# Stable and real-time hand gesture recognition based on RGB-D data

Bo Liu, Guijin Wang*, Xinghao Chen, Bei He
Dept. of Electronic Engineering, Tsinghua University, Beijing 100084, China

## ABSTRACT

Hand gesture recognition has attracted more interest in computer vision and image processing recently. Recent works for hand gesture recognition confronted 2 major problems. The former one is how to detect and extract the hand region from color-confusing background objects. The latter one is the expensive computational cost by considering the kinematic hand model with up to 27 degrees of freedom.

This paper proposes a stable and real-time static hand gesture recognition system. Our contributions are listed as follows. First, to deal with color-confusing background objects, we take the RGB-D (RGB-Depth) information into account, where foreground and background objects can be segmented well. Additionally, a coarse-to-fine model is proposed, which utilizes the skin color and helps us extract the hand region robustly and accurately. Second, considering the principal direction of hand region is random, we introduce the principal component analysis (PCA) algorithm to estimate and then compensate the direction. Finally, to avoid the expensive computational cost of traditional optimization, we design a fingertip filter and detect extended fingers via calculating their distances to palm center and curvature easily. Then the number of extended fingers will be reported, which corresponds to the recognition result.

Experiments have verified the stability and high-speed of our algorithm. On the data set captured by the depth camera, our algorithm recognizes the 6 pre-defined static hand gestures robustly with average accuracy about 98.0%. Furthermore, the average computational time for each image (with the resolution 640×480) is 37ms, which can be extended to many real-time applications.

**Keywords:** hand gesture recognition, RGB-D data, principal component analysis, fingertip filter, coarse-to-fine model

## 1. INTRODUCTION

In the present day the efficient and intelligent human-computer interaction (HCI) is assuming to be of utmost importance in our daily lives. Human hand, compared to other body parts, is considered to be the most natural and effective input device in HCI because of its dexterous functionality in both communication and manipulation. Hand gesture recognition (HGR), which recognizes the meaningful expression of the hand posture, has been applied in numerous fields including virtual reality, medical care, home entertainment and contactless control.

Generally speaking, hand gesture recognition can be divided into two categories: glove-based methods and vision-based methods[1][2][3]. The former one employs data gloves with mechanical or optical sensors inside to collect and transduce the pose information of user's hand, including 3D location and bending angle of each finger[4]. With these complete and reliable configure status, glove-based hand gesture recognition systems can usually achieve remarkable results. However, user is required to wear a cumbersome glove with cables connect to the computer, which hinders the naturalness of HCI. On the contrary, vision-based methods adopt cameras as input devices and can provide more natural and non-invasive interaction, while suffer from low accuracy and processing speed.

Traditional vision-based hand gesture recognition faces two major problems. The former one is how to detect and extract the hand region from color-confusing background objects. Some employ the skin color segmentation and background modeling to search the hand, but they are sensitive to illumination variation and motion in the scene[9][10][13][14]. The latter one is the expensive computational cost by considering the kinematic hand model, since the hand is a non-rigid body with up to 27 degrees of freedom (DOFs) according to the hand anatomy[5]. Furthermore, sophisticated optimization methods are applied to solve the problem, which restricts the real-time applications.

*wangguijin@tsinghua.edu.cn; phone 1 891 138-9502

2013 International Conference on Optical Instruments and Technology: Optoelectronic Imaging and Processing Technology, edited by Xinggang Lin, Jesse Zheng, Proc. of SPIE Vol. 9045, 90450L
© 2013 SPIE · CCC code: 0277-786X/13/$18 · doi: 10.1117/12.2038084

In this paper we propose a stable and real-time static hand gesture recognition system. First, taking into account the RGB-D information acquired from the Kinect sensor, the hand region can be segmented well without wearing auxiliary equipment. Then a coarse-to-fine skin color model is presented, which can describe the accurate skin color distribution adaptively, to help us verify and refine the hand region. Second, considering the principal direction of hand region is random, we introduce the PCA algorithm to estimate and then compensate the direction. Therefore, the stable silhouette of the hand region can be extracted. Finally, to avoid the expensive computational cost of the traditional optimization, we novelly design a fingertip filter and detect extended fingers via calculating their distances to the palm center and curvature values easily. Then the number of extended fingers will be reported, which corresponds to the hand gesture recognition result.

