Multimed Tools Appl (2016) 75:7065–7086
DOI 10.1007/s11042-015-2632-3

Depth-based hand gesture recognition

Chih-Hung Wu [1] · Wei-Lun Chen [1] · Chang Hong Lin [1]

Received: 16 July 2014 / Revised: 9 March 2015 / Accepted: 13 April 2015 /
Published online: 23 April 2015

Abstract In this article, a dynamic gesture recognition system with the depth information is proposed. The proposed system consists of three main components: preprocessing, static posture recognition and dynamic gesture recognition. In the first component, the background subtraction is used to exclude invalid gestures that is not generated by the main user, and then to detect and track the hand. Second, the region of hand is extracted using an adaptive square. Once the region of hand is obtained, the features of hand are extracted and the static hand posture are classified using the support vector machine (SVM). Finally, nine commonly used dynamic hand gestures can be detected using different methods. In the experiments, the static hand posture classification was evaluated in different postures and the performance of dynamic gesture recognition is verified by two different persons at 4 different position with 2 different depths. The experiment results show that the proposed system can accurately detect the dynamic hand gestures with an average recognition rate of 87.6 %, which is good for controlling the embedded systems, such as home appliances.

Keyword Depth cameras · Static hand posture recognition · Dynamic hand gesture recognition · Support vector machine

1 Introduction

Human-machine interaction is a comprehensive research field, where the purpose is to explore the interactions between humans and computers or electronic devices. The hand gesture recognition is one of natural and intuitional way to communicate with human and machine. Hand gesture recognition can be widely applied to many applications such as home appliance, entertainment and medical systems, etc. [24].

In the recent years, depth information is a particularly useful cue in human machine interface (HMI) applications. Since low-cost depth cameras have been launched, depth

✉ Chang Hong Lin
chlin@mail.ntust.edu.tw

1 Department of Electronic and Computer Engineering, National Taiwan University of Science and Technology, Taiwan, Republic of China

cameras become more and more affordable for consumer electronics. Depth cameras can work in various situation where the RGB cameras cannot, such as in low lighting and different illumination conditions. Using low-cost depth cameras can create a new opportunity for the home appliance control. In the key insights of [24], no single method for automatic hand gesture recognition is suitable for every applications. Accordingly, hand gesture recognition using only depth information for embedded control needs to be explored.

In this article, a dynamic hand gesture recognition system using only the depth information is proposed for embedded control. In order to improve the system feasibility, an adaptive square is used to extract the hand with different people in the different depth. With the help of the adaptive square, the proposal system can appropriately obtain the hand region without the wrist area. Eight types of static hand postures are presented in this article. These static hand postures can be adopted by both the left and right hands. These postures would form the basis of dynamic hand gestures. The dynamic hand gestures also can be adopted by both the left and right hands to improve the system feasibility. In order to explore the applicability of depth sensors, only the depth information is used to develop the proposed system. Depth sensors can provide accurate and appropriate depth data for hand gesture recognition. Nine commonly used dynamic hand gestures are included for embedded system control. These gestures can be recognized at different positions with different depths. By using the proposed methods, the experiment results show that the proposed system can recognize several commonly used dynamic hand gestures and can function at different depths.

This article is organized as the follows. Section 2 gives some related work. Section 3 describes the procedure of the proposed system. Experiment results are shown in Section 4. Finally, Section 5 concludes of this article.

2 Related work

Hand gesture recognition has been worked on for a long time, and the number of researches in the depth based hand gesture recognition grew rapidly in recent years since low-cost depth cameras have been released. In the following subsections, various camera modality systems are introduced, and the methods for each type of camera are described as well.

2.1 RGB cameras

Many hand gesture recognition systems detect hands using skin-color modeling and detection [5, 14, 27]. Liu et al. [14] transformed the color space of images from RGB to YCbCr and then two sets of threshold values for Cb and Cr were used for discriminating skin and non-skin pixels. Zhu et al. [27] converted RGB images into YCbCr images and then used the K-means clustering algorithm for representing the hand object and background. Choi et al. [5] segmented a hand-forearm region using a generalized statistical skin color model of the image and distance transform. Statistical skin color model is to built a 3D RGB histogram model to represent the distribution of skin tones in the color space. A comprehensive overview of skin-color modeling and detection methods can be found in [10]. Senanayake et al. [21] proposed a hand gesture based appliance control system. Barkoky et al. [1] proposed a method to recognize the numbers of Persian sign language (PSL) using thinning method. Huang et al. [9] presented a novel method for hand gesture recognition based on Gabor filters and support vector machine (SVM). A common approach to describe dynamic hand gesture is to use statebased model, such as the Hidden Markov Models (HMMs). Many dynamic hand gesture recognition systems are based on the HMM [22, 26]. Shrivastava [22] proposed a novel and faster system for dynamic hand gesture recognition. Huang et al. [26] introduced an HMM-based method to recognize complex single hand gestures.

2.2 Depth cameras

Depth information makes the hand segmentation much easier. Many researchers assume that the hand was the closest object to the camera [4, 12, 16, 20]. Consequently, region growing [4, 16] techniques can be used to extract hand regions using the depth information. A number of researchers took advantage of both the color and depth information to develop their hand gesture recognition system [20, 23, 27]. In contrast to hand gesture recognition using color image, few researchers develop their hand gesture recognition system using only the depth information [11, 12, 15, 16]. Minnen et al. [15] presented a hand shape (gesture) recognition method that made use of three different kinds of image features to characterize segmented hand patches. In [11], a shape classification forest (SCF) was used to classify each depth pixel of hand into a hand shape and the final hand shape was determined by voting. In [16], Oprisescu et al. presented an automatic method for static hand gesture recognition using a ToF camera. In [12], Kurakin et al. proposed a real-time system based on the action graph for dynamic hand gesture recognition. Action graph, which proposed by Li et al. [13] for human action recognition, requires less training data compare to the HMM.

The proposed system uses only the depth information to recognize static and dynamic gestures. The system uses SVM for static posture classification because of its low overhead and efficient classification. The SVM is trained using the shape description of hand to recognize several different static hand postures, and then use the static hand postures, fingertips and hand trajectories and combined to determine the nine different dynamic hand gestures.

