.

1. Introduction

An earthquake may result in thousands of structures suffering different levels of damage. The safety assessment of these structures is of great significance with regard to victim accommodation, intensity evaluation, and emergency aid provision. The traditional strategy is for certified engineers or structure experts to carry out the assessment. To analyze the defects carefully, the inspectors must have direct access to the structures and must move from one structure to another only on foot, due to destruction of the traffic system. Therefore, the traditional method is time-consuming, costly, and expert-dependent [1]. It is impossible to achieve fast and accurate assessment for all the damaged structures.

To develop a real-time and cost-effective method, researchers have applied computer vision technology for the assessment of damaged structures [2]. Recently, image-based methods have been developed to extract information related to defects such as cracks [3, 4] and spalling [5] and to recognize structural columns [6]. Although columns are the primary load-bearing elements, sometimes they may not be obvious to locate, particularly from the outside. In such conditions, the assessment of the walls could provide information relevant to the damage level of the structure. However, automated recognition of walls based on computer vision is still largely absent. Therefore, this study proposes a method to automatically recognize walls between windows.

Developing a recognition or detection method based on computer vision enables safety assessment to be achieved quickly. In the vista of the application of the methods, every person who can take pictures using cameras, mobile phones, or other devices can carry out the assessment task, with the images processed on local devices, on personal computers, or via cloud computing. With this intention, the method developed in this study is based on a single image, and there are no technical requirements for image acquisition.

Usually, there are few obvious and distinctive features on a wall itself, so this study describes how the windows next to the wall are used to locate it. Researchers have developed several methods for window detection from images [7–10]. These methods can detect windows in facades well, but they are based on the points that there are a couple of windows existing in one image and they have some same (or similar) features like shape, size, and so on, which means that the walls to be detected just occupy a small area in the image. In other words, the resolution of the wall is too low to provide enough information for the extraction detection. However, the automated recognition work is to be a basis for later

retrieval of defect information, so the wall in the image must be clear enough to ensure the accuracy of the defect information. In another aspect, when coming to an image only having two windows and a wall between them to assure the resolution of the wall, there are not sufficient same (or similar) features to be used. Therefore, the proposed method is for the image to have only one wall between windows.

The method first detects the line segments in the image by using a state-of-the-art line detection method. Then, the line segments are linked by a linking strategy, which is advanced by the authors. After selection of the long line segments, the color features are calculated to find and label the candidate window edges. The image is then segmented into subimages by the candidate line segments, and an assessment is conducted to identify a window region. Once the window regions are located, the wall between the windows is located.

2. Preconditions

The main purpose of the method described in this paper is to achieve automatic recognition. So, some preconditions are set to simplify the problem. (1) The image should be clear enough for people to observe it without much thinking. (2) The windows in the image should be easily recognizable and fully visible. (3) The window frame should be simple, common rectangles. (4) Only one target wall should exist, and its vertical edges should be vertical (or nearly vertical) in the image. It should be emphasized that condition (4) is not a natural constraint of the method proposed in this paper. As the recognition of the wall is just a prestep for the damage detection, condition (4) ensures that the wall region is clear enough during the following damage detection application. In addition, it could be easily extended to two or three target walls.

3. Framework of the Proposed Methodology

The main idea is to achieve automated recognition based on an obvious feature of the wall: that it is between windows. In other words, if there is a region for which both sides are windows, the region is recognized as a wall between windows. The method proposed in this paper mainly consists of three parts (Figure 1). The first is a line-detecting and linking part, aiming to detect the long line edges in the image. The second part is to design a classifier, which selects the candidate lines for window edges by using color features of the regions on both sides of the line. The third part involves segmenting the whole image by using all the candidate lines, after which the two window areas are located. The region between the two windows is then recognized as a wall between windows. Details of these three parts are explained in the following subsections.

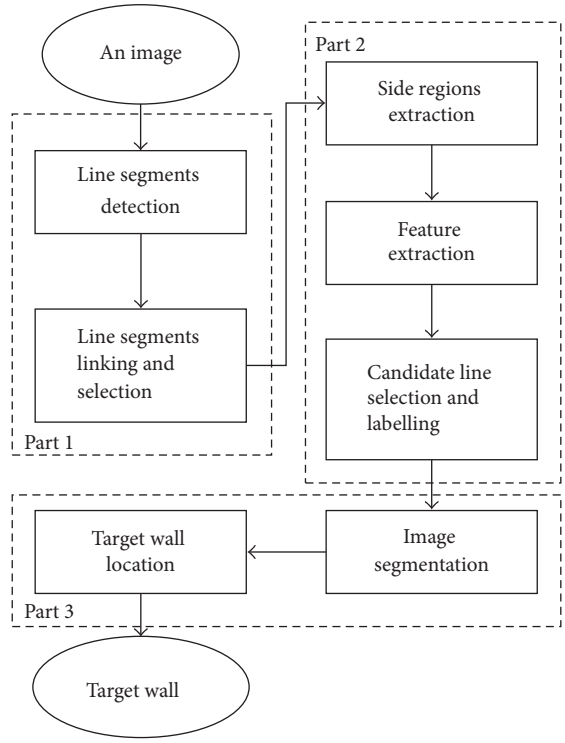


Figure 1: The framework of the proposed methodology.

4. Long Line Detection

4.1. Line Segments Detection. The first step to segment an image into regions is to find the edges of the regions. Considering the purpose of this paper, what we really need are lines to separate regions. Since line segment detection is an important and basic problem in many image processing-based applications, a number of methods have been proposed in the last few decades, from the typical traditional Hough transform [11] based to the newest CannyLines [12]. To detect line segments in images, there are usually two steps: edge detection and line segments detection. For the detection of edges, the Canny edge detection is a classical method and of better performance than others like Sobel, Roberts, and so forth. However, the newly Edge Drawing method can produce the similar quality edge map and runs up to 10% faster than the Canny edge detector [13]. For the detection of line segments, the result of the Hough transform was not very satisfied, due to too many fault detection with more computation time [14]. But the EDLines method, which makes use of the result of the Edge Drawing method, is 7 times faster with accurate results. Since this is not the main focus of the paper, just three methods state of the art, line segment detector (LSD) [15], EDLines [14], and CannyLines, are discussed here.

The LSD produces accurate line segments and controls the number of false detection instances at a low level by efficiently combining gradient orientations and line validation according to the Helmholtz principle [16].

The EDLines method [14] is a linear time line segment detector. It obtains edge segments (clean and continuous pixels) using the Edge Drawing algorithm [13]. According to the experiment of Akinlar and Topal [17], the parameter-free EDLines could run up to seven times faster than LSD with similar results.

The CannyLines method is developed to extract more meaningful line segments robustly from images. The Helmholtz principle, combining both gradient orientation and magnitude information, is used for verification. The line segments detected by CannyLines are longer than those of the other two methods. However, with its more complex steps, it is time-consuming.

Since this paper aims to achieve real-time detection, a method that is adaptive to different images without tuning of parameters and that has good accuracy and a high processing speed is the reasonable choice. Since the three methods are all parameter-free, and there is no large difference in their accuracy, this paper chooses the fastest method, EDLines, as the line segment detecting method.

4.2. Line Segments Linking. Due to image imperfection (e.g., weak contrast, clutter, blur, and noise), problems still persist when applying the line segment detector to practical civil infrastructure images [18]: (1) Edges of objects comprise small line segments with different orientations. (2) Missing line segments lead to edge discontinuity. To segment the image with windows, a long line, instead of many short line segments, is needed to indicate one edge of a window or wall. So before the segmentation, measures should be taken to obtain long lines from the line segments produced by EDLines.