The rest of this paper is organized as follows. In Section 2, related works are reviewed in brief. Two main modules, i.e., hand segmentation and hand gesture recognition are presented in Section 3 and 4 respectively. We show experiments and discussions in Section 5, followed by a concise conclusion in Section 6.

## 2. RELATED WORKS

Traditional vision-based HGR systems employ optical sensors to acquire RGB images of the scene. Skin color segmentation and background modeling are used to detect the hand region[9][10][13][14]. However, these methods are sensitive to illumination variation and motion in the scene. Many researchers employ color markers attached to hand and fingertips to enhance the hand segmentation and finger locating. Chua et al.[6] proposed an algorithm to estimate the 3D hand posture using a single 2D image, in which eight color markers were used to identify the feature points. Similarly, Lamberti and Camastra[7] exploited a color glove and achieved a recognition rate of 97.79% with 13 gestures. These color markers are much lighter and cheaper than the data glove, but they still put extra constraints on the user.

Compared to RGB images, depth images generated by IR-based[8] or ToF cameras are insensitive to light condition and cluttered background, thus can make the hand segmentation more feasible and robust. Bergh and Gool[10] employed both RGB and ToF cameras to build a real-time 3D hand gesture interaction system, in which a hybrid skin color segmentation and depth-based segmentation were combined to detect and extract the hand region. The ToF cameras are usually expensive and a calibration step is needed to project the ToF image onto the RGB image coordinates. Zeng et al.[11] proposed a real-time presentation system controlled by hand gestures. The system adopted a thermal camera for robust human body segmentation to handle the complex background and varying illumination.

An increasing number of HGR systems[12][13][14][15][16] make use of Microsoft's Kinect sensor to enhance recognition since its release in 2010. Ren et al.[12] presented a novel dissimilarity metric, named finger-earth mover's distance, for hand gesture recognition using Kinect. They assumed the hand is the frontmost object facing the sensor and required the user to wear a black belt on the wrist to assist the hand segmentation. The mean accuracy is 90.6% on a 10-gesture dataset, while the mean running time is 0.5004s. Oikonomidis et al.[13] proposed a model-based method for recovering and tracking the 3D position, orientation and full articulation of a human hand using data acquired by a Kinect sensor. The GPU-based implementation of this method achieved accurate and robust 3D tracking of hand articulations in near real-time (15Hz).

## 3. HAND SEGMENTATION

The first and most influential task of a hand gesture recognition system is the hand segmentation. Robust and precise hand segmentation results can greatly facilitate the following modules, while inferior segmentation results may increase the difficulty of recognition.

We use Microsoft's Kinect sensor as the input device, which can generate both RGB image and depth image at 640×480 resolution within the same coordinate system.

### 3.1 Depth Thresholding

We make a reasonable assumption that the user sits facing the sensor within 0.5m to 1.5m and keeps his/her hand in front of the body. Therefore, only the objects located in the valid depth range will be taken into consideration, and these pixels will be divided into background and foreground according to their depth values.

By calculating the histogram of valid depth values and then seeking the two consecutive bins with the biggest count, a depth threshold can be obtained. As shown in Fig. 1(a), the biggest two consecutive bins correspond to the background (i.e., user's body and other objects such as the chair), while the regions with smaller depth values correspond to the foreground (i.e., user's hand, see in Fig. 1(c)).