3 The proposed system

The proposed system is an extension of our previous work [25] with two new gestures, rotate and scale, and more details.

The flowchart of the proposed system is shown in Fig. 1, which system consists of three main procedures: preprocessing, static gesture recognition and dynamic gesture recognition, and can be further divided into nine parts. The first stage, preprocessing, includes four parts, acquirement of depth images, background subtraction, hand detection and hand tracking. The second stage, static gesture recognition, includes three parts, region of hand extraction, hand feature extraction and static hand posture classification. The third stage, dynamic gesture recognition, includes two parts, trajectory of hand extraction and dynamic hand gesture determination.

3.1 Preprocessing

When the user controls embedded systems using hand gestures, the user typically stays close to the camera to interact with the appliance and the user's hands are usually in front of his or her body as well as the background. Accordingly, we can assume the main user stays close to the camera and occupies a significant portion of the camera's field of view (FoV). These

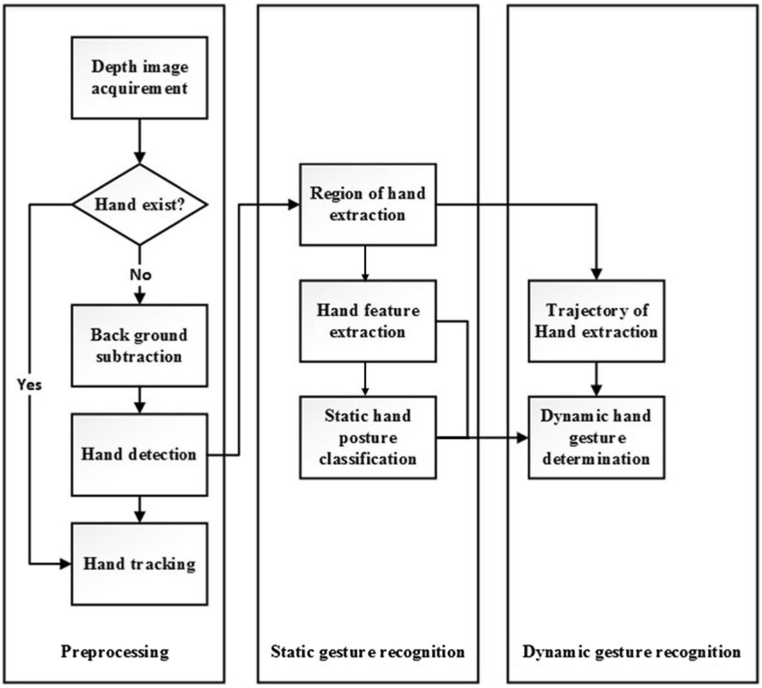


Fig. 1 Flowchart of the system

assumptions are reasonable for numerous practical scenarios. According to the datasheet of Prime Sense [17], the depth camera can only provide the depth of objects within 800 to 3500 mm. In order to detect the hand gesture, the proposed system requires a hand region larger than 1500 pixels, so the user has to stay within 1100 to 1800 mm from the camera. In general, the original depth image contains not only the human body region but also the complex background, as illustrated in Fig. 2a. For the purpose of detecting the main user's hand and excluding invalid gestures that is not generated by the main user, the largest region within 1100 to 1800 mm is extracted as the user's body to filter out the background objects, as shown in Fig. 2b. Figure 2c is the background subtraction without a user within 1100 to 1800 mm.

For fully automatic control appliances, a way to detect the hand is needed to provide the initial hand center position. The prototype system simply used the functions provided by the OpenNI [18] for both hand detection and trucking, which uses the Bayesian Object Localization for hand detection [19]. The detected hand with hand center position is shown in Fig. 2d, with the white point represents the hand center position.

3.2 Static posture recognition

The static posture recognition stage is composed of region of hand extraction, hand feature extraction and static hand posture classification. Region of hand extraction is to find a squire region that can appropriate contain the region of hand without the wrist. Hand feature extraction is to find the fingertip positions. The contour of the hand between the fingertips

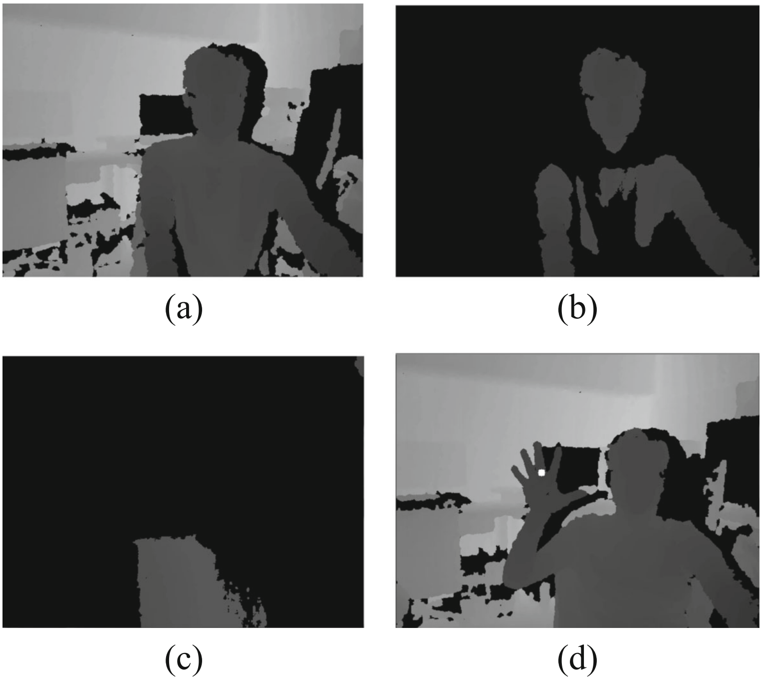


Fig. 2 **a** Original depth image **b** Background subtraction with user **c** Background subtraction without user **d** Hand with center position

of the thumb and the index finger is then used to classify eight sets of static hand posture. The fingertip positions and the static hand postures are used for dynamic gesture recognition.