After selecting vertical or horizontal line segments, an algorithm, a modified version of that proposed by [19], to link line segments is implemented. From computational geometry, two line segments that meet the following two criteria can be linked: (1) The two line segments are nearly collinear. (2) The two line segments are adjacent. For the collinearity problem, Kou et al. [19] transform it into an angle determination problem. For two line segments, there are four endpoints, named A, B, C, and D (Figure 2).

If every two endpoints are connected as a line segment, all these line segments can be matched to three pairs (AB and CD; AC and BD; AD and BC), with no consideration of their orientation and orders. While the three angles (α, β, and γ) constructed by the three pairs are all less than a specified tolerance, the original two line segments (AB and CD) are considered as "collinear." The first criterion could be written as follows:

$$\max(\alpha, \beta, \gamma) < \tau_1, \tag{1}$$

where τ_1 is the specified tolerance.

The second criterion can be written as follows:

$$\min(d_1, d_2, d_3, d_4) < \delta, \tag{2}$$

where d_1, d_2, d_3, and d_4 are the Euclidean distances of the end points AC, AD, BC, and BD, respectively, and δ is the specified tolerance.

This means that if the smallest of the distances between every two end points of the two line segments is smaller than the tolerance, the two line segments (AB and CD) are considered to be "adjacent." According to Kou et al. [19], if two line segments meet both (1) and (2), the two line segments that are mostly parallel, are close, and have no overlap (Figure 3), which commonly occurs in edge detection and line detection, and then in most cases they should be linked as one line.

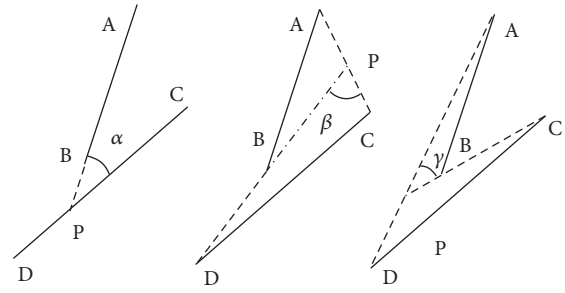
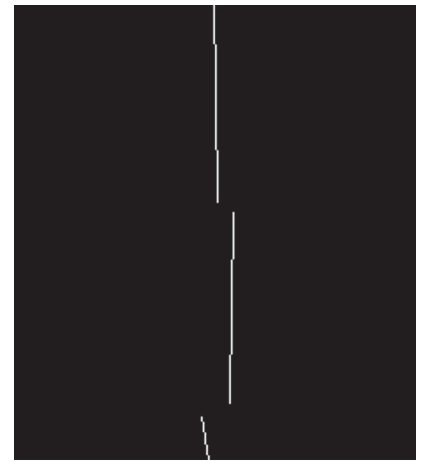
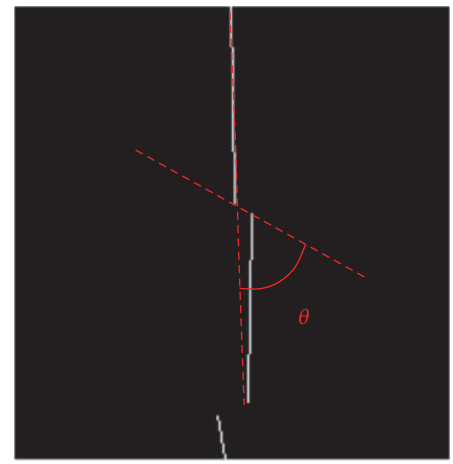


Figure 2: Illustration of constructed angles by the three pairs of endpoints.



(a)



(b)

Figure 3: Illustration of two line segments being mostly parallel, close, and with no overlap but not meeting criteria (1).

Input: line segments set L
Output: the new line segments set T
(1) $T \leftarrow \emptyset$
(2) **for each** line segment $l \in L$ **do**
(3) $S \leftarrow \emptyset$
(4) **for each** line segment $s \in L$ $(s \neq l)$ **do**
(5) **if** s is collinear with l **then**
(6) $S \leftarrow S \cup s$
(7) $T \leftarrow$ newline (l, S)
(8) **function** newline (l, S)
(9) **if** $S \neq \emptyset$ **then**
(10) find the most close line segment $s \in S$
(11) **if** s is adjacent to l **then**
(12) $l \leftarrow$ link (l, s)
(13) delete s from S
(14) newline (l, S)
(15) **return** l
(16) **function** link (l, s)
(17) find the top most vertex u and bottom most vertex v of l and s
(18) draw a line segment between u and v

Algorithm 1: The algorithm of line segments linking.

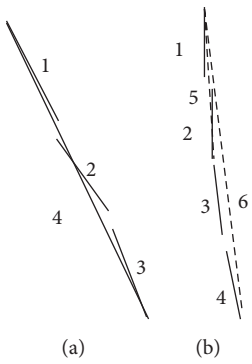


Figure 4: Diagram of line linking. (a) Three short line segments 1, 2, and 3 are linked as a long line segment 4. (b) 1 and 2 are linked as 5 (short dashed), if collinearity is tested between 5 and 3, and then 4, the linking result, would be 6 (long dashed).

But, obviously, this situation does not meet (1). Therefore, based on the two formulas, the authors add a condition to (1):

$$\max(\alpha, \beta, \gamma) < \tau_1 \quad \text{or } \alpha < \tau_2, \tag{3}$$

where α is the angle of the original two line segment. τ_2 is a specified tolerance, which is smaller than τ_1. Therefore, the authors propose that if two line segments meet both (2) and (3), they are linkable.

When two line segments are linkable, the top-most and bottom-most endpoints of the segments are used to construct a new line segment (Figure 4(a)). At the same time, the original line segments are eliminated.

At this time, it is important that any new line segment should not be tested with other line segments to see if they are collinear. Or that would lead to a serious fault (Figure 4(b)). Fortunately, this is naturally avoided in the algorithm (Algorithm 1).

After linking, the long line segments, whose length is larger than a threshold, are picked out to be classified.

5. Classifier Designing

5.1. Subimage Extraction. To determine which long line segment should be the candidate for the window edge, it is necessary to inspect the regions on both sides of the line segment. Therefore, the subimages of both regions are extracted from the image, whose shape and size are chosen experimentally. Since only rectangular windows are considered in this paper, a rectangle is used as the shape of the side region, with its longer side parallel to the line segment and having the same length. The side ratio of the rectangle is set as 0.5.

5.2. Features Extraction. According to human intuition, when a window is seen from outside, the edge of the window usually has two obvious features. (1) The color of the regions on the two sides of the edge has sharp contrast. (2) The gray scale of the window side is lower than that of the wall side. This paper uses feature (1) to select the candidate line segments as the edges of window and then feature (2) to label the window side.

The digital image is specified by a RGB color model, which is a mixture of three primary colors: red, green, and blue. After extracting the side subimages of the line segment, the histogram of each primary image is calculated by dividing the range (0–255) into several bins N_{bins}. Using $\mathbf{H}_{r1}$, $\mathbf{H}_{g1}$, $\mathbf{H}_{b1}$, and $\mathbf{H}_{r2}$, $\mathbf{H}_{g2}$, $\mathbf{H}_{b2}$ represent the histogram vectors of red, green, and blue primary images of each subimage. The subscript 1 represents the left or upper side of the line segment

Table 1: The window position related to the point.

