Hindawi
Advances in Multimedia
Volume 2017, Article ID 2162078, 25 pages
https://doi.org/10.1155/2017/2162078

Research Article

Hand Motion and Posture Recognition in a Network of Calibrated Cameras

Jingya Wang and Shahram Payandeh

Network Robotics and Sensing Laboratory, School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada V5A 1S6

Correspondence should be addressed to Shahram Payandeh; payandeh@sfu.ca

Received 29 April 2017; Accepted 30 August 2017; Published 31 October 2017

Academic Editor: Kjell Brunnström

This paper presents a vision-based approach for hand gesture recognition which combines both trajectory and hand posture recognition. The hand area is segmented by fixed-range CbCr from cluttered and moving backgrounds and tracked by Kalman Filter. With the tracking results of two calibrated cameras, the 3D hand motion trajectory can be reconstructed. It is then modeled by dynamic movement primitives and a support vector machine is trained for trajectory recognition. Scale-invariant feature transform is employed to extract features on segmented hand postures, and a novel strategy for hand posture recognition is proposed. A gesture vector is introduced to recognize hand gesture as an entirety which combines the recognition results of motion trajectory and hand postures where a support vector machine is trained for gesture recognition based on gesture vectors.

1. Introduction

It was shown that nearly 90% of daily communication is nonverbal [1]. Hand gesture has always been a powerful communication tool in people's daily life. As the technology is developing, hand gesture recognition is becoming an important component in innovative applications, such as human computer interface, robotic tele-control, and sign language interpretation. In this paper, hand gesture refers to hand moving trajectory and hand posture stands for the hand shape and appearances.

Decades ago, the interaction between human and computers was through command-line interface through keyboard entries. Through technological advancements, other types of user interface devices have been introduced (such as the computer mouse) which can offer alternatives to the traditional design of human computer interfaces. In addition, it was shown that 65% of human communication involves nonverbal gestures [2]. For hearing-impaired persons, this percentage rises to nearly 100%. Therefore, hand gestures, as one of the most dominant communication tools in our daily life, can express an enriched mode of communication with abundant coded messages. Developing a hand gesture interface to enrich the human computer interface not only offers a more convenient approach for hearing impaired but also can enhance and extend the current and existing modes of interfaces.

Hand gesture recognition can be divided into two main categories: device-based and ambient-based. Device-based hand gesture recognition requires the user to wear devices such as gloves, markers, or other tools in order to acquire hand or arm joint angles corresponding to their spatial position [3–6]. A data glove, modeling a 3D hand is designed in [7]. This glove measures the angles of finger bending using analog flex sensors. A multicolored glove is adopted in [5] to reconstruct the pose of hand based on its color pattern. Due to the current advancements in design of sensors, device-based hand gesture recognition collects relatively accurate information of the hand gestures. In addition, such approach for gesture recognition is robust to illuminate changes which is the main drawback of various vision-based hand gestures recognition.

For ambient-based hand gesture recognition, sensors capture images of the scenes and process the required information needed for determining hand movements and appearances. For RGB type sensors, gesture information

is mainly dependent on the hand color or texture [8–12]. A drawback of the RGB-based system is its sensitivity to illumination variations. Therefore, color spaces such as HSV [13], YCbCr [14], CIE Lab, or CIE Luv can be utilized. For classification, methods such as Bayesian classifier with the histogram technique [15] and Gaussian classifiers [16, 17] were usually introduced. There are other methods for hand gesture recognition. For example, in order to address some of the limitations associated with the RGB imaging, depth sensors were introduced in order to capture hand motion [18–20]. The output of the depth sensor can be encoded as a gray-scale image, where the intensities correspond to the distances between the objects and the camera. For hand gesture recognition, since the hand is usually the closest moving object to the camera, a suitable threshold can be defined on gray-scale images in order to eliminate any background noises. IR camera combined with retroreflective markers is also used in order to determine the position and posture of the hand in view of a single camera [21]. In this paper, two calibrated RGB cameras are employed to record hand gestures. These cameras are utilized under a steady lighting condition. Due to exposure compensation of the cameras, YCbCr color space is adopted for hand area segmentation.

Hands, as one of the most dexterous part of the human body, have 27 degrees of freedom which can occupy various shapes and appearances. To extract hand region from the rest of the image, color cues and motion cues are most often used to segment a hand from the background. The skin color is usually more distinctive and less sensitive to illumination changes in the hue-saturation space than in RGB color space [9]. Most of the color segmentation approaches rely on histogram matching [23, 24]. Color cue is not robust to illumination variation and frequently results in undetected skin regions or falsely detected nonskin area. To solve such problem, some assumptions such as area size (scale filter) or certain spatial position (position filter) are adopted. Another solution to such problem is to allow the users to wear gloves having distinctive colors [5] or special markers (LED light [3, 25], fluorescent material [26]) or clean the background so there is not much noise from it [12]. These methods are robust to illumination variation but lose the intention of liberating hand from gloves. Motion cues are usually used as one of the main components for segmenting moving objects such as hands or arms in image frames. They can also be used in segmenting hand gestures from a stationary background [11, 27, 28].

Feature extraction is very important to posture recognition. The simplest and most frequently used features are hand silhouette which can be easily extracted. Contours are a group of commonly used features. Several different edge detection schemes can be used to produce contours [9]. Contours are often employed with 3D hand model that is build based on hand shape and structure. Hand posture can be recognized by comparing the similarities between detected contours and generated contours based on the hand model [29, 30]. In [29], they build a 3D hand model with 27 degrees of freedom to model the articulates of hand. The hand gesture recognition is done by comparing the generated contours based on the hand model and the input hand images. Another frequently used feature in posture recognition is figure tip. Postures can be recognized based on the positions of five finger tips, extracted by either markers (LED light [3, 25] or spatial color) or convex hull on silhouette [31]. There are also other feature detectors that can be applied to achieve posture recognition, such as scale-invariant feature transform (SIFT) which is insensitive to illumination variations, scale, and orientation changes [32–35]; Haar-like feature which transforms hand postures into a coefficients vector in the Haar wavelet transform [36]; and orientation histogram [37].

Hand gesture is represented in four aspects: hand shape, position, orientation, and movement [38]. The same semantic paths are usually made in different scales, speed, and shapes due to the individual differences. As a statistical model, Hidden Markov model (HMM) has been found efficient in modeling spatiotemporal time series where the same gesture has different shape and duration [39, 40]. Other features extraction methods such as Gaussian mixture model and principle component analysis [41, 42] can be used to enhance HMM recognition process. Finite state machine (FSM) is similar to HMM which models hand movement as an ordered sequence of states in a spatiotemporal configuration space [8, 43]. Dynamic movement primitives (DMP) proposed by [44] is employed in [43] for 2D trajectory recognition and has obtained an impressive accuracy of 98.06%. DMP encodes gesture paths into weight vectors which preserve the topological structures of paths. The benefits of DMP are being (a) robust to the spatiotemporal variations in gesture paths and (b) easy to adjust the dimension of the weight vector based on the gesture paths complexity to adapt different applications.

Hand posture recognition is another key part of gesture recognition. Template matching is a simple method of posture recognition, and it is easy to add or remove template classes. To extract features on hand posture, a convex hull on silhouette [31] or fingertip detection using circular mask as a correlation techniques [45] can be employed. However, these recognition approaches based on hand silhouette or contour usually require a clean background where the hand can be well segmented. In our case, hand posture is made in a cluttered and moving background and segmented using YCbCr color space. Sometimes the hand cannot be well segmented from the background. In such case, hand can be taken as partially occluded. Therefore, a feature detector that is insensitive to illumination variation and partially occlusion is needed. Scale-invariant feature transform (SIFT) can be such a feature detector and descriptor which is also robust to scale and orientation changes. In addition, it is robust to affine distortion within some ranges that can benefit posture recognition. This is due to the fact that the relative position between hands and cameras changes which can cause affine distortion between the input hand postures and posture templates. In our work, SIFT is used for feature detector and posture recognition. Combining recognition results of gesture path and hand postures, a gesture vector is proposed for gesture recognition [46].

In this paper, hand gesture is referred to hand moving trajectory and hand posture which stands for the hand shape and appearance. Hand movement is also called gesture

path. Two calibrated cameras are employed to record hand gestures. These cameras are utilized under a steady lighting condition. Due to exposure compensation of the cameras, YCbCr color space is adopted for hand area segmentation. This paper also only focuses on hand shape and movement since hand position and orientation are involved with body context which is not considered in our work.

2. Preliminaries

This section presents preliminary analysis and results which are obtained pertaining to the main contributions of the paper. Two preliminaries, 3D coordinate reconstruction and hand segmentation, are introduced. 3D coordinate reconstruction method of a point in world space using two calibrated cameras is presented in Section 2.1. Section 2.2 introduces and experimentally compares several approaches of skin color segmentation and the scheme of extracting hand area from the background.

2.1. Camera Calibration. Two calibrated cameras are employed to capture hand motion made in the overlapping field of view to reconstructing 3D hand motion trajectory. The setup of the two cameras is shown in Figure 1. The relationship of coordinates between the cameras and world are marked in Figure 2. Based on the pinhole camera model, the relationship between a point Q in the world coordinates and its projection q_1 on the image plane of camera C_1 is shown in (1). The relationship between projection q_2 in camera C_2 and the same point Q is formulated in (2).

$$q_1 = K_1 [R_1 \mid t_1] Q, \tag{1}$$

$$q_2 = K_2 [R_2 \mid t_2] Q. \tag{2}$$

The intrinsic matrices, K_1 and K_2, are separately calculated by Camera Calibration Toolbox for MATLAB [47] on both cameras. The extrinsic matrices, $[R_1 \mid t_1]$ and $[R_2 \mid t_2]$, represent the rotation R and translation t between the world coordinate system and the camera coordinate system, C_1 and C_2, respectively.

The objective of this section is to reconstruct the location of a point in world space using its projection on image planes taken by the two calibrated cameras. The objective is to reconstruct the coordinates of Q based on q_1 and q_2 in (1) and (2) (this would result in 4 equations and 3 unknowns). The coordinates of Q that have three unknowns can be calculated. Based on the geometry of our experimental setup, the specific values of K_1, K_2, R_1, R_2, t_1, and t_2 are computed as

$$K_1 = \begin{bmatrix} 561 & 0 & 338 \\ 0 & 561 & 230 \\ 0 & 0 & 1 \end{bmatrix},$$

$$K_2 = \begin{bmatrix} 560 & 0 & 339 \\ 0 & 560 & 220 \\ 0 & 0 & 1 \end{bmatrix},$$

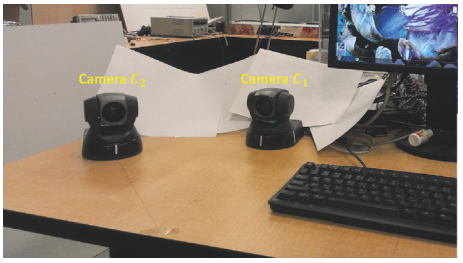


Figure 1: The setup of the calibrated cameras.

$$R_1 = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix},$$

$$t_1 = \begin{bmatrix} 0 \\ 9 \\ 44 \end{bmatrix} (\text{cm}),$$

$$R_2 = \begin{bmatrix} 0.707 & -0.707 & 0 \\ 0 & 0 & -1 \\ 0.707 & 0.707 & 0 \end{bmatrix},$$

$$t_2 = \begin{bmatrix} 0 \\ 9 \\ 40 \end{bmatrix} (\text{cm}). \tag{3}$$

For the experimental evaluation, a box made by transparent plastics is placed in the overlapping view of both cameras. The dimensions of the box are marked in Figure 3. Figure 4 shows the two different views captured by the cameras.

Table 1 lists the coordinates of eight vertices of the box in both image planes and the reconstructed 3D coordinate in the physical world. The ground truths of the eight box corners are also listed in this table. To eliminate any displacement error between coordinate systems, the relative locations between eight corner points on the box are calculated and compared with their ground truths. The results are listed in Table 2 which also showing the relative error between the values.

2.2. Hand Segmentation. An efficient hand segmentation method is the key of success towards visual tracking and further posture recognition. There can exist a large variety of hand appearances in different postures, angles, and orientations. Color cue is an efficient tool to identify the hands from the background. However, segmenting hand from a clutter background is very challenging [48] due to existence of different skin colors associated with people where the color of skin can also change under different illumination. In this section, a proper color space to represent skin colors is explored; a position and size constraint is added to locate hand area.

Given a unicolor or distinct colors background from the hand area, the hand area can be segmented by thresholding

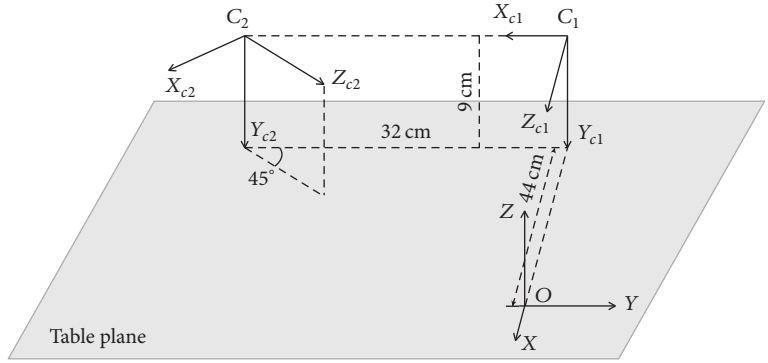


Figure 2: Schematics of spatial relationship of camera C_1 and camera C_2 with respect to world coordinates with origin at O.

Table 1: The 3D coordinates of eight corner points on the plastic box in the world coordinate system recovered by their projected points on the image plane of two cameras.

