Engineering Applications of Artificial Intelligence 49 (2016) 103–113

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A combination of user-guide scheme and kernel descriptor on RGB-D data for robust and realtime hand posture recognition

Huong-Giang Doan [a,b,*], Van-Toi Nguyen [a,c], Hai Vu [a], Thanh-Hai Tran [a]

[a] International Research Institute MICA HUST, CNRS/UMI-2954, INP Grenoble, Vietnam
[b] Industrial Vocational College Hanoi, Vietnam
[c] The University of Information and Communication Technology, Thai Nguyen University, Vietnam

ARTICLE INFO

Article history:
Received 22 May 2015
Received in revised form
18 September 2015
Accepted 17 November 2015
Available online 24 December 2015

Keywords:
Background subtractions
Human computer interaction
User-guide learning scheme
Hand gesture recognition
Color skin model

ABSTRACT

This paper presents a robust and real-time hand posture recognition system. To obtain this, key elements of the proposed system contain an user-guide scheme and a kernel-based hand posture representation. We firstly describe a three-stage scheme to train an end-user. This scheme aims to adapt environmental conditions (e.g., background images, distance from device to hand/human body) as well as to learn appearance-based features such as hand-skin color. Thanks to the proposed user-guide scheme, we could precisely estimate heuristic parameters which play an important role for detecting and segmenting hand regions. Based on the segmented hand regions, we utilize a kernel-based hand representation in which three levels of feature are extracted. Whereas pixel-level and patch-level are conventional extractions, we construct image-level which presents a hand pyramid structure. These representations contribute to a Multi-class support vector machine classifier. We evaluate the proposed system in term of the learning time versus the robustness and real time performances. Averagely, the proposed system requires 14 s in advanced to guide an end-user. However, the hand posture recognition rate obtains 91.2% accuracy. Performance of the proposed system is comparable with state-of-the-art methods (e.g. Pisharady et al., 2012) but it is a real time system. To recognize a posture, its computational cost is only 0.15 s. This is significantly faster than works in Pisharady et al. (2012), which required approximately 2 min. The proposed methods therefore are feasible to embed into smart devices, particularly, consumer electronics in domain of home-automation such as televisions, game consoles, or lighting systems.

1. Introduction

Communications between human and computer (or recently, smart devices) become more and more natural and intuitive interactions. Human–Computer Interaction (HCI) therefore has been a wide field of research in the last thirty years, which has led to uncountable algorithms and methods (Zabulis et al., 2009; Rautaray and Agrawal, 2012). However, utilizing hand postures as a natural interaction is still a very challenging problem due to the complexity of hand shapes, and high computational costs of the vision algorithms. On one hand, robustness of the relevant systems are suffered from various hand postures under different lighting conditions and cluttered backgrounds. On the other hand, in order to ensure the natural and intuitive feedbacks, the vision-based hand posture recognition algorithms should process video stream in real time. As a result, many current approaches are either too limited speed, accuracy and robustness or demand too many prerequisites such as markers, clean backgrounds and so on. Different from above concerned approaches, this paper tackles a trade-off between robustness and real-time performance with a cost of the user-dependence. This cost pays for training end-users so that such critical performances are achievable. It is noticed that while most of the hand posture recognition techniques aim at an user-independence system. In other words, they can adapt different users rather than a specific user. In this paper, we show that a hand posture recognition system is able to be an user-dependence one with a reasonable user-guide scheme.

We take into account designing an efficient user-guide scheme so that it requires minimal training time and user intentions. The proposed user-guide scheme consists of three stages. They are to answer questions: (1) how is the current background scene? (2) How are appearances (e.g., color) of the hand's skin? (3) How far from user's hand to the sensor? Thanks to this scheme, the

* Corresponding author: Tel.: +84 38 68 30 87.
E-mail addresses: huong-giang.doan@mica.edu.vn (H.-G. Doan), van-toi.nguyen@mica.edu.vn (V.-T. Nguyen), hai.vu@mica.edu.vn (H. Vu), thanh-hai.tran@mica.edu.vn (T.-H. Tran).

http://dx.doi.org/10.1016/j.engappai.2015.11.010
0952-1976/

heuristic parameters in the hand detection and segmentation procedures are adaptively selected. For instance, a depth image from Kinect sensor gives not only distance from hand to Kinect, but the distance to other objects in a certain scene. By using the result of (3), the hand-distance parameters can be learnt quickly. The candidates of the hand region therefore can be separated in the depth image using such parameters. Similarly, we prune the hand candidates through skin colors learnt in (2). Consequently, we obtain a high accuracy of the segmented hand on a complicated background.