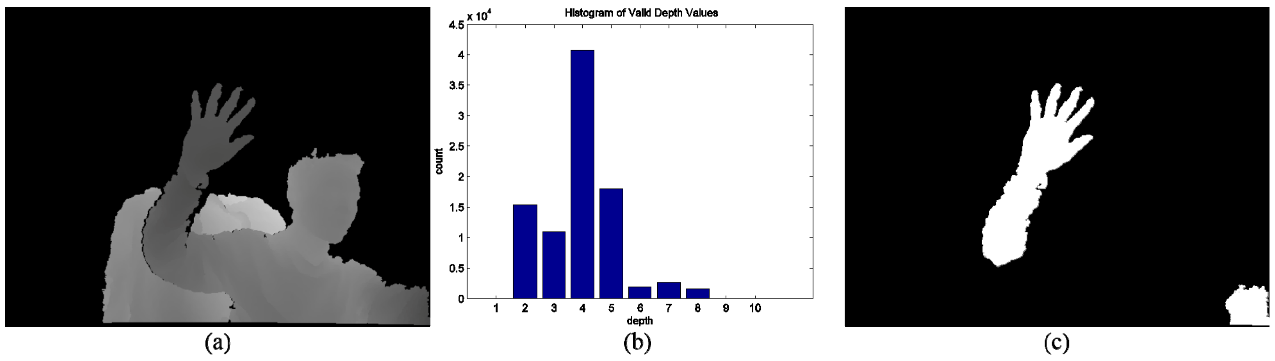


Fig. 1. (a) The valid depth image. (b) The histogram of valid depth values. The valid depth range is from 0.5m to 1.5m, and is converted linearly to an integer range (from 0 to 200). (c) The foreground image after depth thresholding.

## 3.2 Blob Clustering

The depth image captured by Kinect will encounter the "depth holes" problem, caused by the occlusion regions and infrared-absorbed surfaces. Hence, there may exist small holes or even isolate fingers in the hand mask image obtained in the previous step, as shown in Fig. 2(a). Morphological operations or other filtering can fill these holes and concatenate the isolate fingers, but at the meantime lead to much information loss, since the hand shape deforms and contour degrades afterwards.

Using connected components analysis, we can retrieve the foreground blobs that indicate the hand, possible isolate fingers and other disruptive objects. Holes inside these blobs can be filled according to their external contours, while isolated fingers can be connected to the hand using single linkage clustering. We calculate the minimum Manhattan distance between each two blobs, and merge them if the distance is less than a certain threshold, as shown in Fig. 2(b).

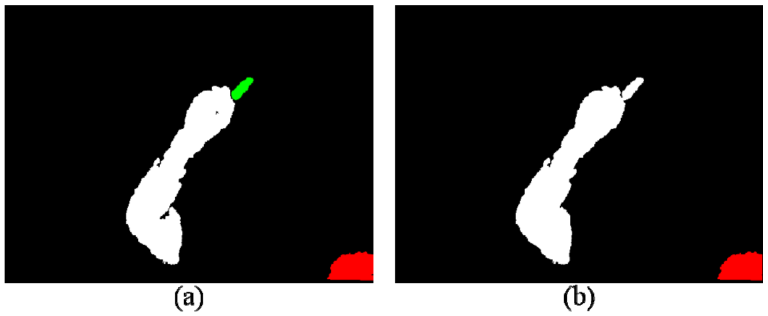


Fig. 2. (a) The small holes and isolate fingers in depth image. (b) The foreground image after blob clustering.

## 3.3 Skin Color Refinement

Other objects placed near the sensor are also considered to belong to the foreground. Moreover, the sleeve that covers the arm sometimes hinders the following palm center locating procedure. Therefore, we adopt the skin color information to verify and refine the hand segmentation result.

Skin color is a very important and intuitive feature of human body, including face and hand. Using a well-trained skin color model, we can easily distinguish the human body from various other objects. Nevertheless, skin color models are highly sensitive to illumination conditions, meanwhile other skin colored objects in the background may affect the segmentation result. To cope with the problems mentioned above, we propose a coarse-to-fine skin color model, which can describe the accurate skin color distribution according to the current scene adaptively.

After converting the RGB image to YCrCb color space, we use a Gaussian model to represent the skin color. Specifically, at the "coarse" stage, we employ a broad Gaussian model which can be trained offline to segment the skin color regions. The variance of this Gaussian model is relatively large, so that it can detect a wide range of skin colors under various lighting conditions. Then at the "fine" stage, combining the skin color regions with the foreground, we can extract the rough hand part and using pixels inside to train a more precise and adaptive Gaussian model with a much smaller variance. Fig. 3(b) shows the coarse skin color segmentation result, while Fig. 3(c) shows the fine one. With this fine Gaussian skin color model, we can check whether each foreground blob belongs to the hand, and eliminate the sleeve when calculating the palm center. Furthermore, to fit the possible illumination variation in the scene, the fine skin model is maintained and updated at set intervals.