Point label	Window position
1	Upper, left
2	Upper, right
3	Below, left
4	Below, right

and subscript 2 the right or lower side. The same meanings of subscripts are used in the following contexts. Then

$$\mathbf{D} = \left\|\mathbf{H}_{r1} - \mathbf{H}_{r2}\right\|_2 + \left\|\mathbf{H}_{g1} - \mathbf{H}_{g2}\right\|_2 + \left\|\mathbf{H}_{b1} - \mathbf{H}_{b2}\right\|_2, \quad (4)$$

where $\mathbf{D}$ represents the color difference between the two sides. The larger the value of $\mathbf{D}$ is, the more obvious one side contrasts with the other.

Taking g_{m1} and g_{m2} as the average gray scale of both subimages, if $g_{m1} > g_{m2}$, then the line segment is labeled with 1, or else labeled with 0. For the edge of the image, the image side is assumed as the window side. So the left and top edges are labeled with 0, while the bottom and right edges are labeled with 1. The reason for including the edge of image is that sometimes windows may not be completely included in the image.

When all the long line segments in one image are processed as above, the average of all their contrast differences $\mathbf{D}$ is calculated as follows:

$$\mathbf{D}_m = \frac{1}{n}\sum_{i=1}^{n}\mathbf{D}_i, \quad (5)$$

where n is the number of long line segments.

$\mathbf{D}_m$ is chosen as the threshold to classify the long line segments. The line segment whose contrast difference $\mathbf{D}$ is larger than $\mathbf{D}_m$ is selected as the candidate edge line segment.

6. Wall between Windows Recognition

6.1. Image Segmentation. After establishing the candidate edge line segments, the image would be segmented into several areas by the line segments.

As the line segments are always shorter or longer than the window edges, all the candidate line segments are extended to the edge of the image. Then, the intersection point of each pair of extended line segments and each line segment and image edge are calculated. Since all the line segments have been labeled above, each intersection point is also labeled (Table 1) to indicate the window position relative to the point (Figure 5).

6.2. Wall Recognition. As mentioned above, the strategy to recognize a wall between windows is to find a region of which both sides are window regions. So firstly, the window regions should be located. To achieve that, all the subimages segmented during the image segmentation part are inspected with two constraints. (1) The area of the subimage should be larger than a threshold T_a. (2) The clockwise order of the vertexes of the subimage should be 4, 3, 1, 2 starting from the upper left vertex (Figure 5).

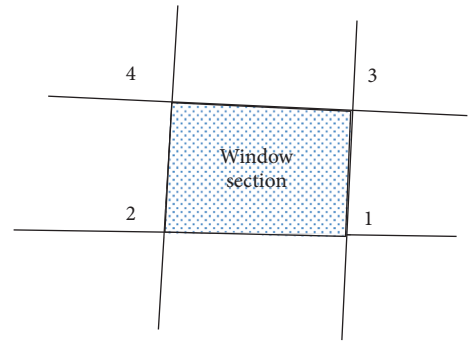


Figure 5: Diagram of the point label.

Table 2: Threshold setting.

Threshold	Value
τ_1	$Ls/120$
δ	$7\pi/180$
τ_2	$\pi/180$
N_{bins}	11
T_a	0.05

Note: Ls is the length of the short edge of the image.

During the inspection, the point where the subimage meets the two constraints would be recognized as a window region, and the four vertexes are recorded. Usually, there should be only two window regions. Vertexes 3 and 1 from the left region and vertexes 4 and 2 from the right are then used as the vertexes of the wall between windows regions.

7. Implementation and Results

In order to evaluate the overall performance of the proposed method, the algorithm is developed in Matlab 2014a. During the line segment detection step, the code provided by Topal and Akinlar [13] is used. All the programs are implemented on a laptop (Lenovo E531 with Intel i5-3230M and 8-GB memory).

The thresholds of the method are set experimentally as shown in Table 2. An example is shown in Figure 6, where (a) is the original image, (b) shows the horizontal and vertical candidate line segments detected, (c) shows the regions segmented by the candidate line segments after extension, (d) illustrates the window sections located, and (e) is the detection result, from which the wall between the windows is extracted.

Figure 7 shows another example of the detection result.

Precision and recall ratios are used to measure the detection performance of the method. They are calculated as follows:

$$\text{precision} = \frac{\text{TP}}{(\text{FP} + \text{TP})}$$
$$\text{recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})}, \quad (6)$$