C_1 (pixel)	C_2 (pixel)	Recovered 3D coordinate (cm)	Ground truth (cm)
$b1 = [334, 334]^T$	$b1 = [293, 375]^T$	$b1' = [-3.9, 0.3, -0.5]^T$	$b1 = [-3, 0, 0]^T$
$b2 = [334, 160]^T$	$b2 = [291, 173]^T$	$b2' = [-4.0, 0.4, 12.8]^T$	$b2 = [-3, 0, 13.1]^T$
$b3 = [468, 158]^T$	$b3 = [412, 161]^T$	$b3' = [-3.6, -9.3, 12.9]^T$	$b3 = [-3, -9.8, 13.1]^T$
$b4 = [471, 341]^T$	$b4 = [410, 406]^T$	$b4' = [-3.4, -9.5, -0.3]^T$	$b4 = [-3, -9.8, 0]^T$
$b5 = [335, 335]^T$	$b5 = [323, 368]^T$	$b5' = [-1.2, 0.3, -0.4]^T$	$b5 = [0, 0, 0]^T$
$b6 = [333, 165]^T$	$b6 = [322, 182]^T$	$b6' = [-1.2, 0.5, 12.6]^T$	$b6 = [0, 0, 13.1]^T$
$b7 = [460, 166]^T$	$b7 = [442, 167]^T$	$b7' = [-0.7, -9.3, 12.7]^T$	$b7 = [0, -9.8, 13.1]^T$
$b8 = [463, 333]^T$	$b8 = [438, 396]^T$	$b8' = [-1.4, -9.5, -0.3]^T$	$b8 = [0, -9.8, 0]^T$

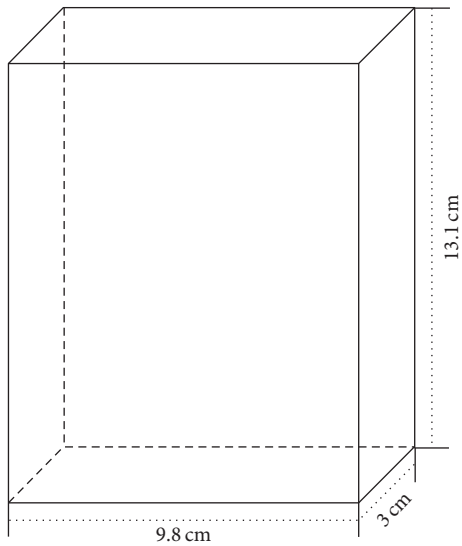


Figure 3: Dimensions of the transparent box.

the background color. For cluttered background, there are multiple colors included inside the camera view. Human skin has relatively consistent colors which are distinct from the colors of many objects [49]. Therefore, skin color can be an essential cue to separate the hand area from the background. A suitable color space and a classification algorithm are essential for successful and efficient skin segmentations. Skin segmentations have adapted many color spaces in the previous researches [13–15]. Red-Green-Blue (RGB) color space is sensitive to illumination variations, which is less efficient for hand segmentation. Hue-Saturation-Value (HSV) [13] and YCbCr [14] color spaces are more robust comparing to RGB, thus widely used in skin segmentations with various lighting conditions.

There exist different classification algorithms, such as piecewise linear classifiers [14, 50] and Bayesian classifier with the histogram technique [15] for skin segmentation. An analysis and comparison of skin segmentation using color pixel classification are carried by [49] demonstrating the performance of different skin color representation and classification. In addition, the classifier based on Bayesian RGB, 3D Gaussian mixture (RGB), and fixed-range CbCr have shown to all obtain good performances on a common data set [49]. This study was reproduced on our experimental setup based on three skin color classifiers. The Bayesian RGB classifier and 3D RGB Gaussian mixture classifier are trained using the methods and data reported in [15]. The fixed-range CbCr classifier ($77 \leq Cb \leq 127$ and $133 \leq Cr \leq 173$) is obtained using the method reported in [14]. Figure 5(a) shows a typical frame in a gesture video. The skin segmentation results using Bayesian RGB, 3D RGB Gaussian mixture, and fixed-range (CbCr) classifiers are shown in Figures 5(b), 5(c), and 5(d), respectively. The fixed-range CbCr classifier gave the best result among the three classifiers, from the accurate skin segmentations represented in Figure 5.

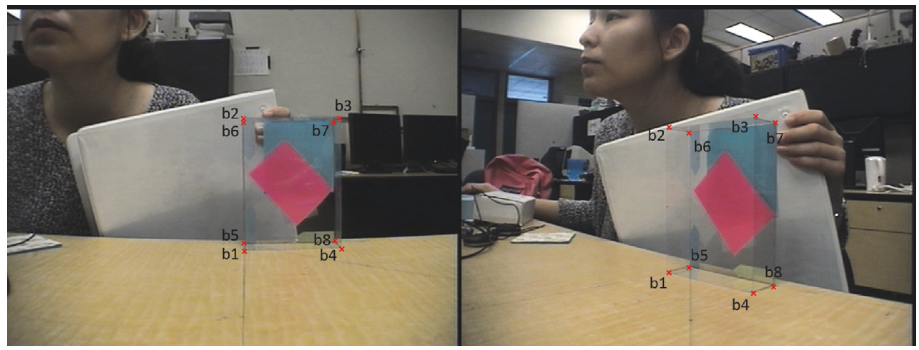


Figure 4: The transparent box in the view of two cameras. The eight corners are marked with red crosses.

Table 2: The relative location relationship between eight corner points.

Points relationship	Recovered distance (cm)	Ground truth (cm)	Error (cm)
$\|b1 - b2\|$	$[0.1, 0.1, 13.3]^T$	$[0, 0, 13.1]^T$	$[0.1, 0.1, 0.2]^T$
$\|b1 - b4\|$	$[0.5, 9.8, 0.2]^T$	$[0, 9.8, 0]^T$	$[0.5, 0, 0.2]^T$
$\|b1 - b5\|$	$[2.7, 0, 0.1]^T$	$[3, 0, 0]^T$	$[0.3, 0, 0.1]^T$

We evaluated the fixed-range CbCr classifier on several other sample gesture frames taken with different skin color, lighting condition, and background. Table 3 shows the segmentation results. In our study, *room light* stands for the existing fluorescent illumination condition in our laboratory (i.e., 300–500 Lux measured using standard light meter (Reed LM-81LX), held vertically at the position of the hand). The *strong light* stands for the existing room light as stated before plus the additional standard LED desk lamp pointing toward the subject (i.e., 500–800 Lux measured using standard light meter (Reed LM-81LX), held vertically at the position of the hand). For the remaining of this study, fixed-range CbCr skin classifier is adopted for skin segmentation in our system for segmenting the hand in a cluttered and moving background. Similar to other segmentation methods, the fixed-range CbCr method can still result in presence of some background noise (blobs showed in the segmentation results in Table 3). For our study, an upper size constraint on the size of segmented blobs is introduced (i.e., 3000 pixels) where all the noise blobs smaller than the constraint will be eliminated.

3. Hand Tracking and Trajectory Reconstruction

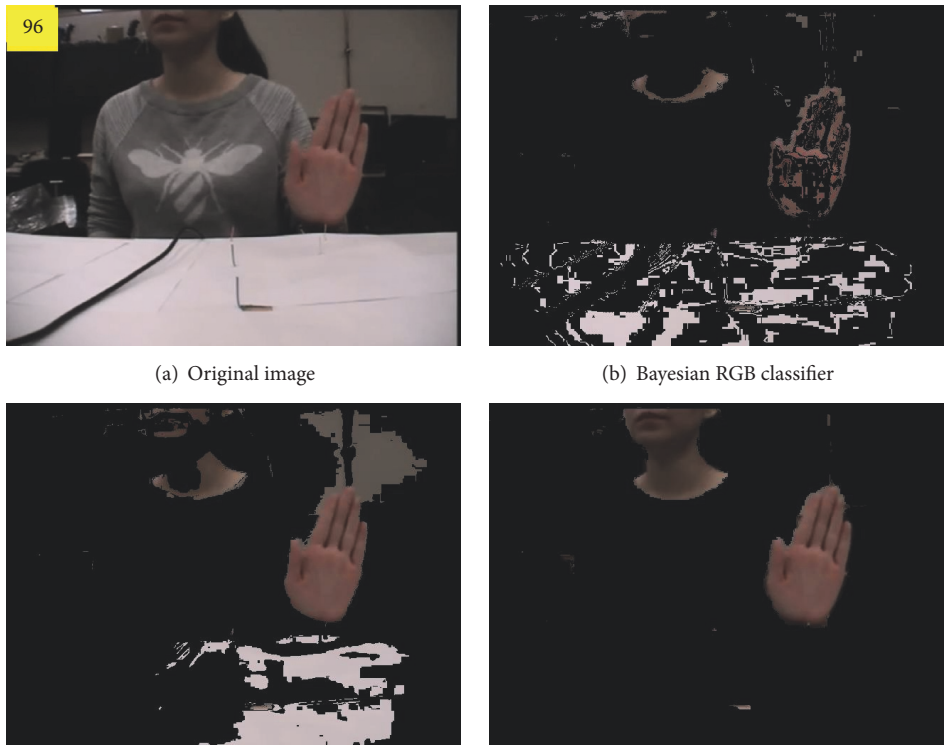
In the previous section, hand blobs are segmented, where in some nonideal cases other skin areas such as face and neck are also included as a part of the segmentation process. Due to the similarities of the hand area and other skin areas shared in various attributes, such as shape and color, a scheme involving position constraint is adopted to separate hand blobs from other skin areas. Kalman Filter (KF) is employed to track hand and reduce the influence of other skin areas on posture and trajectory recognition. Based on the tracking results, trajectories of the moving hands can then be reconstructed.

3.1. Application of Kalman Filter. Kalman Filter (KF) [51] has been extensively used in the computer vision community for object tracking [52]. Here, a simplified representation of the state of the hand x and time t is defined as the position and velocity of the centroid of the hand blob, or

$$x_t = \left[u_t, v_t, u_t', v_t'\right]^T, \tag{4}$$

where (u_t, v_t) represent the pixel values of the centroid of the hand blob, which is calculated by the yellow bounding box in Figure 6. The bounding box is generated by the utmost points in four directions, including up, down, left, and right, of the hand blob. The point is represented by a vector $[u, v]^T$, where u and v are the pixel values of the bounding box center which is represented by a red cross. u_t' and v_t' indicate the velocity of the hand blob in the direction of u and v, respectively, at frame t. The implementation of the filter consists of two steps, the prediction step (updates) and the correction step (measurements). For each video frame, the hand location is predicted from previous frames. The KF model is modified according to the measurements. Figure 7 displays the conceptual diagram of iterations in KF. Since KF is a recursive estimator, in the prediction step, the computation of the estimated state for the next time step only requires the estimated state and the measurement of the current time step.

For example, A can be defined as a 4×4 the state transition matrix models the transition relationship between the current state x_{t-1} and the next state x_t. The measurement matrix H is a 2×4 matrix which maps the state x_t into the observation vector z_t. The KF motion model in this system is simplified as a constant velocity model. The transition model A and measurement model H are remaining the same during the whole tracking process. In the prediction step, the prior estimated state x_t^- and the prior estimated error covariance P_t^- can be



(a) Original image (b) Bayesian RGB classifier

(c) Gaussian classifier (d) Fixed-range CbCr

Figure 5: Hand segmentation by different skin classifier.

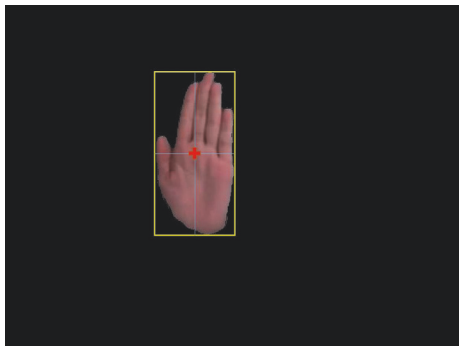


Figure 6: The hand location is represented by the pixel value of the centroid of the segmented hand blob.

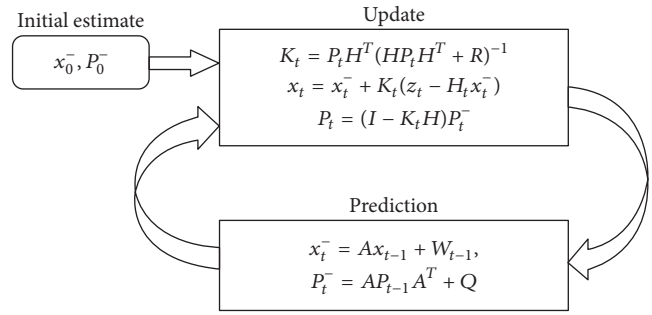


Figure 7: The iteration iterative cycle of KF.

calculated by the following equations, where W is the process noise in Gaussian distribution with the noise covariance Q.

$$x_t^- = A x_{t-1}^- + W_{t-1} \Longrightarrow$$

$$\begin{bmatrix} u_t^- \\ v_t^- \\ u_t'^- \\ v_t'^- \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{t-1} \\ v_{t-1} \\ u'_{t-1} \\ v'_{t-1} \end{bmatrix} + W_{t-1}, \quad (5)$$

$$P_t^- = A P_{t-1} A^T + Q,$$

$$p(W) \sim N(0, Q).$$

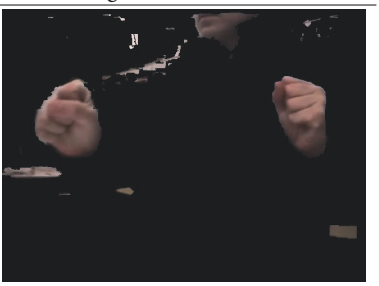
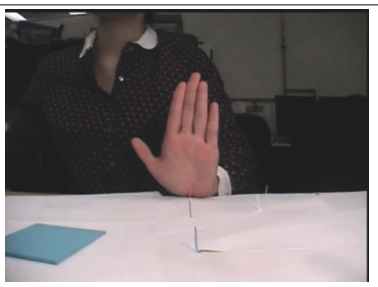
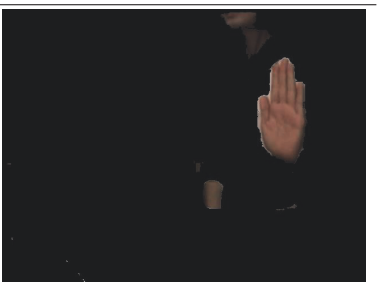
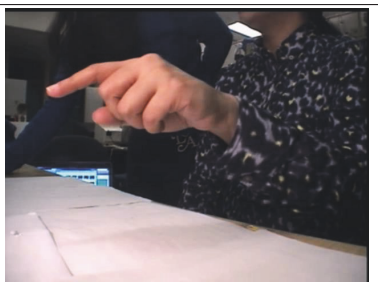
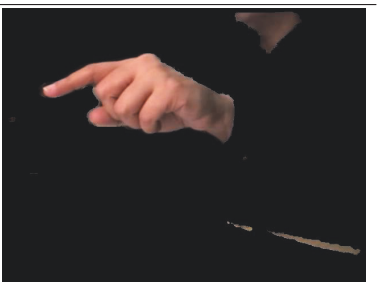
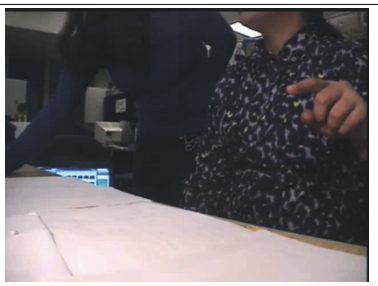
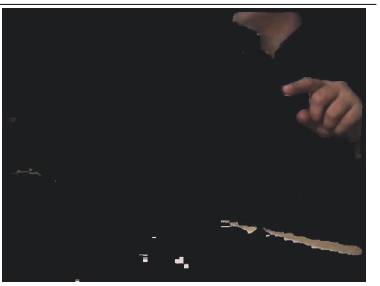
The actual hand location needs to be measured to correct the prior state x_t^- and error covariance P_t^-. In order to generate and improve posterior state x_t with a better accuracy, the measurement z_t can be calculated based on our constant velocity assumption as follows:

$$z_t = H x_t + V_t \Longrightarrow$$

$$\begin{bmatrix} u_{\text{meas}_t} \\ v_{\text{meas}_t} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_t \\ v_t \\ u'_t \\ v'_t \end{bmatrix} + V_t, \quad (6)$$

$$p(V_t) \sim N(0, R),$$

Table 3: The skin segmentation results of fixed-range CbCr classifier.