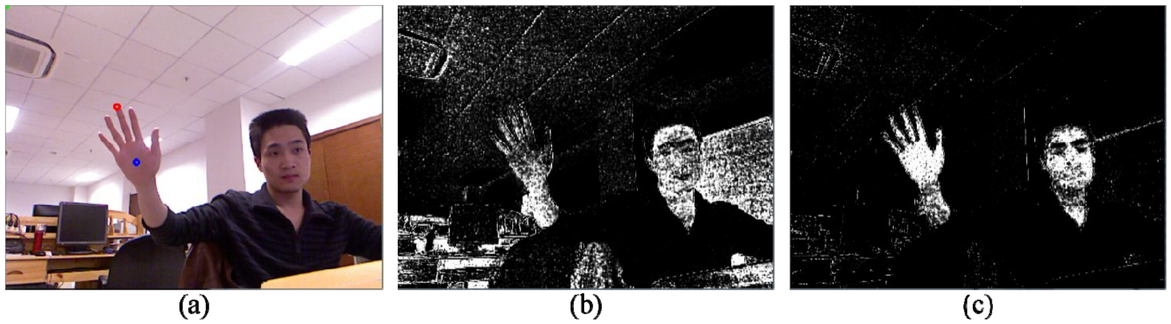


Fig. 3. (a) The RGB image. (b) The skin color region using coarse model. (c) The skin color region using fine model.

## 4. HAND GESTURE RECOGNITION

After getting the hand mask, we can extract useful geometric features and then detect the fingertips.

### 4.1 Arm Orientation Correction

Firstly, applying PCA to the thinned hand mask, we can estimate the tilt angle of the arm, followed by axis correction which rotates the image to make the principle axis of the arm vertical and hence provides orientation invariance to a certain extent. Secondly, exploiting the distance transform, the point with maximal distance to the nearest boundary of the palm can be located, which we define as the palm center. In the meantime, the wrist can be found since the minimal palm center-boundary distance (which we call "palm radius") and hand orientation have already been calculated. Fig. 4(a) shows the original hand mask, while Fig. 4(b) shows the axis correction result. The palm center and wrist are shown in Fig. 4(c), which displays the result after distance transform.

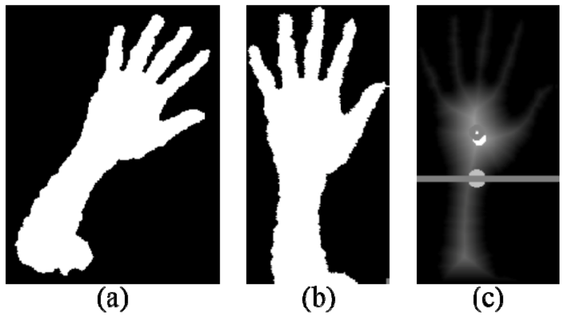


Fig. 4. (a) The original hand mask image. (b) The hand mask after axis correction. (c) The distance transform result.

## 4.2 Fingertip Filter

We design a fingertip filter to detect extended fingers based on their distances to palm center and curvature values.

At first, we calculate the distance and angle from each boundary point to palm center. The local maximums of the distances indicate the extended fingertip locations. Adopting the local and global constraint in Eq.1, the noisy extreme points can be eliminated and only the stable ones can be obtained.

$$\begin{cases} \text{local} \ : R_i > R_j, j \in \mathbf{N}(i) \\ \text{global} : R_i > R_{\min} \bullet k(\theta_i) \end{cases} \tag{1}$$

where $R_i$ and $\theta_i$ are the distance and angle from the $i^{th}$ boundary point to the palm center respectively, $\mathbf{N}(i)$ denotes the neighborhood of the $i^{th}$ point, $R_{\min}$ represents the minimal palm center-boundary distance, and $k(\theta_i)$ refers to an angle-related threshold ratio which decreases as the $\theta_i$ deviates from the vertical direction, as shown in Eq.2.

$$k(\theta_i) = k_{init} - \lambda \bullet |\theta_i - \Theta| \tag{2}$$

where $k_{init}$ denotes the initial threshold ratio, $\lambda$ is a weighting coefficient, and $\Theta$ represents the tilt angle of the arm which is 90° after arm orientation correction. Fig. 5(a) illustrates the global distance threshold which is more suitable for fingertips locating, since the shape of the open palm is more like an ellipse rather than a circle.