For the recognition task, different from robust existing hand posture recognition algorithms, e.g., Pisharady et al., 2012, we pay special attentions of representing structure of the hand regions. The representations of a posture consist of three levels: pixel-level, patch-level, and hand-level. This goal is available because the hand region is separated on a complicated background. Such representations fully describe the composite property of hand postures. Whereas the relevant works manage characteristics of the hand postures based on a stack of various features such as saliency, texture, shape, colors (e.g., a Bayesian-nets in Pisharady et al., 2012; Zabulis et al., 2009). In this study, we deploy a multi-class SVM classifier which utilizes compactness of the proposed hand representations. The proposed method achieves a very promising recognition rate and a real time processing in experimental evaluations. In particular, the recognition rate is significantly increased when hand regions are separated from the complicated background. Cost of the proposed system is that the user-guide scheme must be activated whenever an end-user initiates controlling a device. It consumes averagely 15 s in advance for each end-user. This is an acceptable training cost for end-user. Consequently, the proposed system is feasible to deploy practical applications.

The rest of paper is organized as follows: Section 2 briefly surveys related works. Section 3 describes the proposed user-guide scheme. The experimental results are shown in Section 4. Finally, Section 5 concludes and suggests further research directions.

2. Related work

Hand posture recognition has been mentioned since thirty years ago. Nowadays, it remains an active topic in fields of the computer vision because of its wide range of practical applications, more specifically, sign hand language, lie detection, game, e-learning, and human–robot interaction. Research on vision based hand posture recognition initially employed video sequences provided by conventional RGB camera. With the development of new and low-cost depth sensors, new opportunities for posture recognition have emerged. Microsoft Xbox-Kinect is a successful commercial product that provides both RGB and depth data for recognizing hand postures/gestures to control game consoles (http://www.microsoft.com/en-us/kinectforwindows). Many technology companies launch smart-devices using the Kinect (http://www.microsoft.com/en-us/kinectforwindows) or like-Kinect sensors (e.g., Xtion PRO LIVE – ASUS, softKinect). For instance, Samsung smart-TV manipulates TV-functions using dynamic hand gestures. Omron introduces the smart-TV integrated facial and hand recognition. PointGrab proposed an unique solution based on shape and motion recognition including gesture, face, and user behavior analytic (Company, 2013). From view point of the specific applications of smart homes which utilize computer vision based techniques, readers can refer survey (De Silvaa et al., 2012). Increasingly, in-air gesture recognition is being incorporated into consumer electronics and mobile devices like WiSee system (Qifan et al., 2013). WiSee focuses on gesture recognition and shows how to extract gesture information from wireless transmissions.

In the literature, there are uncountable solutions for a vision-based hand posture recognition system. Readers can refer good surveys such as Zabulis et al. (2009), Rautaray and Agrawal (2012), and Erol et al. (2007), for technical details. Roughly speaking, the vision-based hand posture recognition techniques consist of two main phases: hand detection and posture recognition (Zabulis et al., 2009; Rautaray and Agrawal, 2012; Erol et al., 2007). In order to achieve high performance of the hand detection, the relevant works often customize or combine multiple features such as color, edge, shape, motion of hand (e.g., Mittal et al., 2011; Chen et al., 2003). However, such combination schemes consume a high computational time, e.g., Mittal et al. (2011) require 2 min to detect hands in a photo. To speed up the detection as well as to avoid illumination and skin color changing, Ren et al. (2011, 2013) used depth data captured from Kinect sensor with two main assumptions: the user needs to make sure that the hand is the front-most object facing the sensor; the user needs to wear a black belt on the gesturing hand's wrist. They detect hands by thresholding the depth map to determine hand regions then apply RANSAC to locate the position of black belt in order to precisely localize and segment the hand region. Following the same idea, Yang (2015) proposed to detect hand from human skeleton. They then refine the hand region by thresholding the depth data and black wristband detection. Obviously, the current approaches favor the use of depth data and only use color as complementary.