Image acquiring condition	Original image	Segmentation results
(i) Room light (ii) 300–500 Lux (iii) Lighter skin color		
(i) Strong light (ii) 500–800 Lux (iii) Lighter skin color		
(i) Room light (ii) 300–500 Lux (iii) Lighter skin color		
(i) Strong light (ii) 500–800 Lux (iii) Darker skin color		
(i) Room light (ii) 300–500 Lux (iii) Darker skin color		

where V_t is the measurement noise in Gaussian distribution with noise covariance R. The posterior state x_t and error covariance P_t can be obtained from the correction equations.

$$K_t = P_t H^T \left(HP_t H^T + R_t\right)^{-1},$$
$$x_t = x_t^- + K_t \left(z_t - H_t x_t^-\right),$$
$$P_t = \left(I - K_t H\right) P_t^-, \tag{7}$$

where K is the gain matrix. In addition and in order to constrain the tracking state defined within each frame, a saturation value is defined in-between frames. This added

FIGURE 8: The system setup of two cameras. The example frames taken by camera C_1 and C_2 are listed on the left of this image.

metric which is defined as a function of the area of the blob can further increase the robustness of the tracking results.

3.2. Experimental Study. In this section, experimental results associated with one-hand tracking for different speed and conditions are presented (results associated with tracking two hands can be found in [53]). Figure 8 shows the setup of the two cameras and example frames taken from cameras C_1 and C_2 are also included. Frames taken from camera C_1 and camera C_2 are also referred to as front view and side view, respectively, for the rest of the paper.

Figure 9 shows the results of multicamera tracking of a single hand in the presence of minimum background noise. For the case that involves background noises, Figure 10 shows an example of the tracking result. At the initialization, hand blob is successfully located and tracked when the hand blob is not overlapping with any background noise area (Figure 10(a)). In the next frame (Figure 10(e)), the hand is now overlapped with a noise color blob area. Instead of the previous frame's bounding box being expanded to include the whole color blob, the predicted tracking results of KF are adapted in order to adjust the bounding box at the new hand location. In the next frame, the overlapping is over and the bounding box is updated with the size and location of the hand blob. The KF tracking performed well in such case, but if the hand area is overlapped with the background noise for a longer period, the state of the bounding box associated with the tracking result will drift away based on the predicted hand speed. Figure 11 shows an example. The hand blob is tracked in Figures 11(a), 11(b), and 11(c). When the hand is overlapping with the background noise area for a long time, the system would lose track of the hand area.

4. Trajectory Reconstruction and Smoothing

In Section 2, it was shown that the 3D coordinates of a point which is inside of the overlapping area of the camera views can be reconstructed by the pixel values on projected image planes. For each instance of time t, and the definition of the tracking state, the hand center $[u_1^t, v_1^t]$ and $[u_2^t, v_2^t]$ can be extracted from both camera views. Substitute $[u_1^t, v_1^t]$ and $[u_2^t, v_2^t]$ for q_1, q_2 in (1) and (2); the hand coordinates at time t and the corresponding points Q^t in the world space can be reconstructed. Figure 12 shows example of reconstructed hand trajectory of single-hand and two-hand movements.

4.1. Trajectory Smoothing. The reconstructed trajectory shown in Figure 12 in general is not a smooth trajectory. For example, in Figure 13(a), trajectory starts from the green marker point, follows the blue arrows, and then ends at the red marker point. The fluctuation along the trajectory can be due to the variation in the size of the bounding box. For instance, if the hand area is occluded with other skin-like areas, the center of the segmented color blob does not represent the true position of hand center. To eliminate effects of such fluctuation, each trajectory is represented by 3 spatiotemporal components along x, y, and z directions. Figure 13(b) shows projected information of circle trajectory over time in x, y, and z directions. Two smoothing methods are then implemented and compared on the trajectory, namely, locally weighted scatterplot smoothing (LOESS) [54] and the robust version of LOESS (RLOESS). LOESS is locally weighted scatterplot smoothing using least squares quadratic polynomial fitting. RLOESS is a robust version of LOESS smoothing that assigns lower weight to outliers in the regression.

Figure 14 compared the result of these two methods. In this figure, span is a percentage of the total number of data points, less than or equal to 1. For example, if the span equals 0.1, it implies that 10% of data points are included for each smoothness computation. The red dots in Figure 14 represent the original data points along x direction of 3D reconstruction circle trajectory. The blue curve represents the smoothed data points. Based on observation, if span value is set too big (Figures 14(b) and 14(d)), the smoothed curve is not fitting well with the original data points. If the span value is set too small, the effect of outliers in the original data cannot be completely eliminated (Figure 14(a)). In conclusion, RLOESS is superior to LOESS for this case. In our system, we adopt RLOESS with span of 0.1 to eliminate small fluctuation on reconstructed hand trajectories.

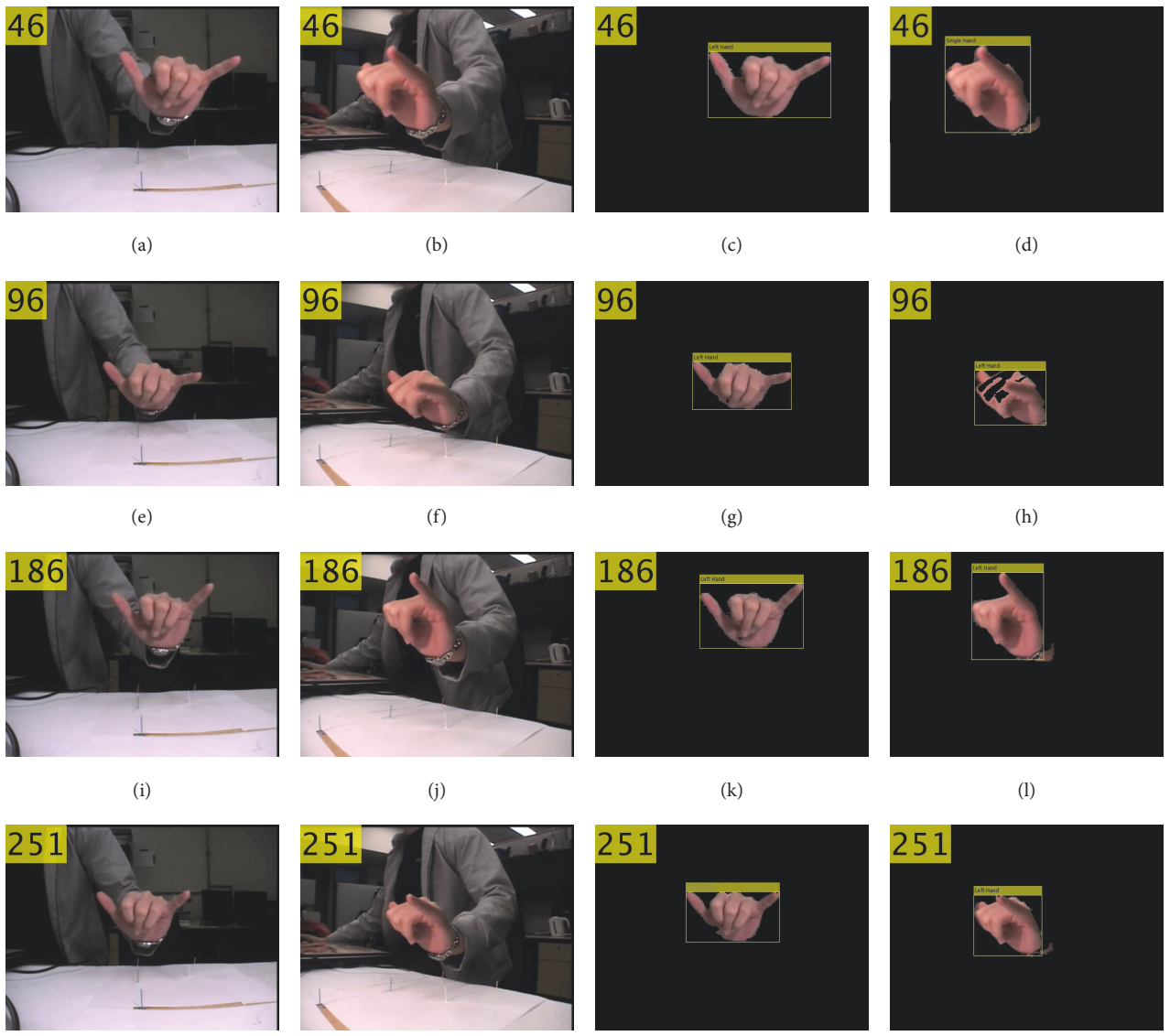


FIGURE 9: Sample frames from gesture video with a hand moving and the KF tracking result. The fist column ((a), (e), (i), and (m)) and the second column ((b), (f), (j), and (n)) display the original frame clipped from camera C_1 and camera C_2, respectively. The tracking result for both cameras views are showed in the third column ((c), (g), (k), and (o)) and fourth column ((d), (h), (l), and (p)).

Figure 15 shows the smoothed trajectory by RLOESS, where the fluctuation has been erased.

5. Hand Trajectory Recognition

In the previous section, methods for hand tracking, trajectory reconstruction, and smoothing were presented. Recognition of hand trajectory is a challenging task due to various patterns that hands can make in space and time. For instance, the same intended motion trajectory performed by different people usually has the same spatial pattern. Dynamic Motion Primitive (DMP) [44, 55] is a method for trajectory control which is shown to have good performances in [6] for handwriting recognition and in [11] for 2D hand trajectory recognition. In this paper, we extended the method for spatial hand trajectory recognition.

5.1. An Overview of Dynamic Movements Primitives. Dynamic Movements Primitives (DMP) method models movements with the given start and end states into a set of differential equations. For example, it is capable of encoding the spatiotemporal information of trajectories of the hand movements into a weight vector that is robust with respect to the spatiotemporal variations along the same hand trajectory.

The differential equations that characterize the spatiotemporal evolution of a dynamic system with the given start and end states are given in (8) and (9). A second-order linear damped spring model with a nonlinear function f is added

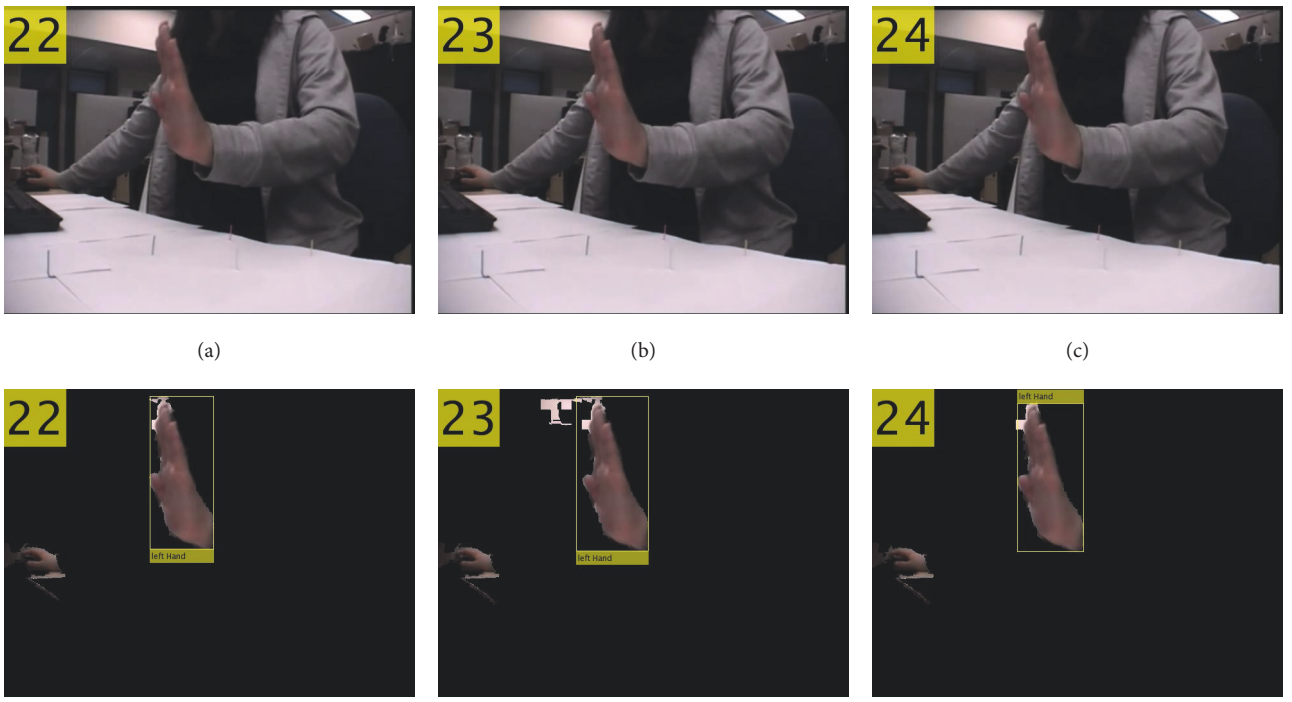


Figure 10: Sample frames clipped from gesture video with hand moving and overlapping with background noise. The first row ((a), (b), and (c)) shows the original frame captured by camera C_2. The second row ((d), (e), and (f)) shows the tracking result.