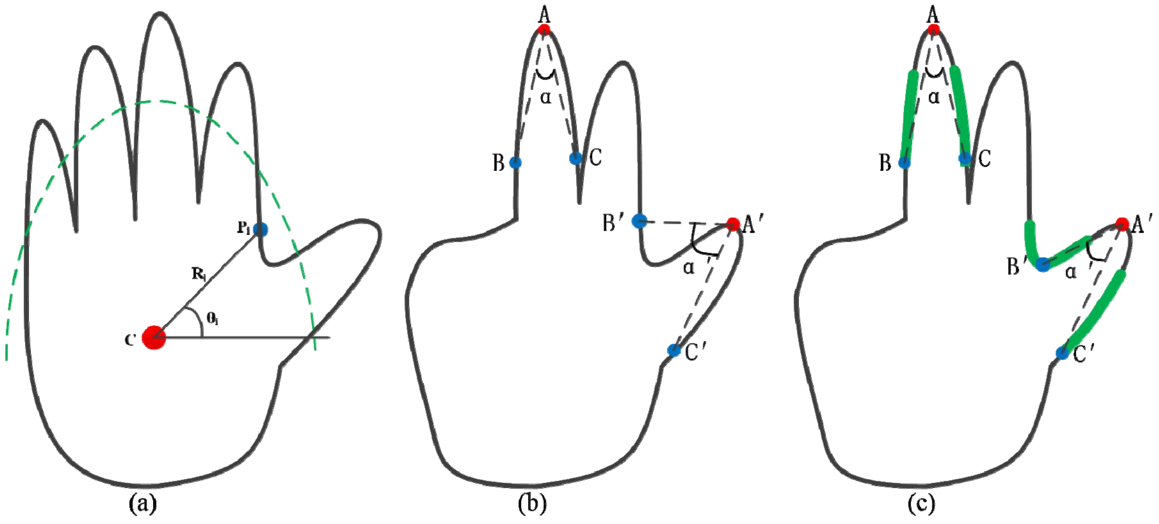


Fig. 5. (a) Global constraint: the green dash line refers to the distance threshold which changes along with $\theta_i$. Schematic of the definition of curvature: (b) old one, where $\alpha'$ is much larger than $\alpha$; (c) new one which is more stable and effective.

Furthermore, we introduce the curvature to improve the fingertip location procedure. Typically two auxiliary points on either side of the point of interest on the curve are required to calculate the curvature value, as shown in Fig. 5(b). The curvature of point A is represented by the included angle $\alpha$, i.e., smaller $\alpha$ value corresponds to larger curvature.

However, the increment between the current point and auxiliary point is generally fixed and hard to choose. The global curve information won't be obtained and many burrs will arise if the increment is too small, while curvature values will decrease significantly if too large, as shown in Fig. 5(b).

To deal with this problem, we modify the previous method to calculate the curvature stably and effectively. At first we set the increment as a proportion (10% in our experiments) of the overall length of the palm silhouette, thus making the results robust to the 3D translation and shape deformation. Then we scan the boundary points around both the left and right auxiliary point to find the new auxiliary point pair whose corresponding included angle $\alpha$ reaches the maximum, as shown in Fig. 5(c).

The local maximums of the distance form the initial fingertip point set, while the curvature values of these points are employed to check their validity, as shown in Eq.3. Only the points with both extreme distance to palm center and large enough curvature are considered to be the fingertips. Finally, all extended fingertips are detected and the count refers to the recognition result, as shown in Fig. 6.

$$\begin{cases} \text{local} & : R_i > R_j, j \in \mathbf{N}(i) \\ \text{global} & : R_i > R_{\min} \bullet k(\theta_i) \\ \text{curvature} & : \alpha_i < \alpha_{th} \end{cases} \tag{3}$$

where $\alpha$ is related to the curvature value of the $i^{th}$ boundary point, and $\alpha_{th}$ denotes a predefined threshold value.