To achieve an acceptable recognition rate, classifiers or statistical models of hand postures always require sophisticated algorithms. For instance, to build a Bayesian network for recognizing hand postures, Pisharady et al. (2012) construct multiple layers of the texture, color, shape of the hand; or in order to extract precisely hand contours, Zabulis et al. (2009) need four layers processing. Obviously, addressing such issues always needs to compute intensively and thus makes current hand posture recognition systems fragile and unstable. Recent works such as Ionescu et al. (2007) and Bergh et al. (2009) utilize depth information to correctly extract hand regions. Bergh and Van-Gool (2011) extracted skeleton of hands and applied Hidden Markov Models for classifying various hand gestures. Park et al. (2012) extracted blob of hand for tracking using both Time-of-Flight and RGB camera. Doliotis et al. (2012) introduced a method for the 3D hand pose and hand configuration. Ren et al. (2013) proposed to represent hand shape by normalized time-series. Then, template matching technique was used for the recognition procedures. In such works, a large database of hand is prepared in advanced.

As mentioned above, our study is motivated by a trade-off between user-dependence and critical performances of the system. There are two main points different from our study and relevant works:

- *User-dependence*: Some research on vision-based hand postures/gestures recognition or tracking need the help of markers or colored gloves (Joslin et al., 2005; Keskin et al., 2003; Lamberti and Camastra, 2011). A full list of the glove-based applications could be referred in Dipietro et al. (2008). However, in this study, without any makers has been used. We design an user-guide scheme so that the proposed system can learn current contexts and environmental conditions such as how far from an end-user to device; or how is the current backgrounds. The proposed scheme is also different from learning phases which appear in recognition algorithms. Such learning approaches are to design intelligent machines so that the target systems can adapt various goals such as detecting car (Nguyen and Nguyen, 2008), recognizing human, or various type of objects (Viola and Jones, 2001; Park and Choi, 1996). In contrast, our scheme trains

an end-user in a passive way. Such training scheme tends to be simple enough for end-user, low costs, and minimal user-intentions. To save training time, further extension such as storing profile of each end-user is available to deploy.

- *The hand posture recognition algorithms*: We have proposed a novel method based on kernel descriptor for hand posture recognition from RGB data. The original idea comes from the works presented in Bo and Sminchisescu (2009). However, we improved kernel descriptors to be more robust to scaling, rotation, and hand structures. In this paper, we evaluate the proposed method on the segmented hand regions. It confirms our observations that by separating correctly hand regions from a complicated background, a classifier based on kernel descriptors is able to increase significantly accuracy of the recognition rate.

3. Proposed approach

3.1. Assumptions and proposed framework

3.1.1. Assumptions

Before describing the proposed approach, we present main assumptions on that our work replies:

- To control smart device, the Kinect sensor is used to capture both depth and color information for hand gesture recognition. The Kinect sensor is mounted on a fixed rack or a tripod. An end-user stands in a valid range of depth feature (e.g., 0.5–4 m for the Kinect sensor (http://www.microsoft.com/en-us/kinectforwindows)).
- An end-user controls devices by raising his/her hand in front of the body and he/she stands at a fixed position during controlling device.

The first assumption is reasonable in real practical application because position of a device such as a television or a game console usually is fixed. The second assumption is acceptable because of habits of end-users when they control a device.

3.1.2. Proposed framework

The proposed frameworks consists of two phases: (i) training user; (ii) hand detection and recognition. Both phases are realized at runtime, one after the other. Training phase aims to learn parameters (background model, distance from user hand to Kinect, skin color) served for hand detection and recognition.