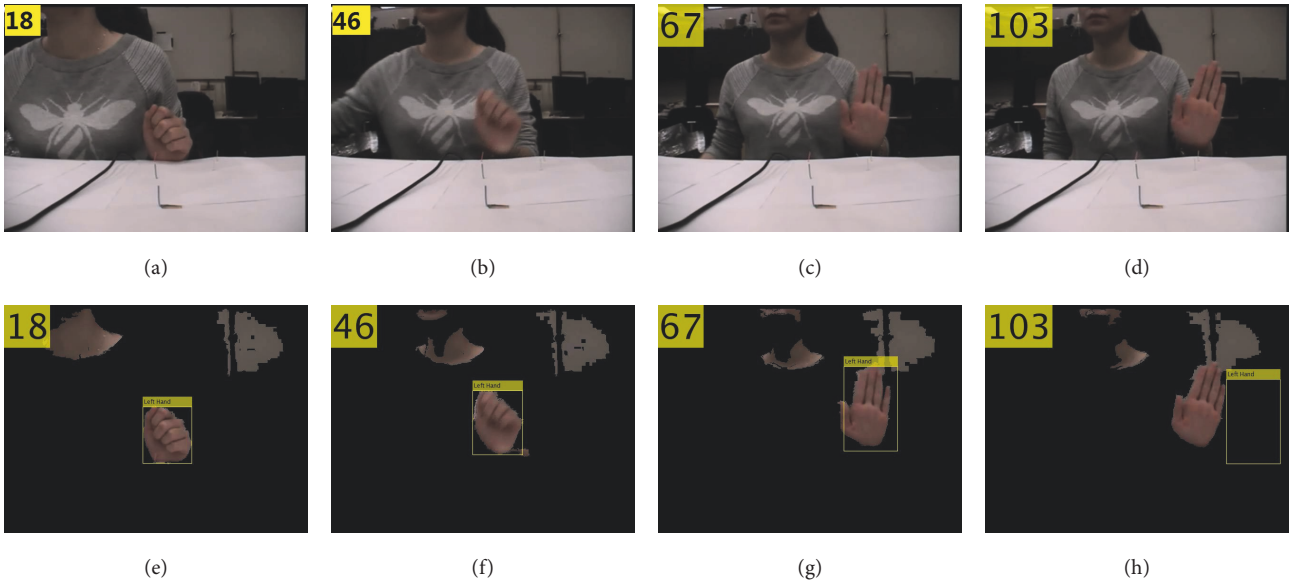


Figure 11: Sample frames clipped from gesture video with a hand blob overlapping with a background blob. The first row ((a), (b), (c), and (d)) shows the original frame captured by camera C_1. The second row ((e), (f), (g), and (h)) shows the tracking result.

in (8) as the forcing term. This nonlinear force function can capture the complexities of motion patterns made by human.

$$\tau \dot{z} = \alpha_z \left(\beta_z \left(g - y\right) - z\right) + f, \tag{8}$$

$$\tau \dot{y} = z. \tag{9}$$

In (8) and (9), y, z, and $\dot{z}$ represent the position, velocity, and acceleration of the hand motion dynamics. τ is a time constant which represents the trajectory duration and g is a known goal representing the final hand position of the trajectory. For a suitable selection of parameters α_z and β_z, the forcing term f would decay to zero over time, which

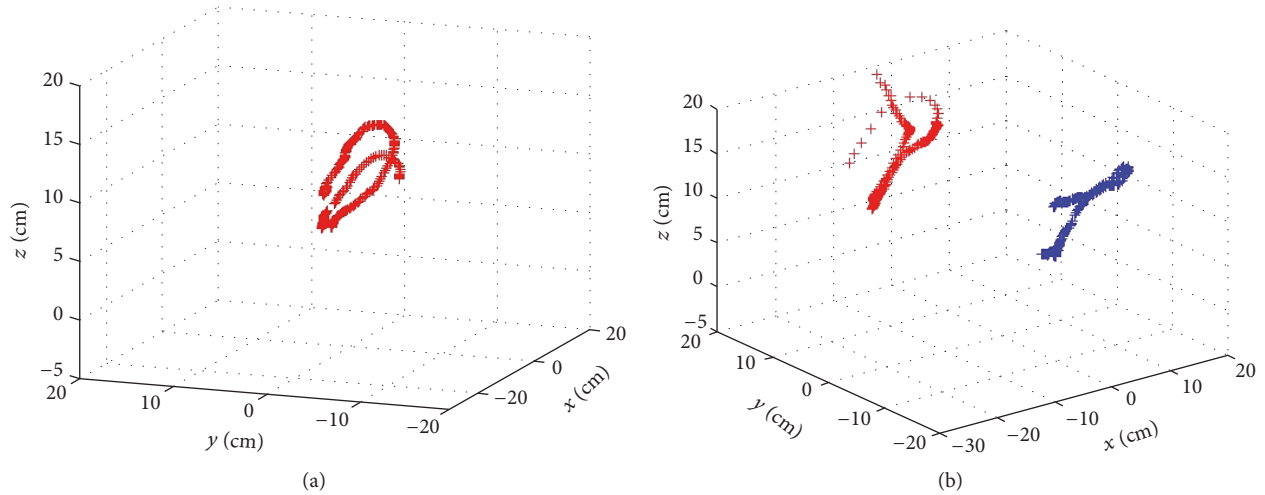


FIGURE 12: Examples of reconstructed 3D trajectory of a single-hand (a) and the multiple hand movements (b).

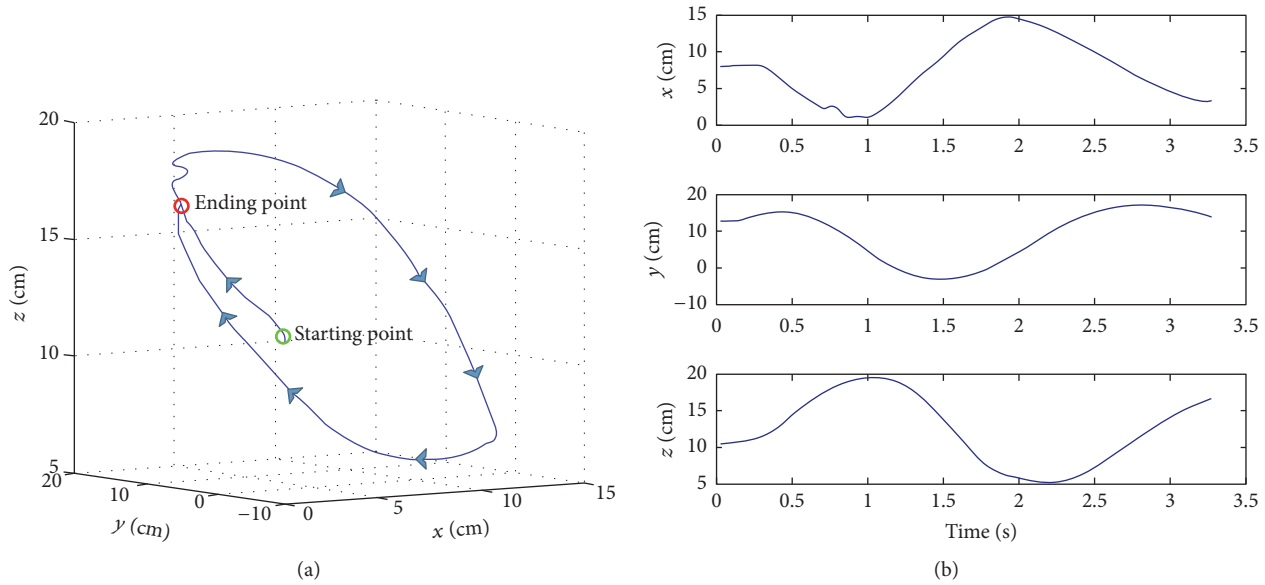


FIGURE 13: (a) Reconstructed 3D trajectory when hand moving in a circle; (b) represented in x, y, and z directions.

allows the system to converge to the goal position $(y, z) = (g, 0)$.

The nonlinear forcing function f is composed of a set of Gaussian-like basis functions as

$$f(x) = \frac{\sum_{i=1}^{N} \phi_i(x) w_i}{\sum_{i=1}^{N} \phi_i(x)} x(g - y_0), \tag{10}$$

where w_i is weights of basis functions and $\phi_i(x)$ are N basis Gaussian-like functions. The forcing term f would be vanished a long time.

The weight vector $\mathbf{w} = [w_1, \ldots, w_N]^T$ in f preserves the shape information of the trajectories. For instance, if $\mathbf{w}$ is fixed and other parameters such as the goal state g or time constant τ change, the DMP will generate topological similar trajectories. In other words, similar trajectories would have similar feature vector $\mathbf{w}$ which is called the invariance properties of the DMP model [55]. With such property, trajectories can be classified based on the weight vectors.

5.2. Weight Vector Extraction from 3D Trajectory. Given a trajectory in one dimension $Q = (q_0, \ldots, q_{t-1})$, the dynamics at each point $q_s = (y, z, \dot{z})^T$ denotes the state vector at time s and can be calculated based on frame rate, where t is the time duration of the whole path. To learn the weight vector for a given trajectory, the initial and goal states and also the time duration can be extracted from the trajectory. From (8), f can be rewritten as

$$\begin{aligned} f &= \alpha_z(\beta_z(g - y) - z) - \tau\dot{z} \\ &= \frac{\sum_{i=1}^{N} \phi_i(x) w_i}{\sum_{i=1}^{N} \phi_i(x)} x(g - y_0). \end{aligned} \tag{11}$$

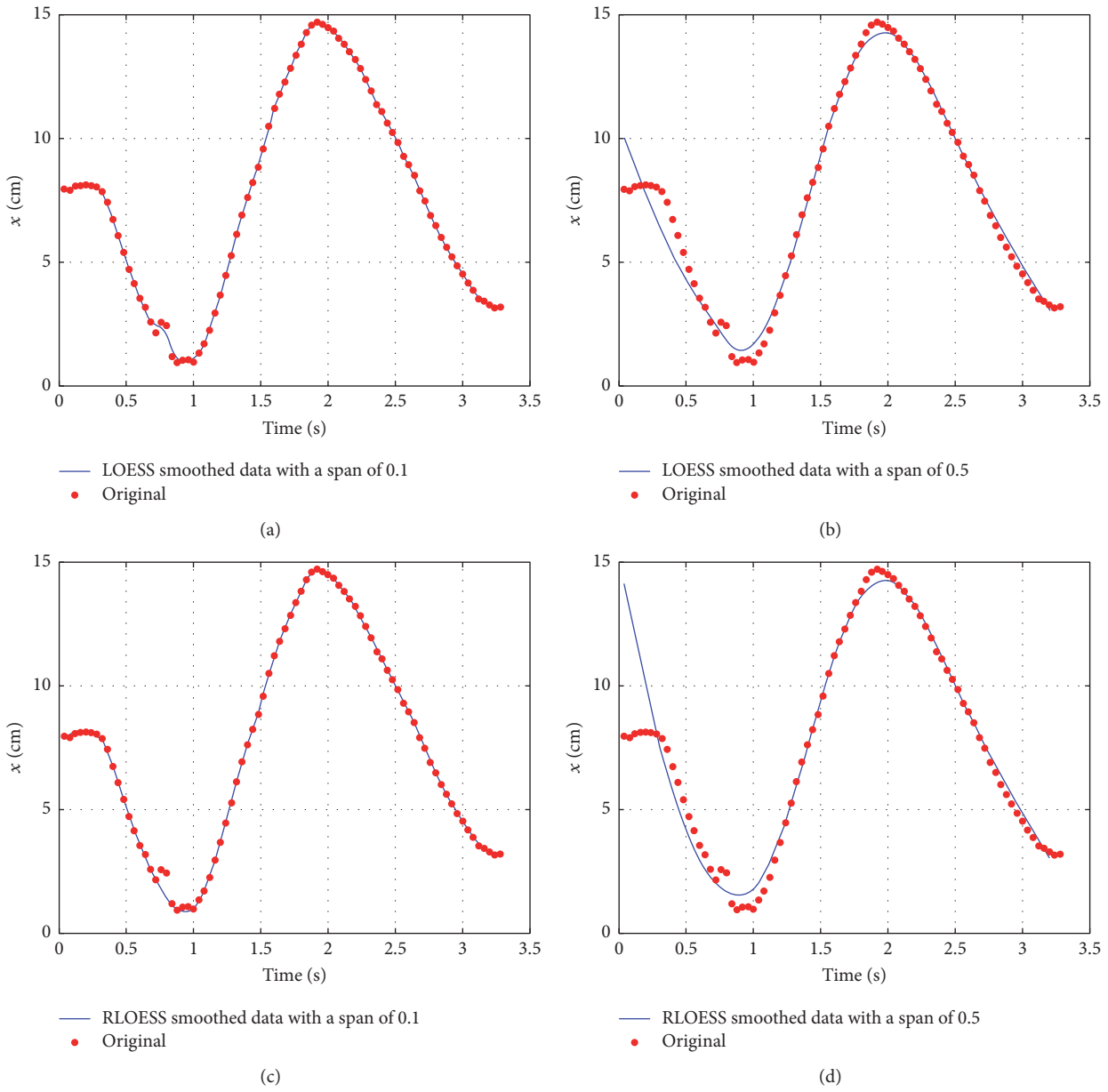


Figure 14: Comparison of original data points (red dots) and smoothed data points (blue curve) using LOESS ((a) and (b)) and RLOESS ((c) and (d)) with different span. (a) span = 0.1; (b) span = 0.5; (c) span = 0.1; (d) span = 0.5.

The weight vector $\mathbf{w}$ can be learned using the locally weighted regression (LWR) as stated in [44].

Before conducting the trajectory recognition, a trajectory instance "Circle" (Figure 17(d)) is selected from the collected trajectories in order to visualize the weight vector and learned DMP models. The acquired hand trajectory is in 3D space, where DMP can be utilized along each projected direction of the world coordinate system. Figure 16 shows the learned DMP models and weight vectors with different dimensions (D_w) of the trajectory projected in x direction. Figures 16(a), 16(c), 16(e), and 16(g) display the original trajectories in green and the learned trajectories by DMP in blue. Figures 16(b), 16(d), 16(f), and 16(h) show the corresponding learned weight vectors. It can be seen that when the dimension N of the weight vector increases, the learned trajectory approaches the original trajectory.

5.3. Training Stage for Hand Trajectory Classifier Using Support Vector Machine. The invariance properties of DMP preserve the shape information of trajectory in weight vectors and can be used for trajectory recognition. In [11], they compare two classification methods: k-nearest-neighbor (k-NN) and support vector machine (SVM). Based on their experiment result, SVM obtains a much more better accuracy than k-NN. In our implementation, multiclass SVM training and testing are performed using the LIBSVM library [56].

Table 4: Recognition accuracy with different number of D_w.