Once background subtraction is achieved and the hand center position is obtained, the next step is to extract the region of hand for further image processing. Since the size of hand region on the image is different according to the distance between the hand and the camera, the system cannot use a fixed-size rectangle to extract the region of hand. One of the advantage of the depth camera is that people can get the three-dimensional (3D) information rather than the two-dimensional (2D) information on the real world. Therefore, the system acquires a square that can approximately contain a 10 cm×15 cm hand (the average size of an adult hand) according to the depth information (z–axis). On the basis of our experiments, when the depth of hand is one meter away from the camera with a VGA resolution (640×480 pixels) depth image, the height of palm is about 87 pixels. However, the point generated by the hand tracking is not always in the middle of the hand, so the system cannot directly use the height of palm to obtain the square with its center at the hand tracking point. Accordingly, the width of square is extended to 112 pixels that can approximately contain a 10 cm×15 cm hand when the depth of hand tracking point is one meter from the camera. The width of square is adjustable according to the depth of the hand tracking point by the following formula:

$$\mathrm{w}_{\mathrm{square}} = 112000/\mathrm{depth}_{\mathrm{point\ of\ hand\ tracking}}$$

where $\mathrm{w}_{\mathrm{square}}$ denotes the width of square with pixel, depth $_{\mathrm{point\ of\ hand\ tracking}}$ denotes the depth of the hand tracking point in millimeter.

First of all, an initial square with its center at the hand tracking point is computed to extract the region of hand and its width is determined according to the depth of the hand tracking point. The initial square is then used to estimate the ratio of hand pixels to the square region to refine the square to remove the wrist area. Based on our observation and experiments, the appropriate ratio of hand pixels to the square region is 0.35 to 1. The square can be adopted by both the left and right hands, as illustrated in Fig. 3. Figure 3a is the result of left hand region extraction and Fig. 3b is the result of right hand region extraction, with the white square region represents the extracted hand region. Figure 3c and d are the results of hand region extraction with different people in the same depth, with the white number represents the depth. Figure 3e and f are the results of hand region extraction in different depths.

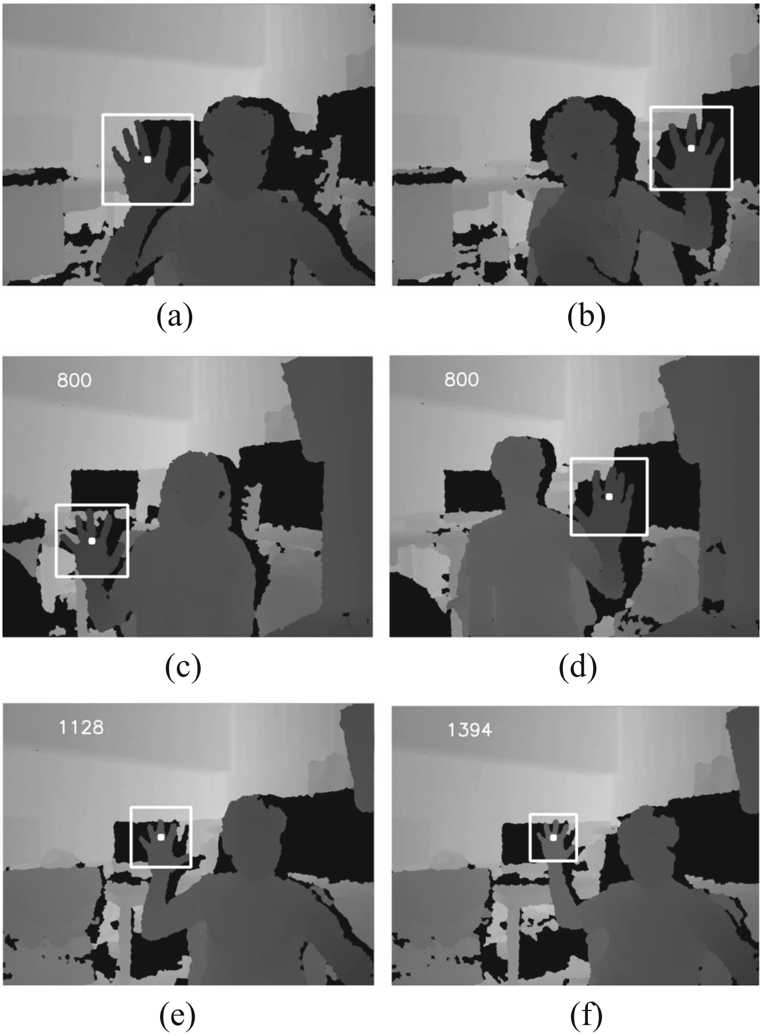


Fig. 3 Results of hand region extraction (**a**) left hand (**b**) right hand (**c**) with user1 (**d**) with user2 (**e**) in the 1128 mm depth (**f**) in the 1394 mm depth

After the region of hand is obtained, the next step is to extract the features of hand. Figure 4a is the flow chart of hand feature extraction, and the pseudo code of