By using a fixed Kinect sensor (http://www.microsoft.com/en-us/kinectforwindows), a RGB image I and a depth data D are concurrently collected. The proposed flow-work for recognizing hand postures from a couple of images (I,D) consists of a series of the cascaded steps, as shown in Fig. 1(a). Main steps contain detection, segmentation and recognition. They are briefly explained below:

- *Pre-processing*: Because I and D images are taken by a Kinect sensor that are not measured by the same coordinates. A pre-processing is required in order to calibrate them. In this work, we utilize calibration method proposed by Herrera et al. (2012).
- *Body detection*: As Kinect sensor and environment are fixed, human body can be detected using background subtraction technique. To avoid illumination changes, we use depth instead of RGB data. Body regions B_d then are extracted as follows:

$$B_d = D|_{Diff(D,BG) > Thresh_{body}} \quad (1)$$

where *Diff* is subtraction operator, *BG* is background model. Learning parameters of *BG* is presented in Section 3.2.

- *Detecting hand candidates*: Replying on the assumption that users usually toward their hand near camera than body, hand candidates H_d could be detected from body regions B_d using distance based thresholding technique.

$$H_d = B_d|_{B_d < Thresh_{hand}} \quad (2)$$

Determination of $Thresh_{hand}$ value is explained in Section 3.3.

- *Pruning hand candidates*: Hand candidates H_d often consist of contaminated regions. We then apply skin color constraint to deal with this issue. Among many candidates of H_d, we keep only ones satisfying skin distribution.

$$H^* = I|_{H_d,\Omega_c} \quad (3)$$

Learning parameters of skin color model Ω_c is presented in Section 3.4. Then we do pruning H^* to obtain fully the hand region.

- *Recognizing hand postures*: The pruning region of H^* is input of the hand representation into a classifier algorithm. Detail is presented in Section 3.6.

The relation between an input image $(I, D), B_d, H_d, H^*$ and a set of skin-color pixels S is illustrated by Venn diagram, as in Fig. 2.

Intuitively, above flow-work is a common solution (e.g., similar to a previous work (Park et al., 2012)). The major advantages are that it requires lowest computational time due to utilizing simple algorithms for detecting and segmenting the hand regions. However, this flow-work also requires many heuristic parameters such as background model, distance from human to Kinect, skin color model parameters. These parameters are usually pre-determined in the common approaches (Park et al., 2012). Contrary that work, in this study, we propose a three-stage learning scheme to adaptively select these heuristic parameters for each end-user. Fig. 1 (b) shows the associated stages to select optimal corresponding parameters in Eqs. (1)–(3).

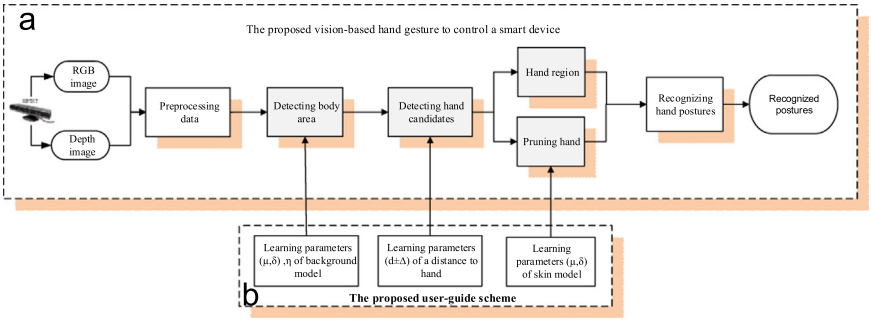

Fig. 1. The proposed framework for hand postures recognition using depth and color information.

3.2. Estimating parameters of background model for body detection

Because sensor and environment are fixed, we could detect human region using background subtraction techniques. Both depth and RGB data can be used for the background subtraction. However, we use depth data because it is less sensitive to illumination changes. Among numerous techniques of the background subtractions, we adopt Gaussian Mixture Model (GMM) (Stauffer and Grimson, 1999) because this technique has been shown to be the best suitable for our system.