Training trajectory number	Weight vector dimension D_w			
	5	10	20	40
150	86.67%	74.12%	59.17%	51.67%

Table 5: Recognition result of testing trajectories.

Class	Jump	Left	Right	Circle	Push
Jump	12	0	0	0	3
Left	0	14	0	0	0
Right	0	0	17	0	0
Circle	0	0	0	18	0
Push	6	4	1	0	15
Accuracy	84.4% (76/90)				

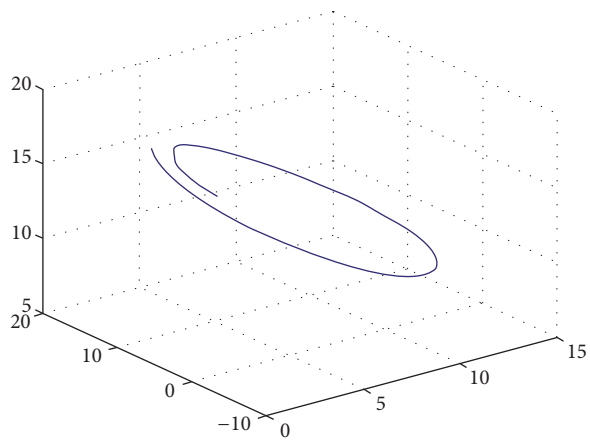


Figure 15: Smoothed trajectory by RLOESS.

For trajectory recognition, five classes of trajectories are collected. These five classes are consisted by "Jump," "Left," "Right," "Circle," and "Forward." Figure 17 shows the example trajectories for each class. Eight people (two male and six female) are asked to perform the trajectories, 5 for each class. Over 200 trajectories are collected, and 3/4 of the data set is taken to train the SVM, while the rest of the data set are held out for testing. The trajectories in training and testing data set are performed by different people.

The SVM with a linear kernel is trained for trajectory recognition. Table 4 gives the 5-fold cross-validation recognition accuracy based different weight vector dimensions. As the dimension increases, for the same number of training data, the accuracy decreases. This is because the bigger the weight vector dimension is, the more parameters in the SVM need to be decided and more training data is needed. The highest recognition rate is obtained at $D_w = 5$. This is also because the classes of trajectories we collected are relatively simple and well distinguished with each other. For complex trajectories, a weight vector with higher dimensions is needed.

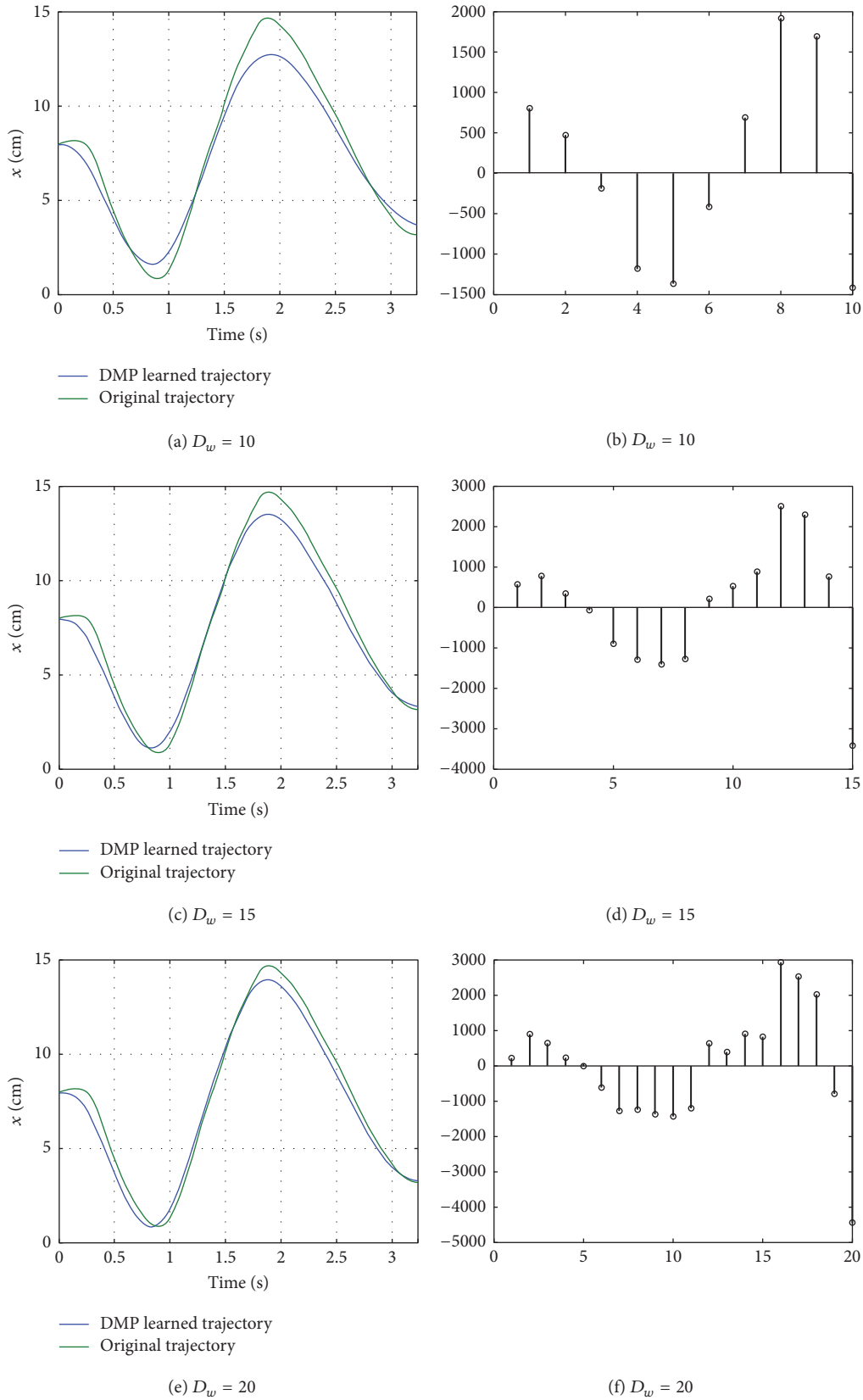
5.4. Testing Stage for Hand Trajectory Recognition. We apply the trained SVM on the testing data set which resulted in accuracy of 88.0%. The recognition results are shown in Table 5. The misclassification appears due to the user habit. Because the trajectories "Jump" and "Push" are both moving forward, for some cases, if the trajectory "Jump" is made with smaller radian, it would be misclassified as "Push." Also the trajectory "left," sometimes, is performed with an angle, and the SVM will recognize it as "Push" as well. But the trajectory "Circle," which is quite distinctive from other classes, can be classified perfectly.

We tested the trained SVM on 90 testing trajectories collected among 3 people, 6 for each class. The recognition result is shown in Table 5. The misclassification appears due to the user habit. As you can see, because the trajectories "Jump" and "Push" are both moving forward, for some cases, if the trajectory "Jump" is made with smaller radian, it would be misclassified as "Push." Also for the trajectory left, sometimes it was presented with an angle, and the SVM would take it as push as well. But for the trajectory "Circle," which is quite distinctive from other classes, it can be classified perfectly.

6. Hand Posture Recognition

Hand posture recognition is another key part of the gesture recognition. Our system samples hand postures and follows the posture changing along hand trajectory (Figure 18). There are three steps for posture recognition, namely, (a) hand image acquisition, (b) feature extraction, and (c) classification. For our system, hand image acquisition is done along the trajectory tracking process where the hand postures are already segmented from the background.

Template matching is a simple method of posture recognition, and it is easy to add or remove template classes. To extract features on hand posture, a convex hull on silhouette [31] or fingertip detection using circular mask as a correlation techniques [45] can be employed. However, these recognition approaches based on hand silhouette or contour usually require a clean background where the hand can be well segmented. In our case, hand posture is made in a clutter and moving background and segmented using fixed-range CbCr skin classier. Sometimes, the hand cannot be well segmented from the background. In such case, hand can be taken as partially occluded. Therefore, a feature detector that



(a) $D_w = 10$

(b) $D_w = 10$

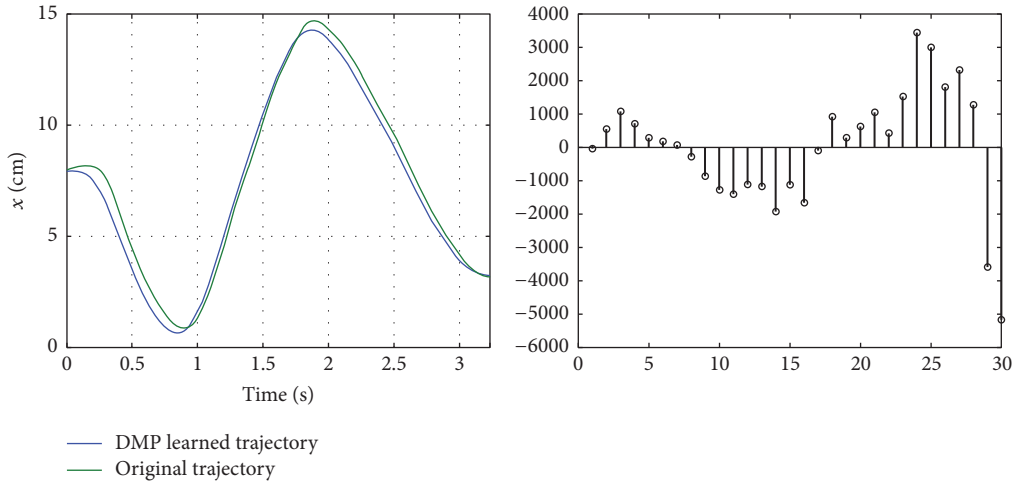
(c) $D_w = 15$

(d) $D_w = 15$

(e) $D_w = 20$

(f) $D_w = 20$

Figure 16: Continued.



(g) $D_w = 30$ (h) $D_w = 30$

FIGURE 16: (a), (c), (e), and (g) are the comparisons between DMP learned trajectories (in blue) and the original trajectories (in green) along with different weight vectors dimension D_w. (b), (d), (f), and (h) show the learned weight vectors.

is insensitive to illumination variation and partially occlusion is needed. SIFT is such a feature detector and descriptor which is also robust to scale and orientation changes. In addition, it is robust to affine distortion in some range, which could benefit posture recognition, since the relative position kept changing between hand and cameras and causes affine distortion between the input hand postures and posture templates. In our work, SIFT is used for feature detector and posture recognition. Method of *Bag of visual words* and SVM are also combined for classification.

6.1. Scale-Invariant Feature Transform. Scale-Invariant Feature Transform (SIFT) is a feature detector developed in [22]. SIFT features are shown to provide robust matching in a range of occlusion and affine distortions, addition of noise, and change in illumination. To acquire features at different scales, SIFT use Gaussian filters to change the variance to convolve with the original image and also the downsampled images. The difference of Gaussian is calculated by subtracting the adjacent images convolved with Gaussian in the same octave (Figure 19).

In order to detect the local maxima or minima of difference of Gaussian (DoG), each sample point is compared to its 26 neighbors in a 3×3 region, 8 in its own image, and 9 in the scale above and below (Figure 20). A point is selected as a feature point only if it is larger or smaller than all of the other 26 neighbors. For the stability of feature points, once a feature point is detected by the method above, a threshold on minimum contrast is performed on these feature points between its neighbors. Also eliminating edge response is applied on the DoG since it has a strong response to edges. In this way, the feature points with strong contrast and within the image remain for feature points matching.

By assigning a constant orientation to each feature point based on local image properties, the feature point is made invariant to rotation. This orientation information is also used to build a feature point descriptor that is a vector containing 128 nonnegative elements. The resulting vector is defined as SIFT keys and is used in a nearest-neighbors approach to find the matching points and detect the same object between images. Figure 21 shows the feature points detected by SIFT in green circles on two-hand palms. The size of the circles presents the scale of feature points, and the radius of each circle represents the direction of each feature point. 36 sets of matching points are found between these two posture examples and connected in blue lines.

Our work is designed to recognize six targeted hand postures, namely, "Palm," "V," "Point," "Y," "Fist," and "Eight" (Figure 22). In order to demonstrate that features detected on these postures have good discrimination between different classes, we apply SIFT to each class of postures shown in Figure 23 and listed the number of matching points in Table 6. The numbers marked in bold font on diagonal are the number of the matching points in the same class. As you can see, it is bigger than other numbers on the same row or column, which present the number of matching points between two different classes. It shows that although there are matching features between different posture classes, the number of matches in the same type posture is larger for any misclassifications among other postures.

6.2. Bag of Visual Words. Bag of visual words is a popular algorithm for image classification. Each image is represented by a set of detected feature points, and bag of visual words is using a vector to represent the occurrence counts of feature points. In other words, it is a histogram over the feature points. In this way, each image is represented by a histogram vector. Figure 24 shows a typical processing pipeline for generating a feature point histogram for each image.