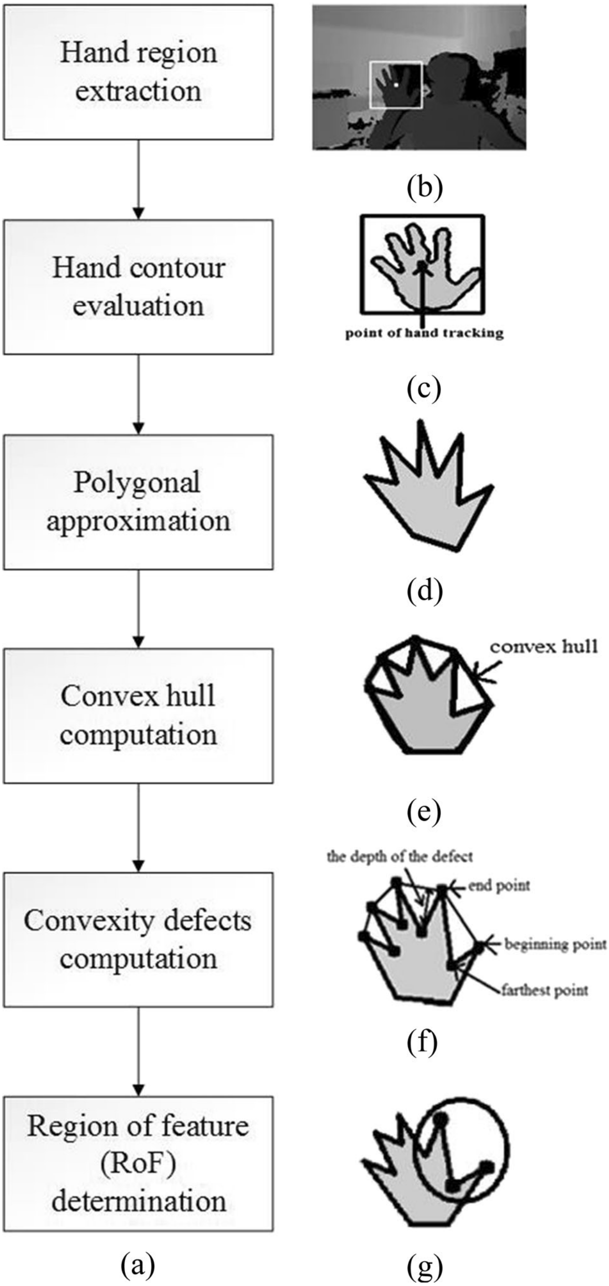


Fig. 4 **a** The flow chart of hand feature extraction **b** the region of hand **c** the contour of hand **d** the Contour with polygonal approximation **e** the convex hull of the hand **f** the convexity defects **g** Region of feature and three points for posture classification

feature extraction algorithm is shown in Fig. 5. One useful way of realizing the shape of hand is to compute a convex hull for the hand and then compute its convexity defects [2]. Before computing the convex hull, pixels with depth value farther than the hand depth are removed from the region of hand. The contour of hand which contains the largest contour area is then evaluated from the remaining pixels, as shown in Fig. 4c. The contour is then simplified by the polygonal approximation [6] as several straight lines in Fig. 4d.

In the proposed system, the Graham scan algorithm [7] is used to compute the convex hull of the hand. Figure 4e shows the result of the convex hull of hand using the Graham scan algorithm. The convex hull covers a given nonempty set of hand contour points. Once the contour and the convex hull are computed, the next step is to compute the convexity defects for finger identification. In the proposed system, the convexity defects were computed using the method proposed by Homma and Takenaka. [8].

Algorithm 1: feature extraction algorithm

Inputs:

1. the region of hand RoH

Output:

1. the region of feature RoF

Begin

Step1: . Obtain the RoH

// as shown in fig. 4 (b)

Step2: If RoH is obtained and pixels of region of hand with depth value farther than the hand depth, then removed the RoH.

Step3: Compute the contour of region of hand.

Step4: Select the largest contour area as the contour of hand. // as shown in fig.4 (c)

Step5: Compute the polygonal approximation of the contour of hand. //as shown in fig.4 (d)

Step6: Compute the convex hull of hand. //as shown in fig.4 (e)

Step7: Compute the convexity defects $\{D_n\}_{n=0}^{N}$. //as shown in fig.4 (f)

Step8: Select RoF= distance ($D_{beginning}$, D_{end}) > 0.43 * RoH_{width} and D_{depth} > 10

Where $D_{beginning}$ is the points on the hull where the defect begins.

D_{end} is the points on the hull where the defect ends.

RoH_{width} is the width of the region of hand.

D_{depth} is the depth of the defect.

Step9:

Return the region of feature RoF, which consist of $D_{beginning}$, $D_{farthest}$ and D_{end}.

//as shown in fig.4 (g)

Where $D_{beginning}$ is the points on the hull where the defect begins.

$D_{farthest}$ is the farthest from the edge of the hull

D_{end} is the points on the hull where the defect ends.

End

Fig. 5 The pseudo code of feature extraction algorithm

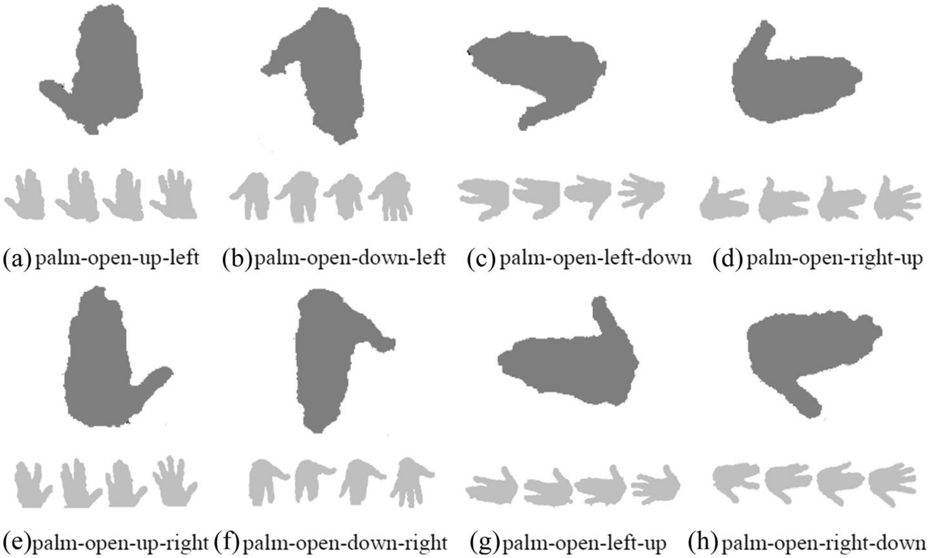


Fig. 6 Sample images showing the eight sets of static hand postures evaluated in this paper

A result of convexity defects is shown in Fig. 4f. It is clear that there are four white regions which the convex hull comprises but not contained in the contour. These regions are the "defects" relative to the hull. The beginning point and end point are the points on the hull where the defect begins and ends. The farthest point indicates the point on the defect which is the farthest from the edge of the hull. Except the beginning point, end point and the farthest point, other useful information is the depth of the defect. The depth of the defect is the distance between the farthest point and the edge of the hull. Otherwise, the beginning and the end point are used to determine the fingertip's position. The positions of fingertips are then used for the static hand posture classification and dynamic gesture recognition.