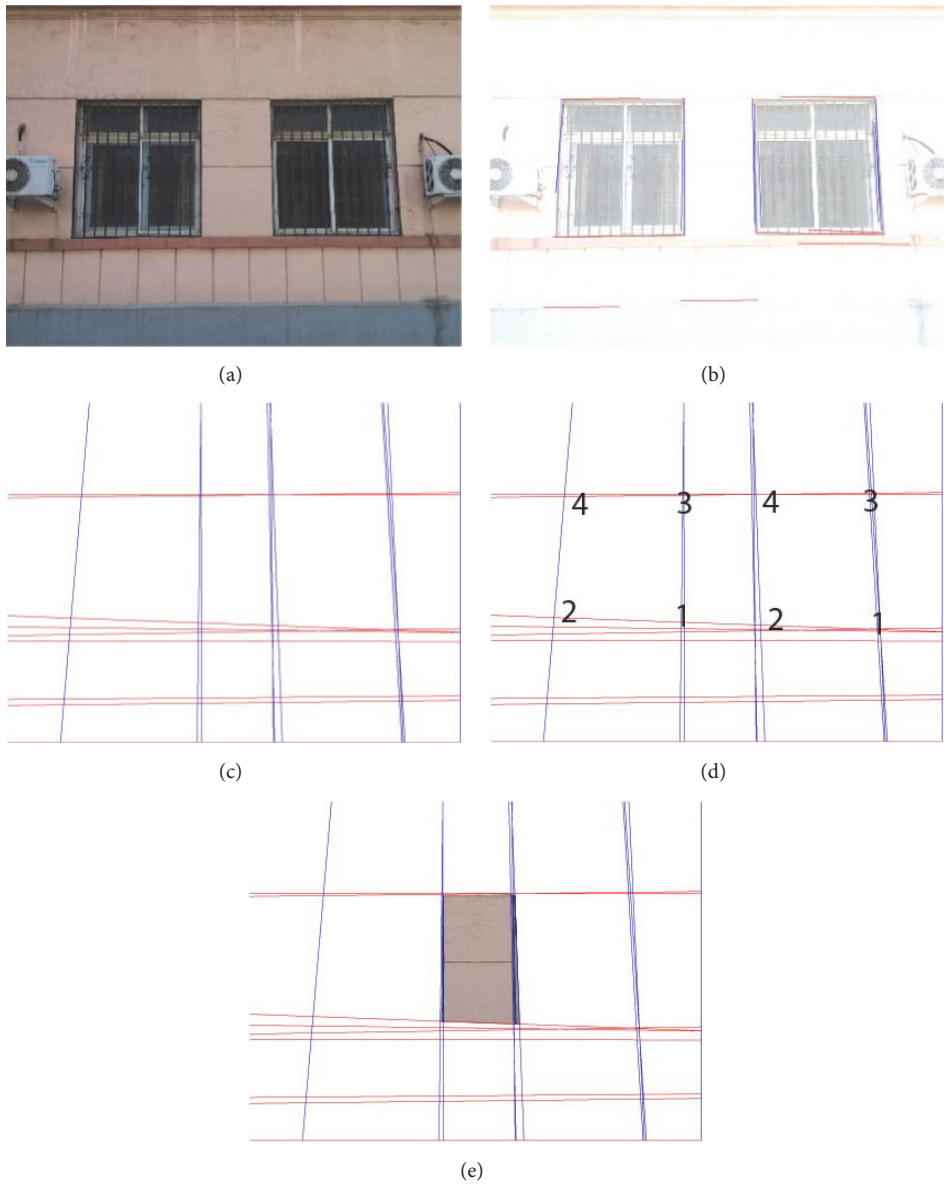


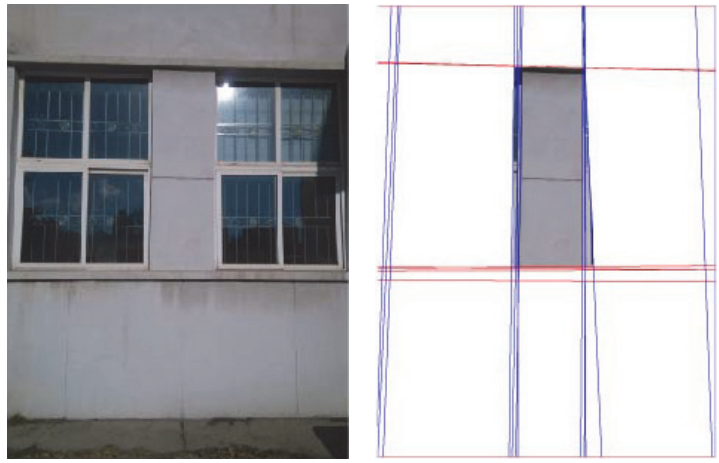
Figure 6: An example of the detection result. (a) The original image. (b) Candidate line segments. (c) Regions segmented. (d) Window section located. (e) Wall between windows extraction.

where TP is the number of walls between windows that are correctly detected as walls between windows. FP is the number of other areas incorrectly detected as walls between windows. FN is the number of walls between windows incorrectly detected as other areas. The incomplete result is also treated as incorrect, because it cannot be used in future damage detection.

High precision means that many detected walls between windows are actual walls between windows, whereas low precision means that few detected walls are actual walls. Similarly, high recall means many actual walls between windows are correctly detected, whereas low recall means that few actual walls are correctly detected. Both sets of results represent the quality of the detection result of the method. A set of images captured by a mobile phone (Xiaomi 2) around the campus is used to test the method. Before testing, the images that do not meet the specified preconditions of the paper are removed manually. So, 20 images are actually tested. The resolution of the images is $2448 * 3264$. The precision and recall ratio of the test results are shown in Table 3.

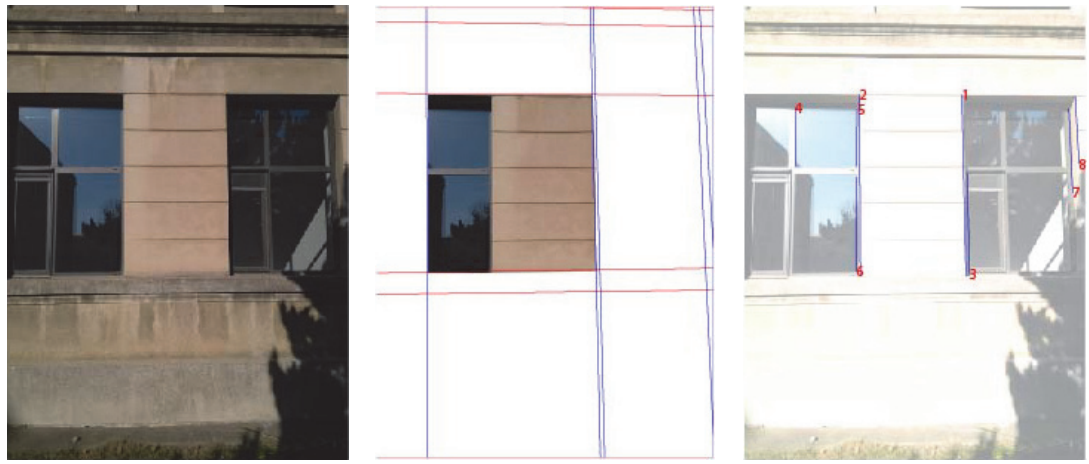
8. Discussion

According to the results, the quality of the method is acceptable. Although the precision is not very high, the recall reaches 85.71%. This means that only a few of the actual walls would be detected as other objects. However, the FP is a little high because some lines that are not window edges also have sharp contrast between their two sides. Take one typical image (Figure 8) as an example. Due to the reflection



(a) (b)

Figure 7: An example of the detection result. (a) The original image. (b) Wall between windows extraction.



(a) (b) (c)

Figure 8: Example for incorrect detection: (a) original image. (b) The incorrect detection result. (c) The candidate edge line segments of the image.

Table 3: Testing results.

Image number	TP	FP	FN	Precision	Recall
20	12	6	2	66.67%	85.71%

of neighboring structures, the contrast of line segment 4 is sufficiently large to lead to an incorrect selection as a window edge. Since its label is the same as that of line segments 2, 5, and 6, the latter three actual window edge line segments have no opportunity to play a part in the correct detection by the proposed method.

The average time cost of the images is 8.3 s. Considering the purpose of the paper, this is acceptable. The time is heavily dependent on the complexity of the image. The length of time required increases with the number of line features and complex textures in the image. For the image in Figure 6 the time is 8.225 s, and for the image in Figure 7 the time is 8.716 s, as there are many line features within the guardrails.

9. Conclusion and Future Work

The traditional method to assess the damage of a structure element after an earthquake is time-consuming, costly, and expert-dependent. Incorporating image technology into the assessment can make the task simpler and faster and is especially feasible when there are many people who can take pictures of the damaged sites. Thanks to the preresearchers who have advanced image processing methods, we can now get more information from images. With the intention of simplifying the assessment, this paper tries to develop a method based on a single image.

As there are few studies on structure element recognition, especially recognition of walls, the paper develops a method to automatically recognize walls between windows from a

single image. The method first detects the line segment in the image and then picks out the horizontal (near horizontal) and vertical (near vertical) line segments and links them to get longer line segments using several principles. The color features of the two sides of each line segment are calculated and used for selecting and labeling candidate window edges. While the image is segmented by the candidate line segments, window regions are located, and then the wall between the windows is located.

Real images taken by mobile phone cameras are used to test the validity of the method. The results show that the method can detect the wall between windows when there is only one target in the single image.

Although the method is for images that have only one wall between windows, it can be easily adapted to apply to two or more targets in a single image. Moreover, with only simple adaption of the region location strategy, other types of walls can be detected.

Since the precision of the method is not very high and it is mostly based on the feature classifier strategy, future work will be conducted regarding this issue. As the work of this paper aims to achieve automatic assessment of structural elements, automatic retrieval of defect information on the wall will also be attempted in future work.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

Innovative research group project (Grant no. 51421064) and general project (Grant no. 51578114) of Natural Science Foundation of China and the Fundamental Research Funds for the Central Universities (DUT16TD03) support this work. The authors would like to thank their sponsor.

References

[1] B. Sun and G. Zhang, "Statistical analysis of the seismic vulnerability of various types of building structures," *China Civil Engineering Journal*, vol. 45, no. 6, pp. 26–30, 2012 (Chinese).