In this paper, the feature points are detected by SIFT and the clustering is accomplished by K-means. K-means clustering encodes each feature point by the index of which cluster it belongs to. Usually, this is done by finding the

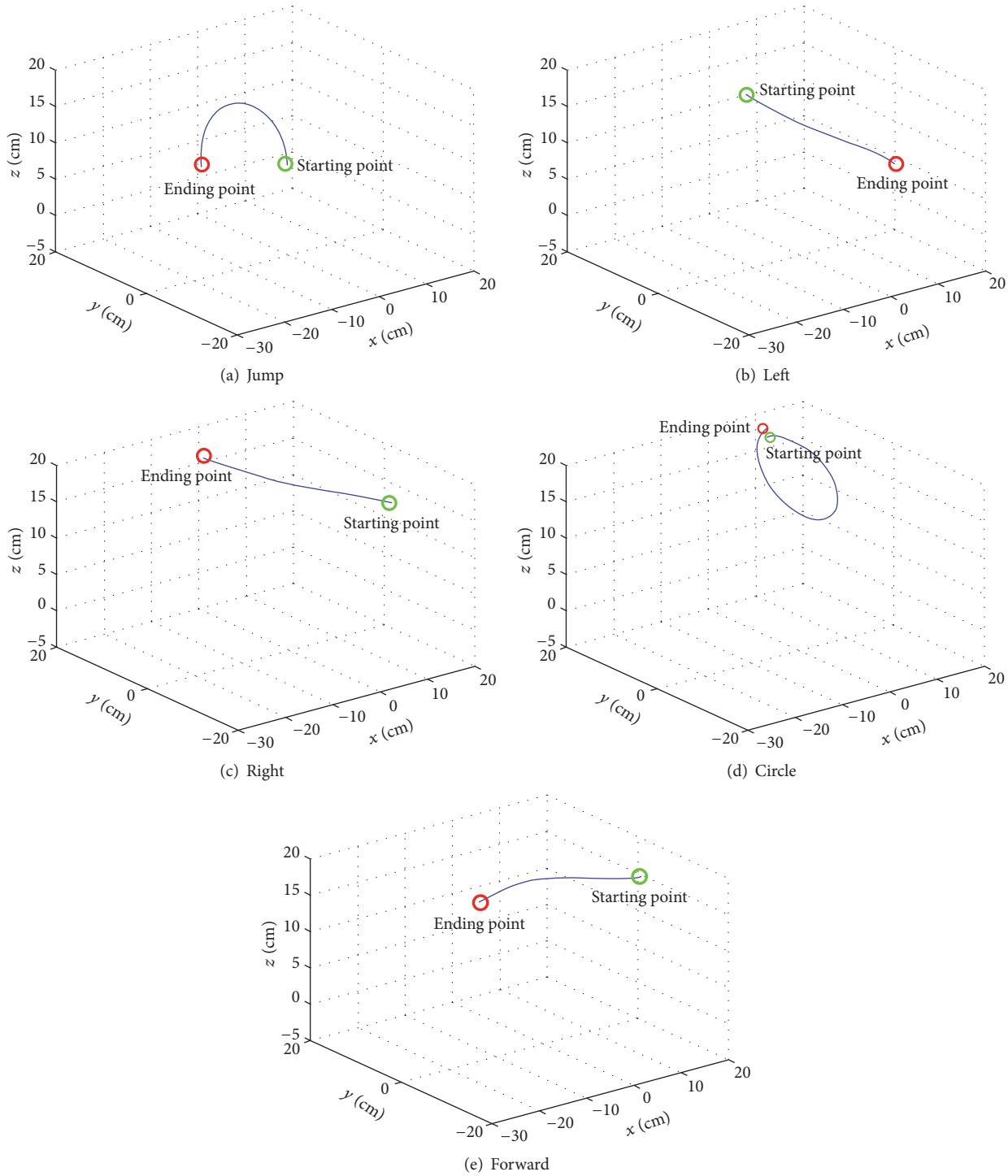


FIGURE 17: Trajectory samples of five classes. The starting point is marked in green, and the ending point is in red.

shortest Euclidean distance between the input feature point and the cluster centers which are trained by a group of feature points extracted from a set of training images. How to determine the number of clusters is a problem. If the number of clusters is too small, there have not been enough discrimination between classes, and the classification accuracy will be decreasing. If the number of clusters is too big, then the features would be over-scattered, and the classification accuracy is still going to be decreasing. In [57], an approach is proposed to determine the number of clustering k.

$$k \approx \sqrt{\frac{n}{2}}, \tag{12}$$

Table 6: The number of matching points between different posture classes.

Posture	Palm	V	Point	Y	Fist	Eight
Palm	**45**	12	6	16	8	10
V	7	**36**	13	8	9	12
Point	7	22	**39**	9	13	11
Y	12	14	7	**25**	11	9
Fist	4	8	15	6	**20**	6
Eight	16	15	8	15	9	**27**

Table 7: Training accuracy with different number of training data.

Number of training data	120	240	340	402	480
Accuracy	98.3%	98.8%	98.5%	98.8%	98.8

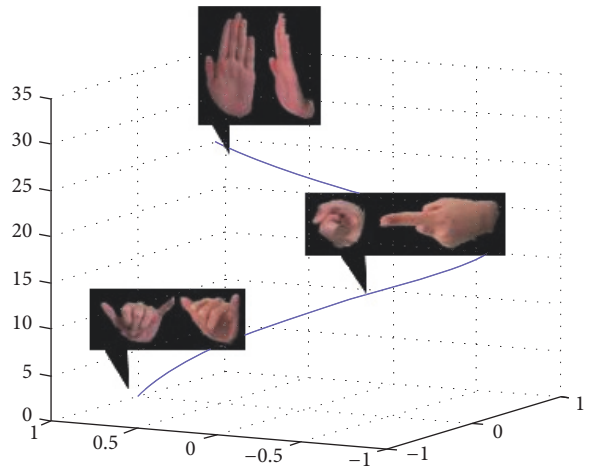


Figure 18: Hand posture is changing along the trajectory, and the hand posture is captured by both cameras.

where n is the number of detected SIFT feature points on the training images. There are over 20,000 feature points detected on our training images and we have selected k to be 150. After clustering, each template is represented by a vector of indexes that shows which cluster the corresponding feature is belonging to. The final step would be to sort this vector into a histogram which has k bins. In this way, each template would be mapped into a 1×150 vector which is called bag-of-words vector.

6.3. Training Stage for Posture Classifier Using Scale-Invariant Feature Transform. After mapping the feature points of each template into one bag-of-words vector, those vectors are employed to train a multiclass SVM classifier model. The SVM is a supervised learning method for classification and regression by creating an n-dimensional hyperplane that optimally divides the data into difference groups.

For posture recognition, six classes of postures are collected; both front view and side view for each class are included. Figure 25 shows the six classes of postures; the front view is shown in Figure 25(a), while the side view is shown in Figure 25(b); from left to right, the postures are "Palm," "V," "Point," "Y," "Fist," and "Eight."

For training a posture classifier, well-segmented hand posture templates are collected from six posture classes clipped from gesture videos; both front views and side views are included. By applying the bag of words, each image is represented by a 1×150 vector. The multiclass SVM is trained based on such vectors. Table 7 shows the 5-fold cross-validation accuracy of different numbers of training data with a linear kernel SVM. By increasing the number of training data, the accuracy would have slightly increased. Here we use 402 posture images as the training data set in order to build SVM classifier.

6.4. Testing Stage for Posture Recognition. The testing set contains 432 postures made by the other four people at 216 time state, 36 time states for each class. At each time state, one front view and side view of the posture are obtained. To test the performance of the trained posture classier, the images of each posture taken from both camera views are recognized individually. The recognition result is shown in Table 8. Each column refers to a posture instance that is classified into the corresponding class. An accuracy of 78.7% is obtained.

The classifier distinguishes most of the testing postures correctly. But for the postures belonging to class "Fist," a fair amount of postures is classified into class "Point," and also a few numbers of postures of "Y" are misclassified into class "Fist." The reason for this is that these three postures all contain a similar hand part of three fingers curling together. Therefore, they share more similar features. But for these postures which are well distinguished with each other, their recognition accuracies are higher.

Also, if we look into the misclassified postures, not well-segmented hand postures can cause the misclassification as well. Figure 26 shows three samples of misclassified postures. Background noises which are segmented as hand area (Figures 26(a) and 26(b)) are a cause of misclassification. Also due to hand appearance changing with viewpoint, some postures cannot be recognized when the appearance changes too much. For instance, in Figure 26(c), the index finger of posture "Eight" is total invisible from the camera view, and it was misclassified into class "Fist."

Considering the fact that the postures taken at the same time state are the same posture but different view, the posture

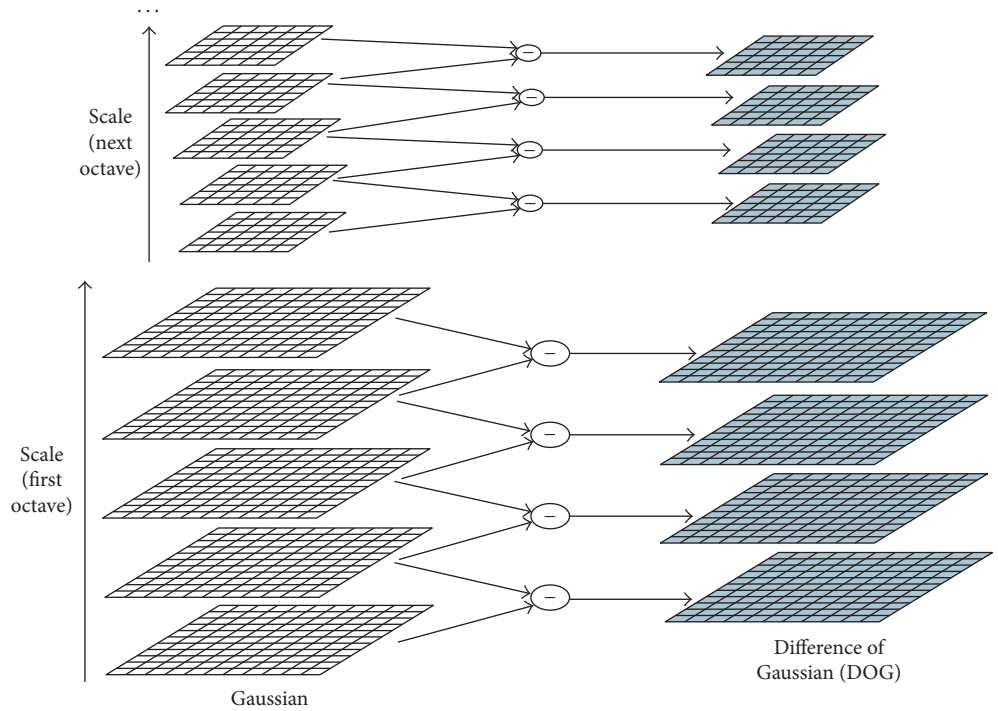


Figure 19: For each level of the octave, the original image is convolved with Gaussian in different variance. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images. After each octave, the Gaussian image is downsampled by a factor of 2, and the process is repeated [22].

Table 8: Confusion matrix for posture recognition on the testing set.

Posture	Eight	Fist	Palm	Point	Six	V
Eight	71	0	1	4	2	0
Fist	0	46	0	15	9	0
Palm	0	0	61	0	0	2
Point	0	17	3	52	7	2
Y	1	9	0	1	43	1
V	0	0	7	0	11	67
Accuracy	98.6%	63.9%	84.7%	72.2%	59.7%	93.06%
	78.7% (340/432)					

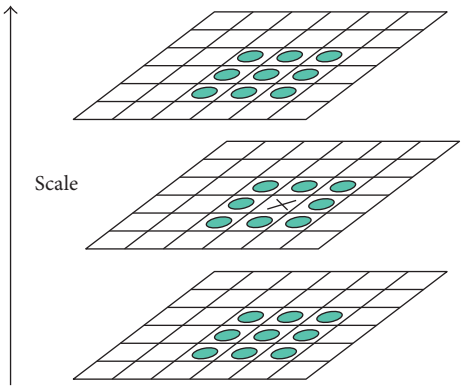


Figure 20: Extrema points are detected by comparing a pixel (marked with ×) to its 26 neighbors (marked with circles) in 3 × 3 regions at the current and adjacent DoGs [22].

recognition results from both camera views are associated with each other in the following way. At the same time state, if the recognition results of the postures taken from two cameras are different, then this posture is taken as ambiguous and discarded. The postures that are recognized as the same class from both camera views will be kept for gesture recognition in later steps. For each posture class, the recognition and abandon rate are shown in Table 9. Although nearly thirty percent of the testing data is abandoned, this scheme increases the recognition accuracy from 78.7% to 96.6%.

7. Gesture Recognition

For gesture recognition, the results of posture and trajectory recognition are combined together into a feature vector we

Table 9: Recognition result and abandon rate for the testing set.

Posture	Recognition accuracy	Abandon rate
Eight	35/35	1/36
Fist	16/20	13/36
Palm	27/27	9/36
Point	17/18	18/36
Y	17/17	19/36
V	31/31	5/36
Accuracy	96.6%	30.1%

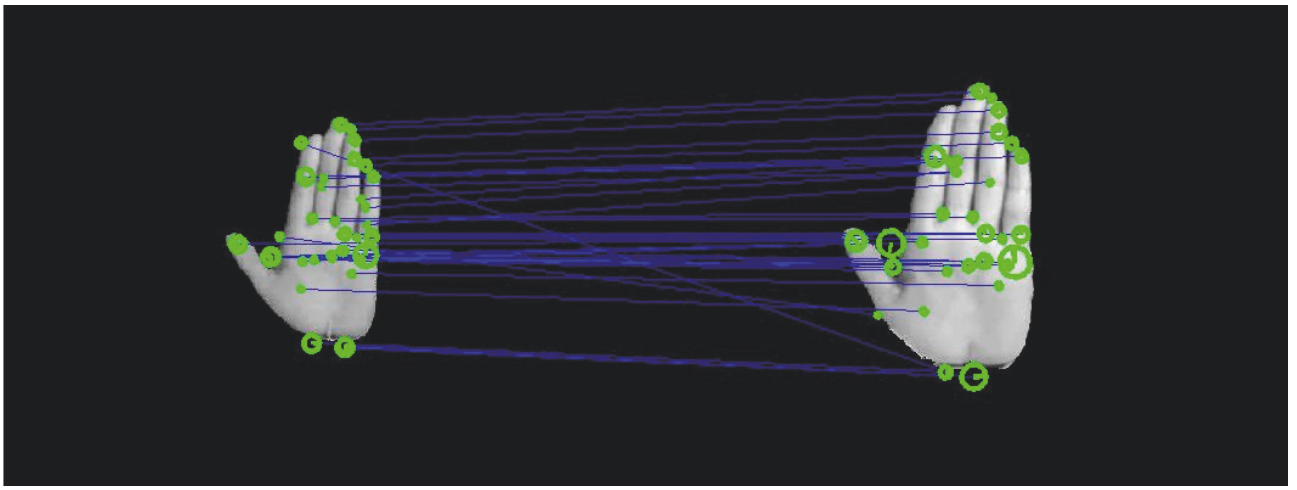


Figure 21: The matching feature points detected by SIFT on two palms.

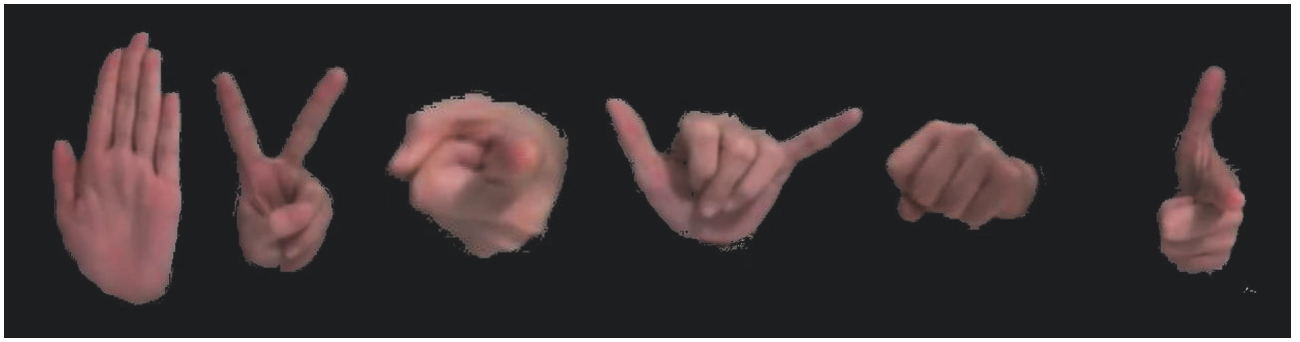


Figure 22: Six classes of posture, from left to right of the postures, are "Palm," "V," "Point," "Y," "Fist," and "Eight."