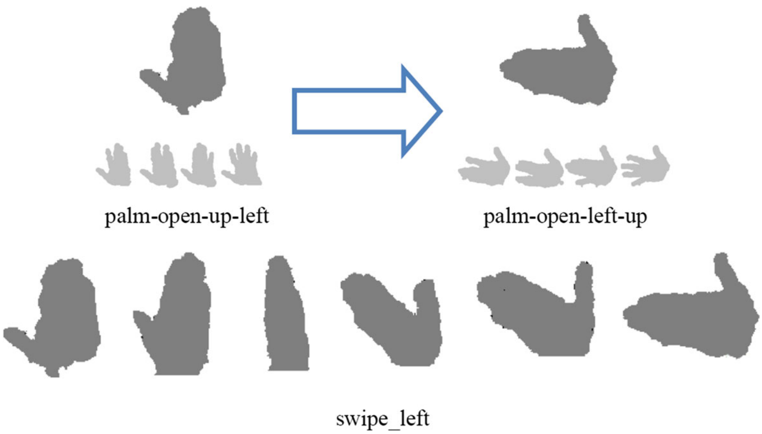


Fig. 7 An example of the transition of dynamic gesture

Table 1 Dynamic hand gestures

Name of gesture	Explanation of gesture	Category
swipe_up	Hand swipe from down to up	static hand posture based
swipe_down	Hand swipe from up to down	static hand posture based
swipe_left	Hand swipe from up to left	static hand posture based
swipe_right	Hand swipe from up to right	static hand posture based
wave	Hand wave from up-left to up-right or up-right to up-left twice	static hand posture based
circle	Use hand to draw a circle	hand trajectory based
push	Hand fast move from far to near	hand trajectory based
rotate	Use hand to rotate something	fingertip based
scale	Use hand to scale something	fingertip based

To obtain the RoF, the fingertip positions of the thumb and the index finger are obtained according to the distance between the beginning and end points of the defect and the depth of the defect. Based on our experiments and observation, the distance between the beginning and end points should be larger than 43 % of the width of the hand region, and the depth of the defect should be longer than 10 pixels. The fingertip positions of the thumb and the index

Table 2 Dynamic hand gestures based on static postures

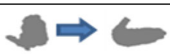
Name of gesture	Static posture transition
swipe_up	
swipe_down	
swipe_left	
swipe_right	
wave	 *2 /  *2

Fig. 8 Transitions of dynamic hand gestures based on static postures

(a) swipe_up (right hand)

(b) swipe_up (left hand)

(c) swipe_down (right hand)

(d) swipe_down (left hand)

(e) swipe_left (left and right hand)

(f) swipe_right (left and right hand)

(g) wave (left hand)

(h) wave (right hand)

finger are found to locate the RoF. The beginning, farthest and end points of convexity defect are used to represent the RoF for posture classification are shown in Fig. 4g. To classify eight sets of static hand postures with different kinds of region of features, the LIBSVM [3] is used as the SVM classifier for static hand posture classification.

For the purpose of characterizing the dynamic hand gestures, eight sets of static hand postures are selected to describe the transition, as shown in Fig. 6. For example, the transition from the palm-open-up-left static hand posture to the palm-open-left-up static hand posture can be used to describe the swipe_left dynamic hand gesture, as shown in Fig. 7. The proposed system first extracts the feature descriptors of the hand, and then classifies the hand into possible postures.

The system uses the contour of the hand between the fingertips of the thumb and the index finger, including the first dorsal interosseous, as the region of feature (RoF) for posture classification, as shown in Fig. 4g.

3.3 Dynamic gesture recognition

For embedded system control, the proposed system can recognize ten commonly used dynamic hand gestures, as shown in Table 1. The gestures can be further divided into three categories, which include static hand posture based, hand trajectory based and fingertip based methods. The swipe_up, swipe_down, swipe_left, swipe_right and wave gestures are based on static hand postures, the circle and push gestures are based on hand trajectories, and the scale, rotate and drag gestures are based on fingertips.

As stated in 3.2, eight sets of static hand postures are used to detect the dynamic hand gestures. Five types of dynamic hand gestures can be detected in the proposed system based on these static hand postures, as described in Table 2. Figure 8 shows the transition of five types of dynamic hand gestures. These five types of dynamic hand gestures can be adopted by the static postures flow from both the left and right hands. The transition of two static hand postures is used to describe the transition of each dynamic hand gesture. For every frame, the system would generate either zero or one possible static posture. In order to detect the dynamic gesture, the system used different intervals (up to 60 frames) to determine the swipe_up, swipe_down, swipe_left and swipe_right gestures with different speed. Each interval was divided into two subintervals with equal length: previous and recent. The posture for each subinterval is determined only when more than half of the static hand postures belong to the same posture, otherwise the posture of this subinterval is considered as unknown. The swipe_up, swipe_down, swipe_left and swipe_right gestures can be detected when the transition of two subintervals conforms with Table 2.

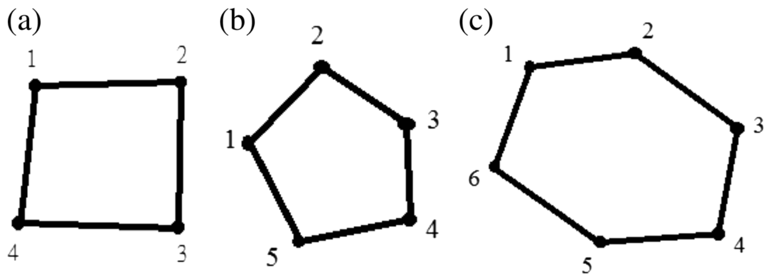


Fig. 9 **a** four samples **b** five samples **c** six samples

Table 3 Circle gesture analysis

User	Total times	4 samples		5 samples		6 samples	
		times	rate (%)	times	rate (%)	times	rate (%)
User 1	66	50	78 %	39	59 %	28	42 %
User 2	58	50	86 %	23	40 %	14	24 %
User 3	88	50	57 %	47	47 %	32	36 %
User 4	57	50	88 %	30	53 %	21	37 %
User 5	68	50	74 %	42	62 %	21	31 %

In order to detect the circle gesture, we analyze the trajectory of the hand tracking point is analyzed within a 36-frame interval. As show in Fig. 9, three different settings are used to determine the circle gesture in the photo type: Four, five and six samples of the hand trajectory with a 7, 6, and 6-frame gap between two samples, respectively. The circle gesture is detected according to the distance of x and y axis between every two successive samples and the size of the polygon region of these samples. Based on the experiments, the distance of x and y axis between every two successive samples should be larger than 15 pixels. Then, the size of the polygon region of these samples should be larger than 10,000 pixels and lower than 220,000 pixels. The circle gesture is analyzed by five different users performing the circle gesture until one of the three methods detect 50 of them. The results of circle gesture analysis are shown in Table 3. Four samples of hand trajectory with a 7-frame gap each has the highest recognition rate. Accordingly, four samples with a 7-frame gap each are chosen to obtain the circle gesture (less than a second). To detect the circle gesture with different speed, six samples with a 6-frame gap each are chosen to obtain a slow circle gesture (more than one second).