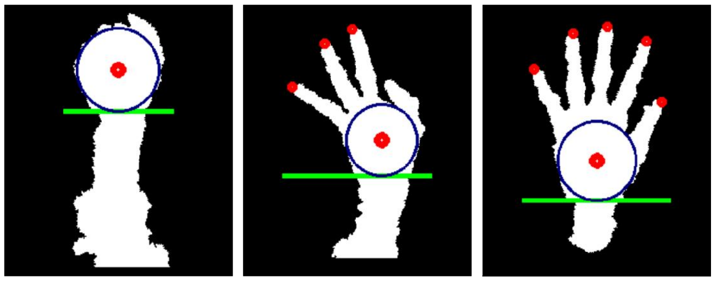


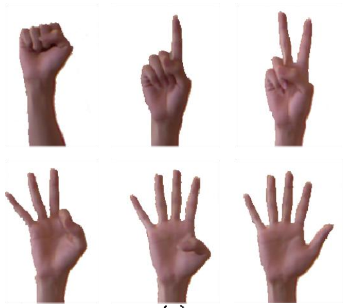
Fig. 6. The location of palm center, wrist and extended fingertips.

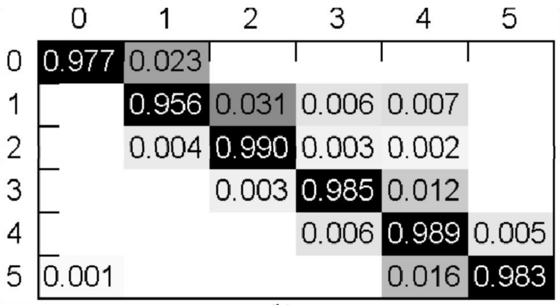
## 5. EXPERIMENTS AND DISCUSSION

The experimental environment is a Windows PC with Intel quad core 2.50GHz CPU, and OpenNI SDK is used to acquire RGB image and depth image from Kinect sensor. Our system is implemented in C++ code.

### 5.1 Dataset

The test dataset we used is collected from 10 subjects, which contains 6 kinds of hand gestures, as shown in Fig. 7(a). Each subject is requested to perform the same gesture in 300 different trials. The total number of test samples in our dataset is then 10×6×300 = 18000, and each sample consists of a pair of RGB and depth image.


(a)



| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0.977 | 0.023 | | | | |
| 1 | | 0.956 | 0.031 | 0.006 | 0.007 | |
| 2 | | 0.004 | 0.990 | 0.003 | 0.002 | |
| 3 | | | 0.003 | 0.985 | 0.012 | |
| 4 | | | | 0.006 | 0.989 | 0.005 |
| 5 | 0.001 | | | | 0.016 | 0.983 |

(b)

Fig. 7. (a) 6 kinds of hand gestures; (b) confusion matrix of our recognition algorithm.

Within the 300 trials of each gesture performed by each subject, the gesture configurations (e.g. 3D location and orientation of the arm, twist angle of the wrist, abduction/adduction of each finger, illumination condition) vary moderately to simulate the real application scenarios, which makes our dataset a very challenging one.

### 5.2 Robustness

At first, our system is robust to the cluttered background and motion in the scene, since we extract the hand region mainly in depth image and the background objects behind the hand can be easily eliminated. Users are not requested to wear any auxiliary equipment, or put his/her hand in a restricted and narrow depth range. Besides, our method is insensitive to the illumination variation in the scene because of the coarse-to-fine skin model with its updating process. Furthermore, the tilt angle of the arm can be compensated, which provides orientation invariance to some degrees. Finally, the fingertip filter we proposed can detect the stable extended fingers using both distance and curvature information, while ignoring the noisy peaks on the bumpy edge of the hand in the depth image.

### 5.3 Accuracy and Efficiency

Theoretically there can be 32 different hand gestures with 5 fingers, while we only consider 6 kinds of them so far because they are most common used. Fig. 7(b) shows the confusion matrix of our hand gesture recognition system. The accuracy of each pre-defined gestures ranges from 95.6% to 99.0%, while the mean accuracy is 98.0%. The values which are not on the diagonal indicate the confusion level between each two gestures, e.g., a sample of gesture 2 is likely to be classified into gesture 1 if the middle finger is too close to the index finger. The average running time of each frame (with the resolution 640×480) is 37ms, which can be extended to many real-time applications.