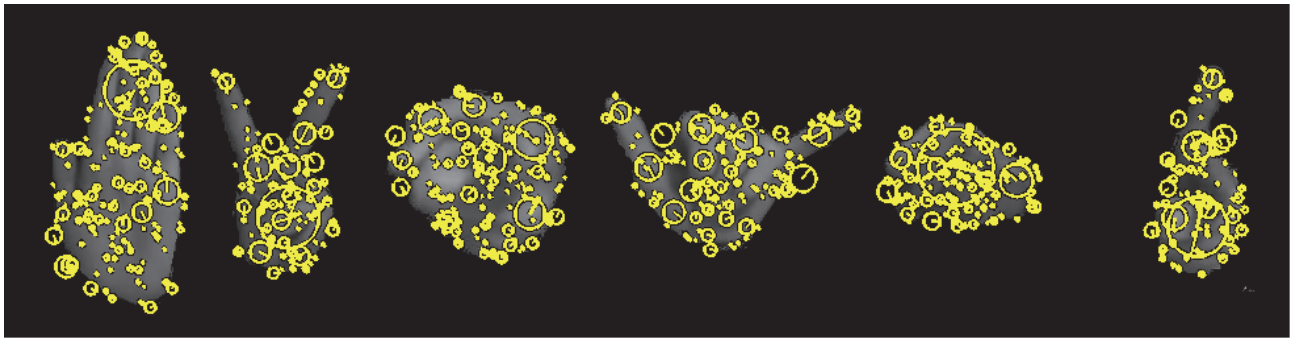


Figure 23: Feature points detected on six postures by SIFT, from left to right of the postures, are "Palm," "V," "Point," "Y," "Fist," and "Eight."

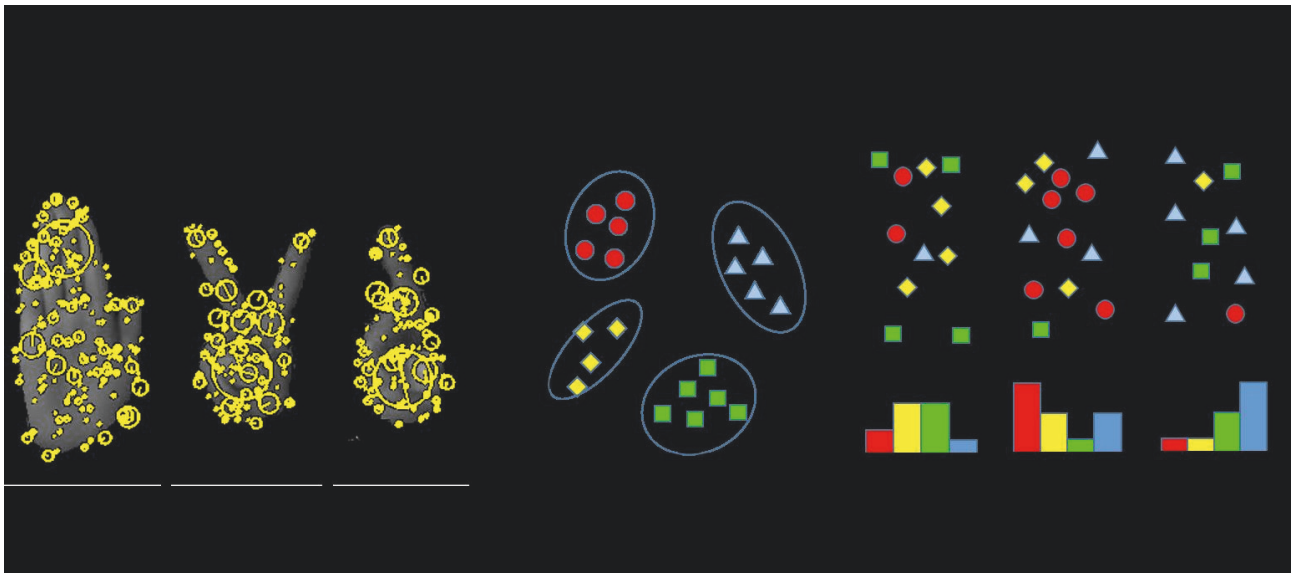
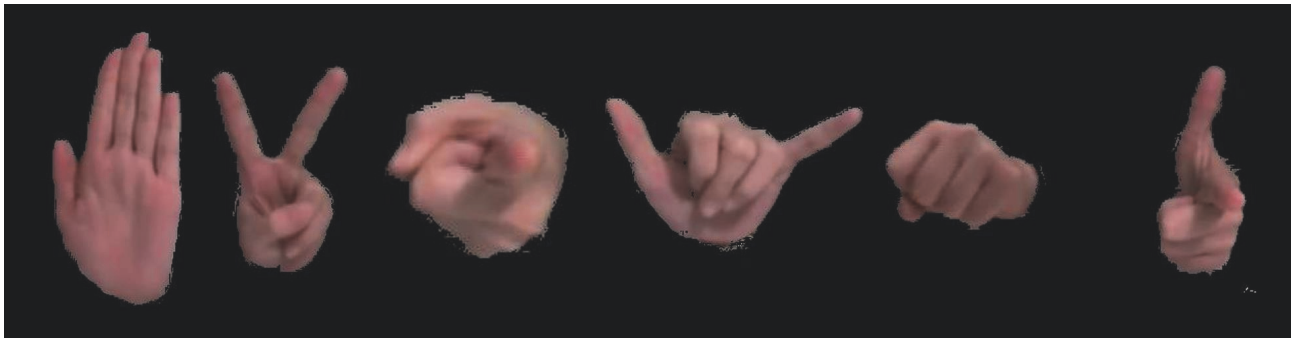
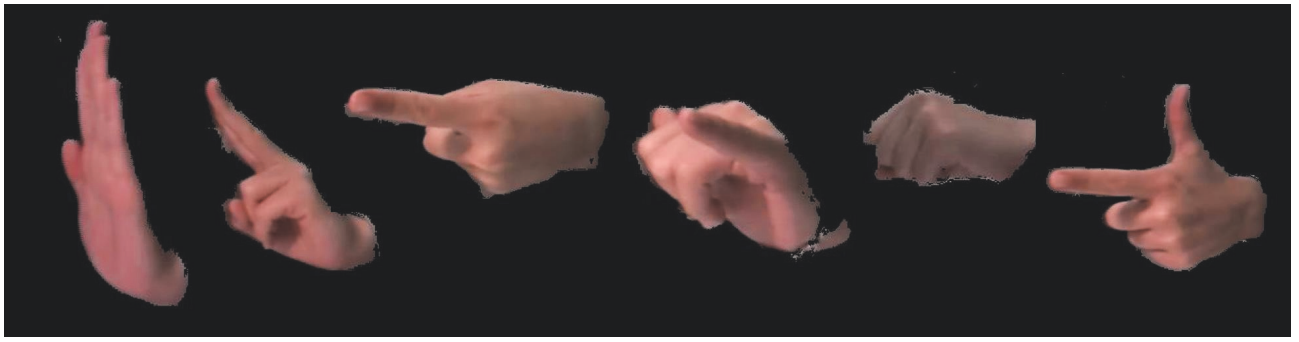


Figure 24: The pipeline of generating histogram of visual word for each image based on the detected feature points.



(a) Front view of 6 posture classes



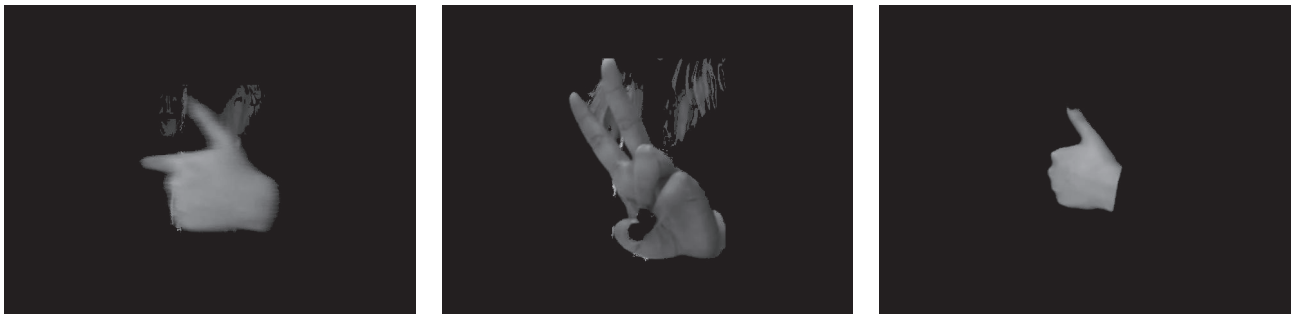
(b) Side view of 6 posture classes

Figure 25: The front view and side view of 6 posture classes.

called gesture vector. A gesture vector is extracted from each gesture video and used for classification by SVM. The details of how to define a feature vector are introduced in the following. Also, the performance of gesture recognition is evaluated in this section.

7.1. Gesture Vector. A gesture vector is constructed from two components: posture elements p_{ij} and trajectory element T. T indicates the recognized trajectory class. Depending on the complexity of gestures, each gesture can be separated into i segments to deal with posture variations. p_{ij} represents the occurrence number of the recognized posture class j in segment i. Equation (13) shows a gesture vector with i segments and j posture class.

$$\mathbf{v}_g = \left[p_{11}, p_{12}, p_{13}, \ldots, p_{ij}, \ldots, p_{mn}, T\right]. \tag{13}$$



(a) (b) (c)

Figure 26: Examples of misclassified postures.

Table 10: The names of gesture classes and their compositions.

Gesture	Posture		Trajectory
Grab	Palm → Fist	+	Forward
Hit	Fist	+	Forward
Call	Y	+	Circle
Poke	Point	+	Forward
Zoom	Eight	+	Left
Move	V → eight	+	Right
Push	Right and left palms	+	Forward
Collision	Right and left fists	+	Left and right

The gesture made in different speed would generate a big number difference on recognized postures. Because the frame rate is fixed for cameras, the faster the hand is moving, the fewer postures can be taken. Therefore, the posture elements in gesture vector are normalized by the total number of recognized postures for each section P_m. Equation (13) is rewritten as

$$\left[\frac{p_{11}}{P_1}, \frac{p_{12}}{P_1}, \frac{p_{13}}{P_1}, \ldots, \frac{p_{ij}}{P_i}, \ldots, \frac{p_{mn}}{P_m}, T\right]. \quad (14)$$

The gesture vector is extracted for each hand gesture and adopted for gesture classification.

7.2. Defining Gesture Classes and Gesture Vectors. To test the recognition performance, we defined eight classes of hand gesture. Both one-hand and two-hand gestures are included. Although we only defined eight classes of gesture for evaluation, more gesture classes can always be added to the system with some training data and gesture definitions. Figure 27 shows the combination of hand moving trajectory and posture. The blue arrows represent the hand moving trajectory, and the posture changing is also shown in this figure.

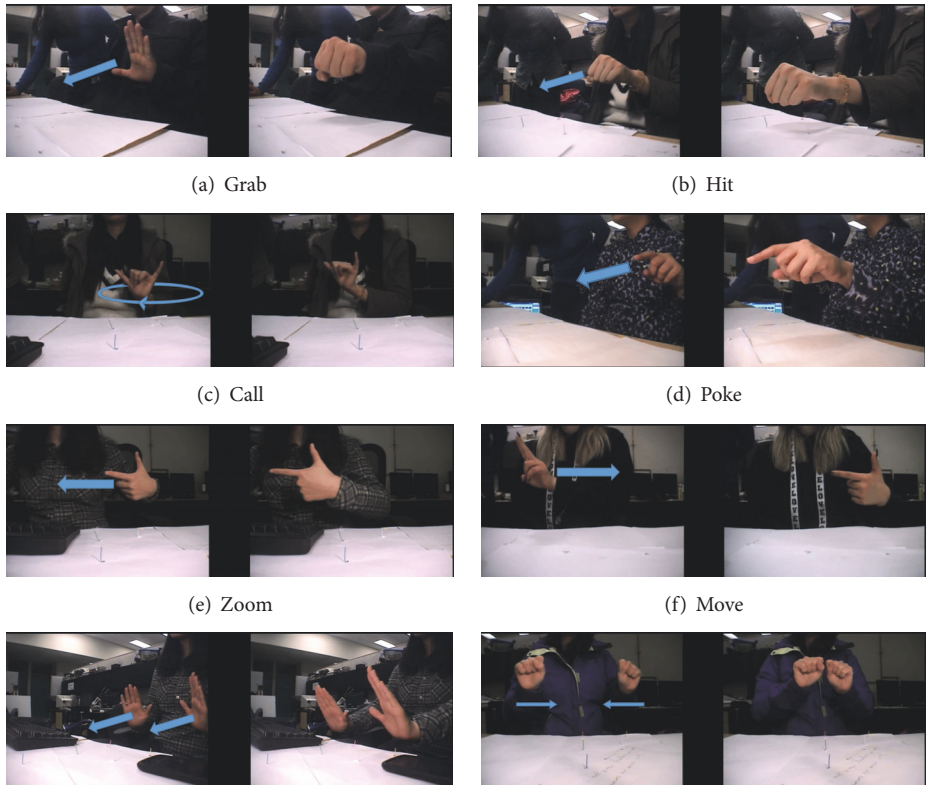
A list of the gesture name and its component of posture and trajectory is listed in Table 10. Since the defined gestures only contain one or no posture change, each trajectory is segmented into two sections which make the gesture vector $2 \times 6 + 1 = 13$ elements long for one-hand gesture. In two-hand case, the gesture vector for each hand is extracted for each hand and then combined together which makes the two-hand gesture vector twice as long as the one-hand case, and it contains $2 \times 6 + 1 + 2 \times 6 + 1 = 26$ elements.

Table 11: The cross-validation accuracy on training gestures.