[2] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[3] T. Yamaguchi and S. Hashimoto, "Fast crack detection method for large-size concrete surface images using percolation-based image processing," *Machine Vision and Applications*, vol. 21, no. 5, pp. 797–809, 2010.

[4] Z. Zhu, *Column Recognition and Defects (or) Damage Properties Retrieval for Rapid Infrastructure Assessment and Rehabilitation Using Machine Vision*, Department of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta, USA, 2011.

[5] S. German, I. Brilakis, and R. Desroches, "Rapid entropy-based detection and properties measurement of concrete spalling with machine vision for post-earthquake safety assessments," *Advanced Engineering Informatics*, vol. 26, no. 4, pp. 846–858, 2012.

[6] Z. Zhu and I. Brilakis, "Concrete column recognition in images and videos," *Journal of Computing in Civil Engineering*, vol. 24, no. 6, pp. 478–487, 2010.

[7] S. C. Lee and R. Nevatia, "Extraction and integration of window in a 3D building model from ground view images," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. II113–II120, USA, July 2004.

[8] M. Recky and F. Leberl, "Windows detection using K-means in CIE-Lab color space," in *Proceedings of 20th International Conference on Pattern Recognition (ICPR)*, pp. 356–359, Turkey, August 2010.

[9] M. Miljanovic, T. Eiter, and U. Egly, "Detection of windows in facades using image processing algorithms," *Indian Journal of Computer Science Engineering*, vol. 3, no. 4, pp. 539–547, 2012.

[10] J. Miao, J. Chu, and G. Zhang, "Window detection based on constraints of image edges and glass attributes," *Journal of graphics*, vol. 36, no. 5, pp. 776–782, 2015 (Chinese).

[11] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[12] X. Lu, J. Yao, K. Li, and L. Li, "CannyLines: A parameter-free line segment detector," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pp. 507–511, Canada, September 2015.

[13] C. Topal and C. Akinlar, "Edge drawing: a combined real-time edge and segment detector," *Journal of Visual Communication and Image Representation*, vol. 23, no. 6, pp. 862–872, 2012.

[14] C. Akinlar and C. Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[15] G. V. G. Rafael, J. Jérémie, M. Jean-Michel, and R. Gregory, "LSD: a fast line segment detector with a false detection control," *Pattern Analysis Machine Intelligence IEEE Transactions online*, vol. 32, no. 4, pp. 722–732, 2010.

[16] A. Desolneux, L. Moisan, and J. M. Morel, *From Gestalt Theory to Image Analysis: A Probabilistic Approach*, Springer, 2008.

[17] C. Akinlar and C. Topal, "Edpf: a real-time parameter-free edge segment detector with a false detection control," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 26, no. 1, Article ID 1255002, 22 pages, 2012.

[18] F. Dai and Z. Zhu, "Line Segment Grouping and Linking: A Key Step Toward Automated Photogrammetry for Non-Contact Site Surveying," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 79, no. 3-4, pp. 371–384, 2015.

[19] Z. Kou, Z. Shi, and L. Liu, "Airport detection based on Line Segment Detector," in *Proceedings of International Conference on Computer Vision in Remote Sensing (CVRS)*, pp. 72–77, China, December 2012.

Hindawi
Journal of Sensors
Volume 2017, Article ID 6471250, 6 pages
https://doi.org/10.1155/2017/6471250

Research Article

Visual Vehicle Tracking Based on Deep Representation and Semisupervised Learning

Yingfeng Cai,[1] Hai Wang,[2] Xiao-qiang Sun,[1] and Long Chen[1]

[1]*Automotive Engineering Research Institution, Jiangsu University, Zhenjiang, Jiangsu 212013, China*
[2]*School of Automotive and Traffic Engineering, Jiangsu University, Zhenjiang, Jiangsu 212013, China*

Correspondence should be addressed to Hai Wang; wanghai1019@163.com

Received 21 December 2016; Revised 4 February 2017; Accepted 15 February 2017; Published 9 March 2017

Academic Editor: Oleg Sergiyenko

Discriminative tracking methods use binary classification to discriminate between the foreground and background and have achieved some useful results. However, the use of labeled training samples is insufficient for them to achieve accurate tracking. Hence, discriminative classifiers must use their own classification results to update themselves, which may lead to feedback-induced tracking drift. To overcome these problems, we propose a semisupervised tracking algorithm that uses deep representation and transfer learning. Firstly, a 2D multilayer deep belief network is trained with a large amount of unlabeled samples. The nonlinear mapping point at the top of this network is subtracted as the feature dictionary. Then, this feature dictionary is utilized to transfer train and update a deep tracker. The positive samples for training are the tracked vehicles, and the negative samples are the background images. Finally, a particle filter is used to estimate vehicle position. We demonstrate experimentally that our proposed vehicle tracking algorithm can effectively restrain drift while also maintaining the adaption of vehicle appearance. Compared with similar algorithms, our method achieves a better tracking success rate and fewer average central-pixel errors.

1. Introduction

Visual vehicle tracking is one of the key technologies used in active vehicle safety applications. It is used in advanced driver assistance systems (ADAS) and in intelligent vehicles. However, in real traffic situation, the camera and the vehicles need to be tracked are all in motion. The visual tracking of target vehicles is often affected by complex backgrounds, changes in illumination, and occlusion by other vehicles or objects. These factors make vehicle tracking a challenging task in real traffic scenarios.

Existing tracking algorithms generally fall into one of two categories: generative models and discriminative models. Discriminative models convert tracking problems into binary classification problems of target and background. They have attracted much research interest in recent years. Many representative methods have been proposed, such as the online AdaBoost algorithm [1], the multiple instance learning algorithm [2], the TLD (tracking-learning-detection) algorithm [3], and the SVT (support vector tracking) algorithm [4].

The biggest challenge for discriminative model based tracking methods is the tracker drifting problem. This is because only a small amount of labeled samples (e.g., there is only one positive sample in most cases) can be used to train the classifier. Additionally, the tracking of subsequent image sequences depends on the classification results of the former frame. So, if the tracked area of former frame is not locked onto the optimal target, this self-training approach can lead to classifier drifting, which causes accumulated errors and further tracking failures.

To solve this shortcoming of the self-training approach, a new method, named semisupervised learning-based tracking, has been proposed [5–7]. In this method, a large amount of auxiliary images is used to maintain a feature dictionary which is able to describe images. Then, the dictionary will be used to update the tracker online. However, this kind of method prefers to use pixels from the original image or handcrafted features (such as the Haar feature, HOG feature, or DIFT feature), to generate the feature dictionary. This

cannot satisfy the requirements of the image classification that is needed for robust tracking.

The newly developed deep learning framework is a bioinspired architecture which describes data such as images, voices, and text by mimicking the human brain's mechanisms of learning and analysis. Through deep learning, features are transformed from their original space in a lower layer to a new space in a higher layer [8]. Compared with handcrafted features, the automatic features generated by deep learning are more capable of expressing the details of internal properties of the data. Inspired by this, a semisupervised vehicle tracking algorithm is proposed that uses deep representation. The overall structure of the proposed method is shown in Figure 1. Firstly, a large number of unlabeled images are selected as training samples, and a 2D deep belief network (2D-DBN) is used as a classifier structure. Then, a multilayer deep network will be trained with those unlabeled samples and the nonlinear mapping node in the top layer will be taken as the feature dictionary. In the tracking process, the subimage containing the vehicle to be tracked in the initial frame is set as the positive sample, and the surrounding background subimages are set as negative samples. After that, an online deep tracker will be learned from the generated samples and the feature dictionary. Finally, a partial filter will be used to estimate and provide a small area for the tracker. The deep learning-based tracking algorithm is able to fully exploit the deep-level feature information embedded in the image. This effectively suppresses drift situations while keeping the vehicle tracking system updated.