The two commonly used trajectory of push gesture, *push then pull* and *push then stay*, can be detected in the proposed system. *Push* means the hand moves forward to camera, *pull* means the hand moves backward from camera and *stay* means the hand without movements. The system used the depth difference of the hand tracking point between two consecutive frames to determine the depth movement. The system used different interval (up to 30 frames) to determine the push gesture with different speed. For each interval, the accumulated depth differences of the first and second halves are calculated to determine the posture as either *push*, *stay* or *pull*. The push gesture would be detected when the interval consists either *push* and *pull* or *push* and *stay*.

To detect the rotate gesture, the rotation of open hand is used to determine the rotate gesture as rotate left or rotate right. An ellipse is found around the hand contour and the center of

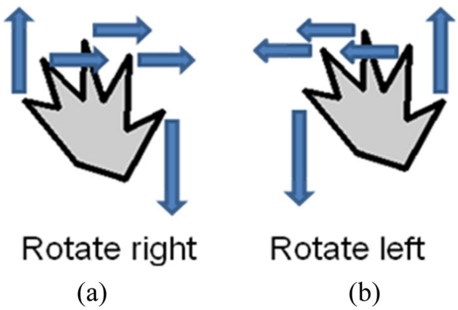


Fig. 10 Rotate gesture

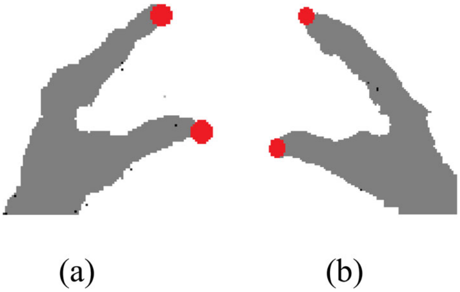


Fig. 11 Initial postures of scaling

ellipse is used as the center of hand, because the position of the center of hand cannot vary too much when the rotate gesture is performed. When the fingertips of the index finger, middle finger and ring finger all move from left to right, and the left-most fingertip (the fingertip of thumb or the little finger) move from down to up and the right-most fingertip (the fingertip of the little finger or thumb) move from up to down, this rotation is determined as rotate right, as shown in Fig. 10a. When the fingertips of the index finger, middle finger and ring finger all move from right to left, and the left-most fingertip (the fingertip of thumb or the little finger) move from up to down and the right-most fingertip (the fingertip of the little finger or thumb) move from down to up, we determine this rotation as rotate left, as shown in Fig. 10b. If the system cannot detect five fingertips, only the fingertips of thumb and the index finger are used to determine the rotate gesture as rotate left or rotate right. In the proposed system, the rotate right (left) can be detected when the rotate right (left) is determined four times in a small period.

To detect the scale gesture, an initial posture is used to start the scale gesture, as shown in Fig. 11 The initial posture can be detected when the number of convexity defect is one and the distance between the point of hand center and the farthest point of defect is less than 16 pixels. The distance between two fingertips (red points) are used to determine the scale gesture as scale up or scale down. In the proposed system, the scale up gesture can be detected when the distance between two fingertips become longer between every two consecutive frames for six consecutive frames, and the scale down gesture can be detected when the distance between two fingertips become shorter between every two consecutive frames for six consecutive frames.

4 Experimental results

We implemented all the proposed algorithms in the C++ programming language and the experiments were executed on a PC with an Intel i7 3.4 GHz CPU and 16GB memory with

Table 4 Xtion specification [17]

Item	Valid range
Depth sensor range	0.8 – 3.5 m
Horizontal field of view	58°
Vertical field of view	45°
Frame Per Second	VGA (640×480): 30fps
Depth Resolution	VGA (640×480)

Fig. 12 Examples of the eight difference static postures

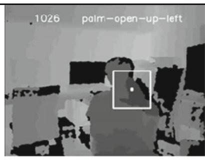
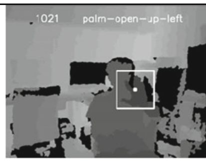
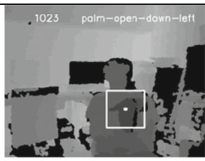
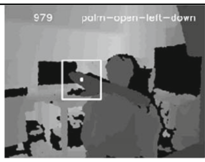
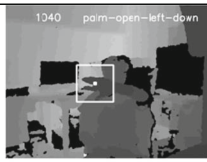
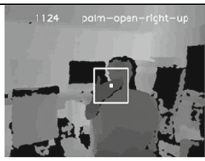
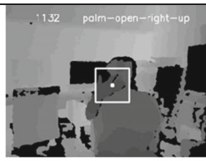
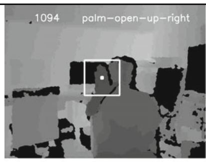
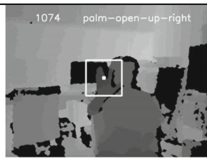
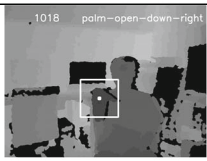
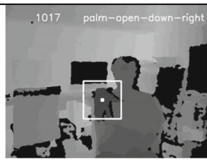
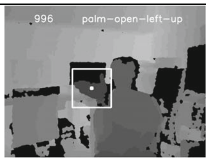
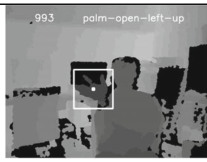
palm-open -up-left		
palm-open -down-left		
palm-open -left-down		
palm-open -right-up		
palm-open -up-right		
palm-open -down-righ t		
palm-open -left-up		
palm-open -right-dow n		

Table 5 The confusion matrix of static posture recognition

Posture	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
(a)	0.9606	0	0	0.0043	0	0	0	0
(b)	0	0.9829	0	0	0	0	0	0
(c)	0	0	0.9548	0	0	0	0	0
(d)	0	0	0	0.9456	0.0526	0	0	0
(e)	0	0	0	0	0.9925	0	0	0
(f)	0	0	0	0	0	0.9893	0	0
(g)	0	0	0	0	0	0.0306	0.9645	0
(h)	0	0	0	0	0	0	0	0.9886

(a) up-left *(b)* down-left *(c)* left-down *(d)* right-up *(e)* up-right *(f)* down-right *(g)* left-up *(h)* right-down

Windows 7 environment. An ASUS Xtion series depth camera was chosen as the input device. The input depth images are captured at 30 fps (frame per second), and the resolution of each frame is 640×480 pixels. Table 4 shows the specification of the depth camera [17].