Gesture	Single-hand gesture	Double-hand gesture
Accuracy	94%	100%

7.3. Training Stage for Gesture Classifier Using SVM. In the training stage, 10 gestures for each class, 80 gestures in total, are collected for the training stage among four people. The one-hand and two-hand posture models are trained separately by SVM. The ground truth of trajectory and the recognition result of hand postures are utilized to compose the gesture vector. A linear kernel SVM is trained which obtained a 5-fold cross-validation accuracy at 94% for single-hand gestures and 100% percent for double-hand gestures. The training accuracy for hand gestures is listed in Table 11.

7.4. Testing Stage for Gesture Recognition. For this section, another 80 gestures are collected for testing. These 80 gestures, 10 for each gesture class, are collected among four people. Based on how many hand postures are detected in the initial areas, the gesture would be classified as single-hand or double-hand gesture automatically. Then, based on the trajectory and posture recognition for each hand, the trained SVM model would recognize each hand gesture based on its



(a) Grab (b) Hit

(c) Call (d) Poke

(e) Zoom (f) Move

(g) Push hard (h) Collision

Figure 27: The example of eight gesture classes. The beginning and ending frames are shown, and the moving directions are shown in blue arrows.

Table 12: Confusion matrix for gesture recognition on the testing set.

Gesture	Grab	Hit	Call	Poke	Zoom	Move	Push	Collision
Grab	6	2		2				
Hit		8		2				
Call			10					
Poke				10				
Zoom					10			
Move						10		
Push							10	
Collision								10
Accuracy	60%	80%	100%	100%	100%	100%	100%	100%
					92.5% (74/80)			

gesture vector. Figure 28 shows the testing pipeline. Table 12 shows the recognition results and accuracies for each class.

All the postures are well recognized except for classes "Grab" and "Hit." This is because, at posture recognition stage, the trained classifier could misclassify "Fist" into "Point" (Table 8) due to the similarity of these two postures. Therefore, there is a high chance of misclassification for this class. For the gestures that involve posture changing, there would be a period of interval where postures are not defined in the posture classes. With our posture recognition scheme, such posture would have a high chance to be discarded, while the posture recognition results are not consistent. For instance, in the class "Move," the interval posture for changing postures "V" to "Eight" is discarded and acquires a high recognition accuracy.

8. Conclusions and Future Works

This paper proposed a vision-based approach for both trajectory and posture recognition of hand using multiple calibrated cameras. A fixed-range CbCr is adopted for hand area segmentation. Kalman Filter is adopted for hand tracking in the presence of occlusion or background noise. It was found that the hand trajectory contains both temporal and spatial

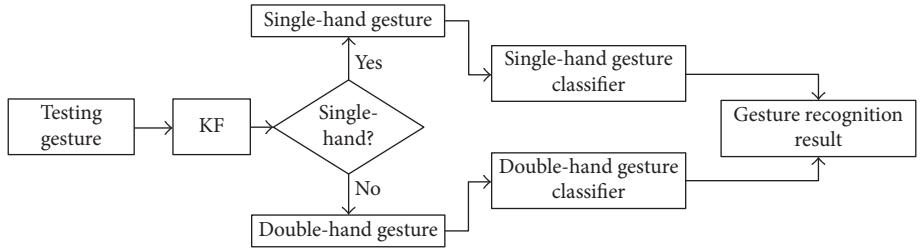


Figure 28: The testing pipeline of gesture recognition.

patterns that can vary among different individuals. DMP is applied as a method that could preserve the spatiotemporal information in weighted vectors. For example, topological similar trajectories would have similar weight vectors that can be utilized as feature vectors for trajectory classification using SVM. It was shown that with only a few trajectory training data, the recognition can be achieved with an accuracy of 88.0%.

Feature points of a hand posture are detected using SIFT method. A bag-of-words approach is employed to represent each posture as a unisize histogram vector. Such histogram vector is used for posture recognition using SVM. The training postures contain both front and side views taken by the cameras. The hand posture is only considered as being recognized when the recognition results from both camera views are consistent. With such scheme, although some of the postures are taken as unrecognizable which are then discarded (30%), the recognition rate can have an accuracy of 96.6%. For hand gesture recognition, a gesture feature vector is defined which combines the results of both trajectory and posture recognition. In our experimental study, gesture recognition using SVM approach has resulted in 92.5% accuracy.

8.1. Future Work. A combination of RGB and depth sensors (RGB-D) can be utilized in future for determining various silence features when combining both sensing modalities. In particular, various features can be associated with each of the scan planes in the depth map which can be used as a part of a deep learning algorithm [58]. The current experimental setup utilizes two calibrated cameras placed at a fixed distance to the hand. For applications where the cameras cannot be placed in proximity of the users (e.g., for the case of elderly living and for application of elderly computer interface), pan-tilt-zoom cameras can be deployed [59]. This active camera setup can offer a robust approach where one camera can zoom in on the hand of the user for gesture recognition, while following (through pan and tilt movements) the predicted movement pattern of the hand which is obtained through the other cameras.

Conflicts of Interest

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

References

[1] M. Albert, *Silent Messages*, Wadsworth publisher, Belmont, calf, USA, 1971.

[2] E. T. Hall, *The Silent Language*, Anchor publisher, New York, NY, USA, 1973.

[3] D. J. Sturman and D. Zeltzer, "A Survey of Glove-based Input," *IEEE Computer Graphics and Applications*, vol. 14, no. 1, pp. 30–39, 1994.

[4] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D hand pose reconstruction using specialized mappings," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 1, pp. 378–387, 2001.

[5] R. Wang and J. Popovic, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, article 63, 2009.

[6] S. Strachan, R. Murray-Smith, I. Oakley, and J. Ängeslevä, "Dynamic primitives for gestural interaction," in *Proceedings of the International Conference on Mobile Human-Computer Interaction*, vol. 3160, pp. 325–330, 2004.

[7] T. G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill, "A Hand gesture interface device," in *Proceedings of the SIGCHI/GI Conference on Human Factor in Computing Systems and Graphics Interface*, pp. 189–192, 1987.

[8] S. Mitra and T. Acharya, "Gesture recognition: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311–324, 2007.

[9] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[10] Y. Wu and T. S. Huang, "Vision-based gesture recognition: a review," in *International Workshop on Gestue-Based Communication in Human-Computer Interaction*, pp. 103–115, 1999.

[11] Z. Liu, F. Hu, D. Luo, and X. Wu, "Visual gesture recognition for human robot interaction using dynamic movement primitives," in *systems, man and cybernetics*, pp. 2094–2100, IEEE, San Diego, CA, USA, October 2014.

[12] A. Utsumi and J. Ohya, "Multiple-hand-gesture tracking using multiple cameras," in *Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition*, 1999.

[13] D. Saxe and R. Foulds, "Toward robust skin identification in video images," in *Proceedings of the International Conference on Automatic Face And Gesture Recognition*, pp. 379–384, 1996.

[14] D. Chai and K. N. Ngan, "Face segmentation using skin-color map in videophone applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 4, pp. 551–564, 1999.

[15] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 46, no. 1, Article ID 390108, pp. 81–96, 2002.

[16] H. Greenspan, J. Goldberger, and I. Eshet, "Mixture model for face-color modeling and segmentation," *Pattern Recognition Letters*, vol. 22, no. 14, pp. 1525–1536, 2001.

[17] M. Yang and N. Ahuja, "Gaussian mixture model for human skin color and its application in image and video databases," in *Proceedings of The SPIE: Storage And Retrieval for Image and Video Databases VII*, pp. 458–466, 3656, 1999.

[18] X. Liu and K. Fujimura, "Hand gesture recognition using depth data," in *Proceedings of the IEEE International Conference on Automatci Face and Gesture Recognition*, pp. 529–534, 2004.

[19] Z. Ren, J. Meng, and J. Yuan, "Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction," in *Proceedings of the 8th International Conference on Information, Communications and Signal Processing (ICICS '11)*, pp. 1–5, IEEE, December 2011.

[20] Y. Wang, T. Yu, L. Shi, and Z. Li, "Using human body gestures as inputs for gaming via depth analysis," in *Proceedings of the 2008 IEEE International Conference on Multimedia and Expo, ICME 2008*, pp. 993–996, deu, June 2008.

[21] C. Diaz and S. Payandeh, "Multimodal Sensing Interface for Haptic Interaction," *Journal of sensors*, vol. 2017, Article ID 2072951, 24 pages, 2017.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[23] S. Ahmad, "A usable real-time 3D hand tracker," in *Proceedings of the IEEE Conference on Signals, Systems and Computers*, pp. 1257–1261, Pacific Grove, CA, USA, 1994.

[24] Y. Lu and S. Payandeh, "Cooperative hybrid multi-camera tracking for people surveillance," *Canadian Journal of Electrical and Computer Engineering*, vol. 33, no. 3-4, pp. 145–152, 2008.

[25] J. Park and Y.-L. Yoon, "LED-glove based interactions in multi-modal displays for teleconferencing," in *Proceedings of the 16th International Conference on Artificial Reality and Telexistence - Workshops, ICAT 2006*, pp. 395–399, chn, December 2006.

[26] H. Kim and D. W. Fellner, "Interaction with hand gesture for a back-projection wall," in *Proceedings of the Proceedings - Computer Graphics International, CGI 2004*, pp. 395–402, grc, June 2004.

[27] W. T. Freeman and C. Weissman, "Television Control by Hand Gestures," in *Proceedings of the International Workshop on Automatic Face and Gesture Recognition*, pp. 179–183, 1995.

[28] R. A. Bolt, "Eyes at the Interface," in *Proceedings of Conference on Human Factors in Computing Systems*, pp. 360–362, Gaithersburg, Md, USA, March 1982.

[29] T. Starner, J. Weaver, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.

[30] D. M. Gavrila and L. S. Davis, "3-D model-based tracking of humans in action: a multi-view approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 73–80, June 1996.

[31] Y. Li, "Hand gesture recognition using Kinect," in *Proceedings of the 2012 IEEE 3rd International Conference on Software Engineering and Service Science, ICSESS 2012*, pp. 196–199, chn, June 2012.

[32] H. Zhou, L. Xie, and X. Fang, "Visual Mouse: SIFT Detection and PCA Recognition," in *Proceedings of the IEEE International Conference on Computational Intelligence and Security*, pp. 263–266, 2007.

[33] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proceedings of the 15th ACM International Conference on Multimedia (MM '07)*, pp. 357–360, September 2007.

[34] C. Wang and K. Wang, "Hand Posture recognition using adaboost with SIFT for human robot interaction , recent progress in robotics," *Springer Lecture Notes in Control And Information Science*, pp. 317–329, 2008.

[35] S. Pandita and S. P. Narote, "Hand gesture recognition using SIFT," *Proceedings of the IEEE International Conference on Computational Intelligence and Security*, vol. 2, no. 1, p. 4, 2013.

[36] C. Qing, N. D. Georganas, and E. M. Petriu, "Real-time vision-based hand gesture recognition using haar-like features," in *Proceedings of the IEEE Conference on Instrumentation And Measurement Technology*, pp. 1–6, IEEE, Warsaw, Poland, May 2007.

[37] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," in *Proceedings of the International Workshop on Automatic Face And Gesture Recognition*, pp. 296–301, 1995.

[38] W. C. Stokoe and M. Marschark, "Sign language structure: An outline of the visual communication systems of the american deaf," *Journal of Deaf Studies and Deaf Education*, vol. 10, no. 1, pp. 3–37, 2005.

[39] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, "A Hidden markov model-based continuous gesture recognition system for hand motion trajectory," in *Proceedings of the IEEE International Conference on Pattern Recognition*, pp. 1–4, December 2008.

[40] S. Payandeh, *Visual Tracking in Conventional Minimally Invasive Surgery*, CRC Press, Cleveland, Ohio, USA, 2016.

[41] F. Bashir, W. Qu, A. Khokhar, and D. Schonfeld, "HMM-based motion recognition system using segmented PCA," in *Proceedings of the IEEE International Conference on Image Processing 2005, ICIP 2005*, pp. 1288–1291, ita, September 2005.

[42] F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Object trajectory-based activity classification and recognition using hidden Markov models," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1912–1919, 2007.

[43] P. Hong, M. Turk, and T. S. Huang, "Gesture modeling and recognition using finite state machines," in *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2000*, pp. 410–415, fra, March 2000.

[44] S. Schaal, "Dynamic movement primitives- a framework for motor control in humans and humanoid robotics," in *In Springers Adaptative Motion of Animals And Machines*, pp. 261–280, Springer, Germany, 2006.

[45] J. Letessier and F. Bérard, "Visual tracking of bare fingers for interactive surfaces," in *Proceedings of the ACM Symposium on User Interface Software and Technology*, pp. 119–122, 2004.

[46] J. Wang and S. Payandeh, "A study of hand motion/posture recognition in two-camera views," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 9475, pp. 314–323, 2015.

[47] Bouguet, J., Camera Calibration Toolbox for Matlab, http://www.vision.caltech.edu/bouguetj/calib_doc/, Lasted accessed October 2017.

[48] Y. Fang, K. Wang, J. Cheng, and H. Lu, "A Real-Time Hand Gesture Recognition Method," in *Proceedings of the IEEE international conference on multimedia and expo*, pp. 995–998, 2007.

[49] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color pixel classification: Analysis and comparison," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 1, pp. 148–154, 2005.

[50] K. Sobottka and I. Pitas, "A novel method for automatic face segmentation, facial feature extraction and tracking," *Signal Processing: Image Communication*, vol. 12, no. 3, pp. 263–281, 1998.

[51] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[52] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: a survey," *ACM Computing Surveys*, vol. 38, no. 4, article 13, 2006.

[53] J. Wang, *Hand Tracking and its Pattern Recognition in a Network of Calibrated Cameras [Masters of applied science thesis]*, Simon Fraser University, Canada, 2015.

[54] W. S. leveland, "LOWESS: a program for smoothing scatterplots by robust locally weighted regression," *The American statistician*, vol. 35, no. 1, 1981.

[55] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[56] C. Chang and C. Lin, "LIBSVM: a Library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, article 27, 2011.

[57] K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate Analysis*, Academic press, 1979.

[58] N. Mohsin, X. Liu, and S. Payandeh, "Signal processing techniques for natural sleep posture estimation using depth data," in *Proceedings of the 7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference, IEEE IEMCON 2016*, can, October 2016.

[59] Y. Lu and S. Payandeh, "Intelligent cooperative tracking in multi-camera systems," in *Proceedings of the 9th International Conference on Intelligent Systems Design and Applications, ISDA 2009*, pp. 608–613, ita, December 2009.