Generally, compared with other semisupervised learning-based tracking methods which use handcrafted features [5–7], this work introduces deep learning framework to the traditional semisupervised learning. This work applies 2D-DBN deep model to generate features automatically in unsupervised offline training and then uses small amount number of samples to adjust the tracker online.

In this article, in Section 2, the establishment of the deep model based semisupervised tracking framework is introduced in detail which contains both offline and online training steps. The vehicle position prediction method based on partial filter is given in Section 3. In Section 4, the experiment results and its analysis will be shown. Finally, a brief conclusion of this work is given in Section 5.

2. Establishment of Vehicle Tracking Based on Deep Modeling

The establishment of deep tracking requires two steps (Figure 2). (1) Offline training session: a large number of unlabeled images are selected as training samples and the 2D-DBN is trained unsupervised. (2) Online training session: the labeled samples are refined online using the parameters of the 2D-DBN with a backpropagation algorithm.

2.1. Unsupervised Offline Training. In the unsupervised offline training process, many road images containing vehicles are selected and a large amount of unlabeled samples with $M * M$ pixels is generated from them. Then, all sample images

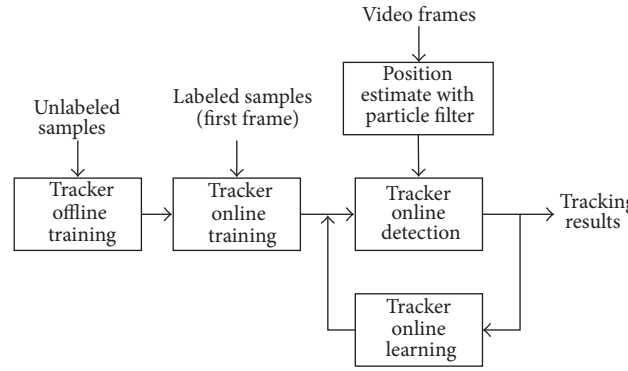


Figure 1: Overall structure of proposed vehicle tracking algorithm.

are converted to gray scale and normalized to the $[0, 1]$ interval. After that, the entire normalized matrix is input to the 2D-DBN to perform feature dictionary extraction.

The structure of the proposed 2D-DBN is shown in Figure 3. It is constructed with one visible layer, V^1, and two hidden layers, H^1 and H^2. The visible layer contains $M \times M$ units which is equal to the dimension of input samples. The hidden layers H^1 and H^2 contain $P \times P$ and $Q \times Q$ units, respectively. The visible layer and hidden layer are connected with a group of weights, θ. After unsupervised training to adjust the weights, the unit in hidden layer H^2 can be considered as the feature dictionary.

In unsupervised training, the greedy-wise reconstruction algorithm is used to adjust the weights between each two layers [9]. Taking visible layer V^1 and hidden layer H^1 as an example, V^1 and H^1 can be seen as a restricted Boltzmann machine (RBM). The state energy (v^1, h^1) of any two units of V^1 and H^1 can be written as

$$\begin{aligned} E\left(\mathbf{v}^1, \mathbf{h}^1, \theta^1\right) &= -\left(\mathbf{v}^1 \mathbf{A} \mathbf{h}^1 + \mathbf{b}^1 \mathbf{v}^1 + \mathbf{c}^1 \mathbf{h}^1\right) \\ &= -\sum_{i=1,j=1}^{i\le M, j\le M} \sum_{p=1,q=1}^{p\le P, q\le P} v_{ij}^1 A_{ij,pq}^1 h_{pq}^1 \\ &\quad - \sum_{i=1,j=1}^{i\le M, j\le M} b_{ij}^1 v_{ij}^1 - \sum_{p=1,q=1}^{p\le P, q\le P} c_{pq}^1 h_{pq}^1, \end{aligned} \tag{1}$$

where $\theta^1 = (\mathbf{A}^1, \mathbf{b}^1, \mathbf{c}^1)$ are the weight parameters of the units between visible layer V^1 and hidden layer H^1. $A_{ij,pq}^1$ is the connecting weight of unit (i, j) in visible layer V^1 and unit (p, q) in hidden layer H^1. b_{ij}^1 and c_{pq}^1 represent the bias between corresponding layers.

So, the RBM can be considered as a joint probability distribution:

$$P\left(\mathbf{v}^1, \mathbf{h}^1; \theta^1\right) = \frac{1}{Z} e^{-E(\mathbf{v}^1, \mathbf{h}^1; \theta^1)} = \frac{e^{-E(\mathbf{v}^1, \mathbf{h}^1; \theta^1)}}{\sum_{v^1} \sum_{h^1} e^{-E(\mathbf{v}^1, \mathbf{h}^1; \theta^1)}}, \tag{2}$$

where Z is a normalized parameter.

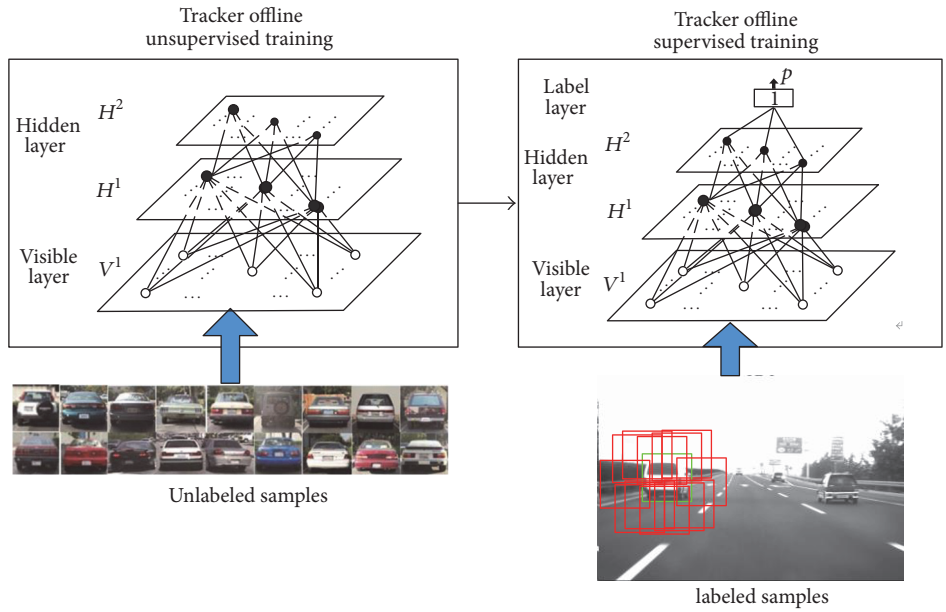


Figure 2: Overall structure of proposed semisupervised learning and deep model based method.

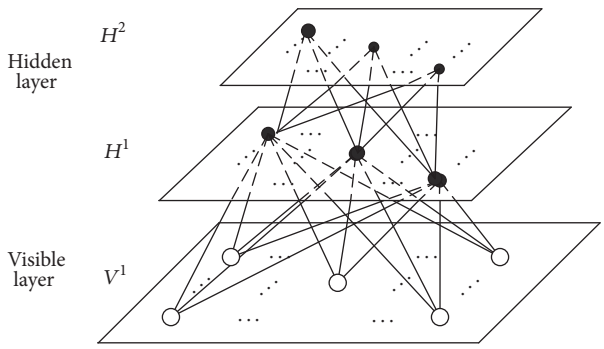


Figure 3: Structure of the deep belief network.