## 6. CONCLUSION

Hand gesture recognition plays an important role in the field of HCI. In this paper we propose a stable and real-time static hand gesture recognition system. First, with the RGB-D data acquired from the Kinect sensor, we can segment the hand region under scenes of different illumination variation, camera motion and color-confusing background objects. Second, a coarse-to-fine skin color model which is insensitive light condition is presented to verify and refine the hand region. Third, the direction of the arm can be estimated and compensated, and the stable silhouette of the hand region can then be extracted. Finally, a fingertip filter is designed to detect extended fingers using their distance and curvature information. Experiments verify the robustness, accuracy and efficiency of our hand gesture recognition system. In the future, we will also focus on dynamic hand gesture recognition and utilize the temporal information to improve the stability and accuracy of the proposed algorithm further.

## ACKNOWLEDGE

The work is partially sponsored by National Natural Science Foundation of China (No.61132007 and No.61271390).

## REFERENCES

[1] Erol, A., Bebis, G., Nicolescu, M., Boyle, R. D., and Twombly, X., “Vision-based hand pose estimation: A review,” Computer Vision and Image Understanding 108(1), 52–73 (2007).

[2] Mitra, S. and Acharya, T., “Gesture recognition: A survey,” Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 37(3), 311–324 (2007).

[3] Khan, R. Z. and Ibraheem, N. A., “Survey on gesture recognition for hand image postures,” Computer and Information Science 5(3), p110 (2012).

[4] Sturman, D. J. and Zeltzer, D., "A survey of glove-based input," Computer Graphics and Applications, IEEE 14(1), 30–39 (1994).

[5] Albrecht, I., Haber, J., and Seidel, H.-P., "Construction and animation of anatomically based human hand models," in [Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation], 98–109, Eurographics Association (2003).

[6] Chua, C.-S., Guan, H., and Ho, Y.-K., "Model-based 3d hand posture estimation from a single 2d image," Image and Vision computing 20(3), 191–202 (2002).

[7] Lamberti, L. and Camastra, F., "Real-time hand gesture recognition using a color glove," in [Image Analysis and Processing–ICIAP 2011], 365–373, Springer (2011).

[8] Wang, G., Yin, X., Pei, X., and Shi, C., "Depth estimation for speckle projection system using progressive reliable points growing matching," Applied optics 52(3), 516–524 (2013).

[9] Sha, L., Wang, G., Yao, A., Lin, X., and Chai, X., "Hand posture recognition in video using multiple cues," in [Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on], 886–889, IEEE (2009).

[10] Van den Bergh, M. and Van Gool, L., "Combining RGB and ToF cameras for real-time 3d hand gesture interaction," in [Applications of Computer Vision (WACV), 2011 IEEE Workshop on], 66–72, IEEE (2011)

[11] Zeng, B., Wang, G., and Lin, X., "A hand gesture based interactive presentation system utilizing heterogeneous cameras," Tsinghua Science and Technology 17(3), 329–336 (2012).

[12] Ren, Z., Yuan, J., and Zhang, Z., "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in [Proceedings of the 19th ACM international conference on Multimedia], 1093–1096, ACM (2011).

[13] Oikonomidis, I., Kyriazis, N., and Argyros, A. A., "Efficient model-based 3d tracking of hand articulations using kinect," in [BMVC], 1–11 (2011).

[14] Park, S., Yu, S., Kim, J., Kim, S., and Lee, S., "3d hand tracking using kalman filter in depth space," EURASIP Journal on Advances in Signal Processing 2012(1), 1–18 (2012).

[15] Tang, M., "Recognizing hand gestures with microsofts kinect," Web Site: http://www.stanford.edu/class/ee368/Project_11/Reports/Tang_Hand_Gesture_Recognition.pdf (2011).

[16] Li, Y., "Hand gesture recognition using kinect," in [Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on], 196–199, IEEE (2012).