4.1 Static postures

The training data set of static postures contains 5,185 samples, which was extracted from depth images acquired by ourselves, and the LIBSVM [3] is used to generate the classifier. The static hand posture classification was evaluated in different postures, including the postures in Fig. 6. Figure 12 show the results of the eight difference postures, and the hand depth and posture type are shown at the white number and the text. As can be seen in Fig. 12, the system can filter out the non-RoF (region of feature) than to classify the different postures into the same static hand posture. Static hand posture classification was tested by video sequences containing eight sets of posture. Here, we evaluate our performance by Frames, Time (sec), Frame Per Second (FPS) and Miss Rate (MR). The confusion matrix of static posture recognition is shown in Table 5. The left-most column in Table 5 means the gesture we performed and the top row means the gesture classified. The evaluation results are shown in Table 6, and the proposed system can classify eight kinds of static hand

Table 6 Static hand posture performance evaluation

posture	Frames	Time (sec)	FPS	MR (%)
Palm-open-up-left	940	32	29.38	3.94
Palm-open-down-left	760	26	29.23	1.71
Palm-open-left-down	420	14	30	4.52
Palm-open-right-up	570	19	30	5.44
Palm-open-up-right	800	27	29.63	0.75
Palm-open-down-right	750	25	30	1.07
Palm-open-left-up	620	21	29.52	3.55
Palm-open-right-down	700	23.5	29.78	1.14

postures. The miss rates of the posture classification are lower than 6 % and the average accuracy can achieve about 97 %.

4.2 Dynamic gestures

The dynamic gesture recognition was evaluated in different depths. Table 7 shows the results of the nine different gestures in Table 1. In all the figures, the white number represents the hand

Table 7 examples of the nine different gestures

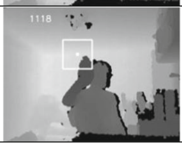
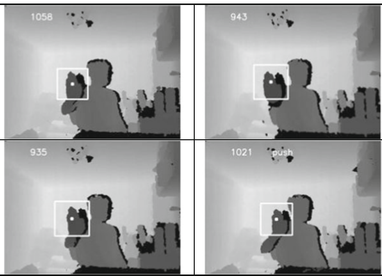
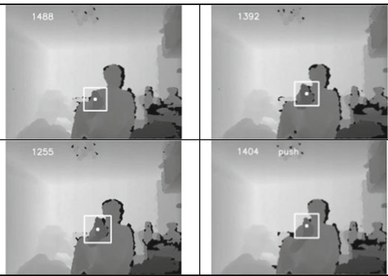
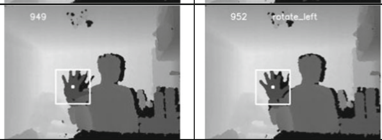
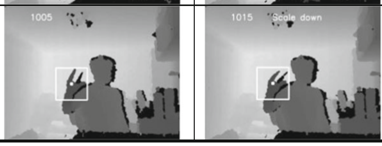
gesture	near		far	
swipe_up				
swipe_down				
swipe_left				
swipe_right				
wave				
				
circle				
				

Table 7 (continued)

gesture	near	far
push		
rotate left		
rotate right		
scale up		
scale down		

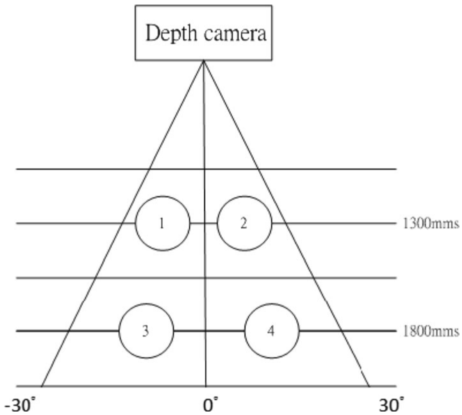


Fig. 13 Environment for dynamic gesture recognition experiment

Table 8 Dynamic gesture recognition performance evaluation

Gesture\user	Total	Average
swipe_up	138/160	0.86
swipe_down	135/160	0.84
swipe_left	140/160	0.88
swipe_right	142/160	0.89
wave	140/160	0.88
circle	147/160	0.92
push	148/160	0.93
rotate left	136/160	0.85
rotate right	136/160	0.85
scale up	136/160	0.85
scale down	135/160	0.84
Total	1533/1760	0.87

depth and the white text represents the detected gesture. As can be seen in Table 7, the system can recognize nine different gestures with different depths. The performance of dynamic gesture recognition is verified by two different persons at 4 different positions with 2 different depths, as shown in Fig. 13. At every position, user performs every dynamic gestures 20 times. When the person stays at 1300mms from the camera, the depth of hand is about 700 to 1100mms. When the person stays at 1800mms from the camera, the depth of hand is about 1200 to 1600mms. For gesture interface that allow users to easily control home appliance, we define the postures used for different dynamic gestures, as shown in Figs. 8, 10 and 11. To detect the rotate gesture, the system needs to detect at least three fingertips. If a user uses arbitrary hand posture to perform the dynamic gesture except the rotate gesture, the rotate gesture can possibly be detected. Accordingly, using postures for dynamic gestures can exclude the unintentional rotate gesture generated by wave gesture or other gestures except the rotate gesture. As can be seen in the aforesaid experimental result analyses, the proposed system can recognize nine different types of dynamic gesture at different depths. The average accuracy of dynamic gesture recognition can achieve about 87 %. The static hand posture based methods and fingertip based methods can have good recognition rate in the near range due to the fitting hand size. The hand trajectory based methods are not affected by the distance, so the average accuracy of circle and push gesture recognition can achieve about 92 %. The accuracy of the dynamic gesture recognition is summarize in Table 8, and the average computation time per frame is 16.32 ms.