Then, the conditional probability distributions of input state $\mathbf{v}^1$ and hidden state $\mathbf{h}^1$ are able to be expressed with logistic functions:

$$p\left(\mathbf{h}^1 \mid \mathbf{v}^1\right) = \prod_{p,q} p\left(h_{pq}^1 \mid \mathbf{v}^1\right),$$
$$p\left(h_{pq}^1 \mid \mathbf{v}^1\right) = \sigma\left(\sum_{i=1,j=1}^{i\le M, j\le M} v_{ij}^1 A_{ij,pq}^1 + c_{pq}^1\right)$$
$$p\left(\mathbf{v}^1 \mid \mathbf{h}^1\right) = \prod_{i,j} p\left(\mathbf{v}_{ij}^1 \mid \mathbf{h}^1\right), \quad (3)$$
$$p\left(\mathbf{v}_{ij}^1 \mid \mathbf{h}^1\right) = \sigma\left(\sum_{p=1,q=1}^{p\le P, q\le P} h_{pq}^1 A_{ij,pq}^1 + b_{ij}^1\right),$$

where $\sigma(x) = 1/(1 + \exp(-x))$.

Based on the analysis above, the connecting weight and bias will be updated with a contrast divergence algorithm [9]:

$$A_{ij,pq}^1 = \vartheta A_{ij,pq}^1 + \varepsilon_A\left(\left\langle v_{ij}^1(0)\, h_{ij}^1(0)\right\rangle_{\text{data}} - \left\langle v_{ij}^1(t)\, h_{ij}^1(t)\right\rangle_{\text{recon}}\right) \quad (4)$$
$$b_{ij}^1 = \vartheta b_{ij}^1 + \varepsilon_b\left(v_{ij}^1(0) - v_{ij}^1(t)\right)$$
$$c_{pq}^1 = \vartheta c_{pq}^1 + \varepsilon_c\left(h_{pq}^1(0) - h_{pq}^1(t)\right),$$

where $\langle\cdot\rangle_{\text{data}}$ is the data's expected distribution, $\langle\cdot\rangle_{\text{recon}}$ is the reconstruction distribution, and $t = 1$ is the update step size.

By repeating these steps from lower layers to higher layers, the weights of (V^1, H^1) and (H^1, H^2) can be maintained. The weights of every unit in H^2 can then be considered as the nonlinear feature dictionary.

In real practice, many subimages captured from road videos that are vehicles will be input to the visible layer one by one. Then with the unsupervised training process given before, all the weights of the 2D-DBN will be adjusted and if we graphically display the weights in the top layer, the features are able to be viewed. Some of the features generated by the proposed method of unsupervised learning are shown in Figure 4. It can be seen from the feature that the features are more sensitive to the shape of edge, corner, and so on which exist more usually in vehicles.

2.2. Tracker Online Training. When the tracked area is identified in the first frame of a video, positive and negative samples are generated. Firstly, the tracked area will be chosen as

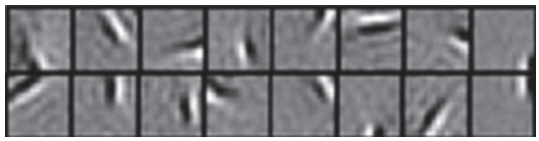


Figure 4: Example of features generated by unsupervised training.

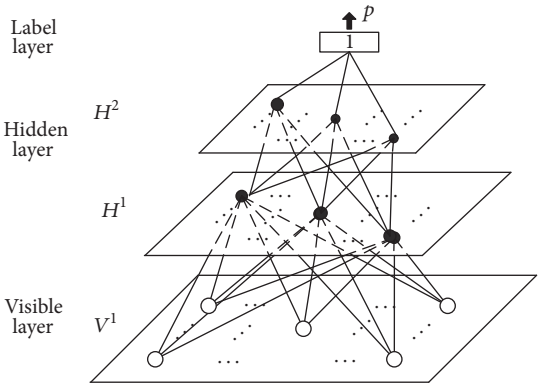


Figure 5: The structure of the proposed artificial neural network based on a DBN.

the positive sample and will then be rotated from −5 to 5 degrees with an intersection of 1 degree to generate more positive samples. Secondly, negative subimages are generated in a neighborhood ring area around the tracked area. The neighborhood area is defined as $\alpha < \|I_{\text{neg}} - I\| < \beta$, in which I_{neg} is the center of the negative samples, I is the center of the tracked area, and α and β are the inner and outer diameters of the ring area.

Based on the offline trained DBN, a label layer is introduced with a sigmoid function to generate a two-dimensional forward artificial neural network (Figure 5). In the online training process, the positive and negative samples and their labels are input to this neural network. The backpropagation algorithm is used to perform supervised training and adjust all the network weights. Finally, all the subimages in consequent frames are loaded to the newly updated network to make judgments.

3. Position Prediction Based on Particle Filtering

In the tracking process, the online trained classifier needs to verify a large number of subimages in consequent frames. To reduce processing time, a particle filter is used to estimate the target position.

The particle filter uses a Monte Carlo algorithm with Bayesian filtering and demonstrates good object tracking performance. Set $\mathbf{x}_t$ and $\mathbf{z}_t$ as the state and observation values of the target at time t. Then, the vehicle position estimate can be described as follows: in the condition of knowing observation value $\mathbf{Z}_t = \{\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$ at time t, iteratively estimate the state of the system at time t.

Set the state space of system as

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}) + \mathbf{v}_{t-1} \\ \mathbf{z}_t &= h(\mathbf{x}_t) + \mathbf{n}_t, \end{aligned} \tag{5}$$

where f and h are the state transfer function and observation function, respectively, and $\mathbf{v}$ and $\mathbf{n}$ represent system noise and observation noise, respectively.

The filtering process contains two main steps: forecasting and updating. In forecasting process, the posterior probability density $P(\mathbf{x}_{t-1} \mid \mathbf{z}_{t-1})$ at time $t-1$ is used to obtain the prior probability density at time t:

$$P(\mathbf{x}_t \mid \mathbf{z}_{t-1}) = \int P(\mathbf{x}_t \mid \mathbf{x}_{t-1}) P(\mathbf{x}_{t-1} \mid \mathbf{z}_{t-1}) \, d\mathbf{x}_{t-1}. \tag{6}$$

In the updating process, the newest system state observation value $\mathbf{z}$ at time t and the prior probability density $P(\mathbf{x}_{t-1} \mid \mathbf{z}_{t-1})$ will be used to calculate the posterior probability density $P(\mathbf{x}_t \mid \mathbf{z}_t)$ at time t:

$$P(\mathbf{x}_t \mid \mathbf{z}_t) = \frac{\int P(\mathbf{z}_t \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{z}_{t-1})}{P(\mathbf{z}_t \mid \mathbf{z}_{t-1})}. \tag{7}$$

Assume that $\{\mathbf{x}_{0:t}^i, \omega_t^i\}_{i=1}^N$ is a group of samples with weights from the posterior probability density and $\sum \omega_t^i = 1$, $\mathbf{x}_{0:t} = \{\mathbf{x}_j,\ j = 0, 1, 2, \ldots, t\}$ is the sample set from time 0 to time t. Based on the Monte Carlo principle, the posterior probability density at time t is able to be approximated with a discrete weighting formula:

$$P(\mathbf{x}_{0:t} \mid \mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} \omega_t^i \delta(\mathbf{x}_{0:t} - \mathbf{x}_{0:t}^i) \tag{8}$$

in which $\delta(\cdot)$ is a Dirac function.

Further, the estimated value of system state $\mathbf{x}_t$ at time t is

$$\widehat{\mathbf{x}}_t = \sum_{i=1}^{N} \omega_t^i \mathbf{x}_t^i. \tag{9}$$

4. Experiment and Analysis

The key parameters of the deep network and the particle filter are set out as follows: the number of units of the visible layer is 24×24 ($M = 24$); the number of units of the two hidden layers is 18×18 and 12×12 ($P = 18$, $Q = 12$); the initial training weights of offline training are $[0, 1]$, which satisfies the Gaussian distribution; the number of particles in the particle filter is $N = 1000$. For the unsupervised offline training, around 5000 subimages that only contain vehicle are selected for tracker training. Some of the typical images for training are shown in Figure 6.

In the experiments, different road scenarios were selected, including daytime, nighttime, and raining conditions. To evaluate performance, two criteria were used: the tracking success rate and center pixel offset. In these two criteria, tracking success is defined as $\text{area}(BB_T \cap BB_G)/\text{area}(BB_T \cup BB_G) \geq 50\%$ and center pixel offset is defined as the Euclidean

Figure 6: Some of the typical images for unsupervised offline training.

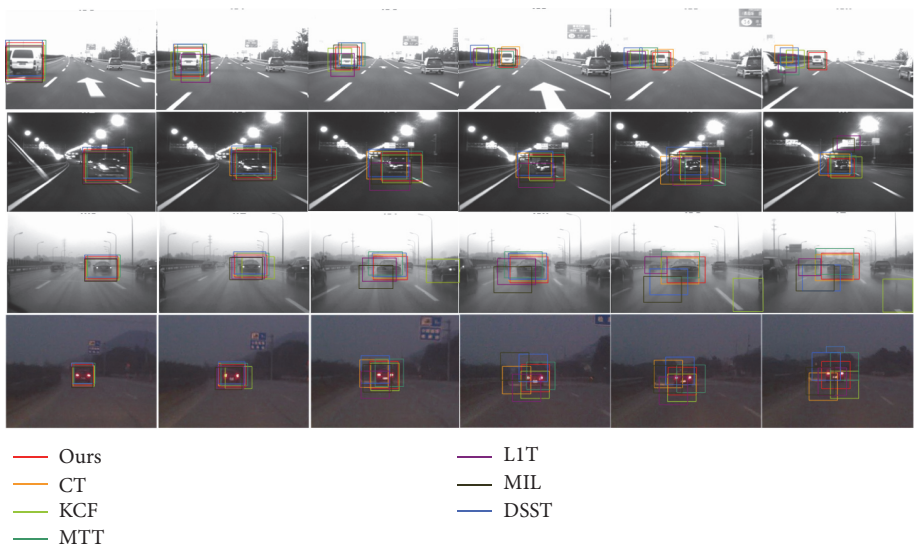


Figure 7: Vehicle tracking results of each algorithm in four typical scenarios.

distance between BB_T and BB_G, in which BB_T is the outline of the tracking box and BB_G is the ground truth of the real target outline box.

The proposed algorithm is comparable with many state-of-the-art algorithms such as MTT [10], CT [11], KCF [12], MIL [2], L1T [13], and DSST [14].

The experiment results are shown in Tables 1 and 2. It can be seen from the table that since the proposed tracking algorithm is able to generate richer deep features, the proposed method performs better than the existing state-of-the-art algorithms under the four different test conditions. The four typical scenarios that are chosen are daytime with good illumination, nighttime with road lamp, daytime with heavy raining, and twilight time. Examples of real tracking results are shown in Figure 7.

It should be mentioned that although the performance of our method is improved, the processing speed is relatively low. The average processing speed for one image is around 75 ms which is among the second worst of all the methods that are used in the experiments (Table 3). The possible reason for the big time cost is that the deep model contains a big amount of neurons and weight connections between each of neurons and the computing of weight in each connection may need much more processing time. The experimental platform was an Intel 2.67 GHz CPU with 4 GB RAM and the Windows 7 operating system.

5. Conclusion

To solve the problem of traditional tracking methods being not robust enough for vehicle tracking in ADAS, a deep representation and semisupervised onboard vehicle tracking algorithm was proposed. Relying on the strong feature extraction ability of deep modeling, this method dramatically inhibits drifting. Generally, the proposed semisupervised and deep model based tracking algorithm performs better than most of the existed methods in the merits of tracking success rate and average center pixel offset. However, the real-time performance of this work still needs to be further improved. There are two ways that are supposed to be used to solve this problem. First, concise deep model may be designed to reduce calculation burden. Second, parallel computing technology may be applied to speed the calculation process.

Competing Interests

The authors declare that they have no conflict of interests.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61601203, 61403172, and U1564201), China Postdoctoral Science Foundation (2015T80511 and

Table 1: Tracking success rate of each algorithm (%).

	MTT	CT	KCF	MIL	L1T	DSST	Our method
Video 1 (day 1)	92.4	64.7	83.2	69.7	75.8	50.2	**98.3**
Video 2 (night)	**89.6**	59.8	74.9	60.8	71.2	51.4	87.3
Video 3 (rain)	91.0	62.3	80.4	62.3	78.7	58.8	**93.5**
Video 4 (twilight)	80.4	59.8	69.9	56.6	70.5	60.7	**89.5**
Average	88.3	61.6	77.1	62.3	74.0	55.3	**92.1**

Table 2: The average center pixel offset of each algorithm (pixels).

	MTT	CT	KCF	MIL	L1T	DSST	Our method
Average center pixel offset	11.4	18.8	16.5	23.1	17.9	25.2	**8.3**

Table 3: The average processing time of each algorithm (pixels).

	MTT	CT	KCF	MIL	L1T	DSST	Our method
Average processing time (ms)	86	37	42	34	57	28	**75**

2014M561592), Key Research and Development Program of Jiangsu Province (BE2016149), Natural Science Foundation of Jiangsu Province (BK20140555), and Six Talent Peaks Project of Jiangsu Province (2015-JXQC-012 and 2014-DZXX-040).

References

[1] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proceedings of the 17th British Machine Vision Conference (BMVC '06)*, pp. 47–56, September 2006.

[2] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[3] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[4] S. Avidan, "Support vector tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1064–1072, 2004.

[5] Q. Wang, F. Chen, J. Yang, W. Xu, and M.-H. Yang, "Transferring visual prior for online object tracking," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3296–3305, 2012.

[6] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proceedings of the 10th European Conference on Computer Vision (ECCV '08)*, pp. 234–247, Springer, Marseille, France, October 2008.

[7] A. Teichman and S. Thrun, "Tracking-based semi-supervised learning," *International Journal of Robotics Research*, vol. 31, no. 7, pp. 804–818, 2012.

[8] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[9] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[10] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2042–2049, Providence, RI, USA, June 2012.

[11] K. Zhang, L. Zhang, and M. Yang, "Real-Time Compressive Tracking," in *Computer Vision—ECCV 2012*, vol. 7574 of *Lecture Notes in Computer Science*, pp. 864–877, Springer, Berlin, Germany, 2012.

[12] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[13] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust L1 tracker using accelerated proximal gradient approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 1830–1837, June 2012.

[14] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proceedings of the 25th British Machine Vision Conference (BMVC '14)*, Nottingham, UK, September 2014.