5 Conclusions

In this article, we proposed a dynamic hand gesture recognition system using only the depth information. The proposed system can recognize nine commonly used dynamic hand gestures for embedded systems. These dynamic hand gestures can be recognized according to several methods, including static hand posture based methods, hand trajectory based methods and fingertip based methods. These gestures can be recognized at different positions with different depths. The proposed system can accurately detect the dynamic hand gestures with an average recognition rate of 87.6 %, which is good for controlling the embedded systems. In the future work, we would like to apply the proposed system to many applications such as home appliance, entertainment and medical systems, etc.

References

1. Barkoky A, Charkari N M (2011) Static Hand Gesture Recognition of Persian Sign Numbers using Thinning Method. International Conference on Multimedia Technology (ICMT)
2. Bradski G, Kaehler A (2008) Learning OpenCV: Computer Vision with the OpenCV Library,, vol. 1, 1st edn, O'reilly Media
3. Chang CC, Lin CJ (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2(3):pp. 27:1–27:27
4. Chen C-P, Chen Y-T, Lee P-H, Tsai Y-P, Lei S (2011) Real-time Hand Tracking on Depth Images. IEEE Visual Communications and Image Processing (VCIP)
5. Choi J, Park H, Park, J-I (2011) Hand Shape Recognition Using Distance Transform and Shape Decomposition. 18th IEEE International Conference on Image Processing (ICIP)
6. Douglas D, Peucker T (1973) Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Can Cartographer 10(2):112–122
7. Graham RL (1972) An efficient algorithm for determining the convex hull of a finite planar set. Inf Process Lett 1:132–133
8. Homma K, Takenaka E-I (1985) An image processing method for feature extraction of space-occupying lesions. J Nucl Med 26:1472–1477
9. Huang D-Y, Hu W-C, Chang S-H (2009) Vision-based Hand Gesture Recognition Using PCA+Gabor Filters and SVM. Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing
10. Kakumanu P, Makrogiannis S, Bourbakis N (2007) A survey of skin-color modeling and detection methods. Pattern Recogn 40(3):1106–1122
11. Keskin C, Kirac F, Kara Y E, Akarun L (2012) Randomized Decision Forests for Static and Dynamic Hand Shape Classification. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)
12. Kurakin A, Zhang Z, Liu Z (2012) A Real Time System for Dynamic Hand Gesture Recognition With A Depth Sensor. Proceedings of the 20th European Signal Processing Conference (EUSIPCO)
13. Li W, Zhang Z, Liu Z (2008) Expandable data-driven graphical modeling of human actions based on salient postures. IEEE Trans Circuits Syst Video Technol 18(11):1499–1510
14. Liu Y, Yin Y, Zhang S (2012) Hand Gesture Recognition Based on HU Moments in Interaction of Virtual Reality. 4th International Conference on Intelligent Human-Machine Systems and Cybernetics
15. Minnen D, Zafrulla Z (2011) Towards Robust Cross-User Hand Tracking and Shape Recognition. IEEE International Conference on Computer Vision Workshops (ICCV Workshops)
16. Oprisescu S, Rasche C, Su B (2012) Automatic Static Hand Gesture Recognition Using TOF Cameras. Proceedings of the 20th European Signal Processing Conference (EUSIPCO)
17. Prime Sense (2010) Prime sense Xtion's hardware. http://www.primesense.com. Accessed 22 March 2013
18. Prime Sense OpenNI User Guide. pp 26-29
19. Raheja J L, Chaudhary A, Singal K (2011) Tracking of Fingertips and Centres of Palm using KINECT. Third International Conference on Computational Intelligence, Modelling & Simulation
20. Ren Z, Yuan J, Zhang Z (2011) Robust Hand Gesture Recognition Based on Finger-Earth Mover's Distance with a Commodity Depth Camera. Proceedings of the 19th ACM international conference on Multimedia
21. Senanayake R, Kumarawadu S (2012) A Robust Vision-based Hand Gesture Recognition System for Appliance Control In Smart Homes. IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC)
22. Shrivastava R (2013) A Hidden Markov Model based Dynamic Hand Gesture Recognition System using OpenCV. IEEE 3rd International Advance Computing Conference (IACC)

23. Suryanarayan P, Subramanian A, Mandalapu D (2010) Dynamic Hand Pose Recognition using Depth Data, 20th International Conference on Pattern Recognition (ICPR)
24. Wachs JP, Kolsch M, Stern H, Edan Y (2011) Vision-based hand-gesture applications. Commun ACM 54(2): 60–71
25. Wu C-H, Lin C-H (2013) Depth-based hand gesture recogn ition for home appliance control. IEEE 17th International Symposium on Consumer Electronics (ISCE)
26. Yang Z, Li Y, Chen W, Zheng Y (2012) Dynamic Hand Gesture Recognition Using Hidden Markov Models. The 7th International Conference on Computer Science & Education (ICCSE 2012)
27. Zhu X, Wong KK (2012) Single-Frame Hand Gesture Recognition Using Color and Depth Kernel Descriptors. 21st International Conference on Pattern Recognition (ICPR)

Chih-Hung Wu received the B.S. degree in computer science and engineering from Tatung University, Taipei, Taiwan, and M.S. degree in electronic and computer engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 2013.

Wei-Lun Chen received the B.S. degree in computer science from Dong Hua University, Taipei, Taiwan, and is working on his M.S. degree in electronic and computer engineering from National Taiwan University of Science and Technology.

Chang Hong Lin received the B.S. and M.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, and the M.A. and Ph.D degree in electrical engineering from Princeton University, Princeton, NJ, in 1997, 1999, 2003, and 2007, respectively. His research interests include ubiquitous camera framework, code compression for embedded system, and hardware/ software co-synthesis. He is currently an associate professor at the department of electronic and computer engineering in the National Taiwan University of Science and Technology. He is the corresponding author of this article.