Because of stability of the depth feature, our background model contains only one Gaussian. Given a depth sequence including n frames, an observation of a pixel p along temporal dimension is denoted by $s_p = [D_{1,p}, D_{2,p}, \ldots, D_{n,p}]$ with a standard deviation $\sigma_p = std(s_p)$. The background model of pixel p is represented by $BG_p = (\mu_p, \eta_p, \sigma_p)$ and computed as follows:

- Noise model η_p:

$$\eta_p = \begin{cases} 0 & \text{if } \sigma_p < \tau \\ 255 & \text{otherwise} \end{cases} \tag{4}$$

- Mean value μ_p:

$$\mu_p = \begin{cases} \dfrac{\sum_{t=1}^{n} D_{t,p}}{n} & \text{if } \sigma_p < \tau \\ \dfrac{\sum_{t=1}^{k} D_{t,p} \,|\, D_{t,p} < i.d.v}{k} & \text{otherwise} \end{cases} \tag{5}$$

where k is the number of observations from s_p having depth value smaller than a $i.d.v$ (invalid depth value – or a white pixel on depth data)

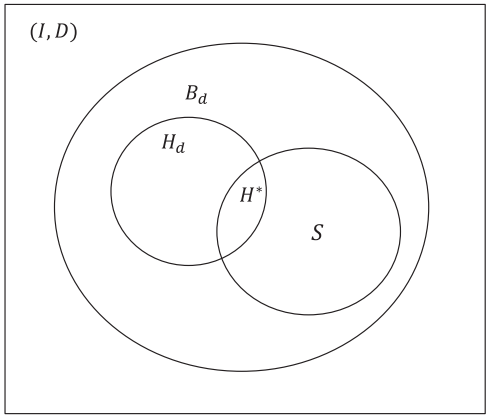

Fig. 2. The Venn diagram representing the relationship between the pixel sets (I, D), B_d, H_d, S and H^*.

According to (5) and (4), the parameter τ aims to evaluate how a signal s_p is stable. The parameters $(\mu_p, \eta_p, \sigma_p)$ are learnt for every pixel of a depth frame. Utilizing results of the background model μ_p and η_p, we firstly do denoising procedure on depth image that is captured from the Kinect sensor. At each pixel p, the denoised depth value D'_p is defined as:

$$D'_p = \begin{cases} \mu_p & \text{if } (\eta_p = 255) \;\&\; (D_p \text{ is } i.d.v) \\ D_p & \text{otherwise} \end{cases} \tag{6}$$

The body region is then segmented from D'_p. A pixel p belongs to a body region when the condition (7) below is ensured:

$$B_{d,p} = D'_p \,|\, _{D'_p - \mu_p > 0} \tag{7}$$

Fig. 3(a–c) shows results of the background subtraction using (7). Given a region of human body (as shown in Fig. 3(c)), we continuously extract candidates of the hand.

3.3. Estimating the distance from hand to the Kinect sensor for extracting hand candidates

As denoted in the second assumption, hand of end-user is raised in front of the body during the controlling. Candidates of the hand can be extracted from the body regions B_d by evaluating the distances features, as shown in (2). $Thresh_{hand}$ deciding hand regions from B_d is learnt by following procedure.

Firstly, we build a histogram of depth data of the detected body regions B_d. Intuitively, there are two local peaks in this histogram. One peak covers a range from the Kinect sensor to hand, and another peak represents depth data from the Kinect sensor to other body parts. Two peaks are separated by $Thresh_{hand}$ which an end-user is asked to move his/her hand in few times. This trick elicits hand regions in consecutive frames because area of the hand would have to be strongly fluctuated. The moving parts are calculated using differences between consecutive depth frames D_{t-2}, D_{t-1}, D_t by:

$$\begin{cases} D_{t-2,t-1} = D_{t-1} - D_{t-2} \\ D_{t-1,t} = D_t - D_{t-1} \\ D_{hand} = D_{t,t-1} \cap D_{t-2,t-1} \end{cases} \tag{8}$$

Fig. 4 illustrates how to detect the moving parts from three consecutive frames with fixed position of an end-user. Fig. 4(d) shows differences of depth between frames Fig. 4(a) and Fig. 4(b), whereas moving parts between Fig. 4(b) and (c) are shown in Fig. 4(e). An intersection operator and following by a binary threshold one give hand regions, as shown in Fig. 4(f).

Next, given hand regions H_d, we compute depth histogram of the H_d. Fig. 4(g) shows a depth histogram of the hand regions in left panel. Obviously, the parameter $Thresh_{hand}$ is identified. By using $Thresh_{hand}$, an intersection histogram operator is applied to B_d, D to eliminate body parts using (2).

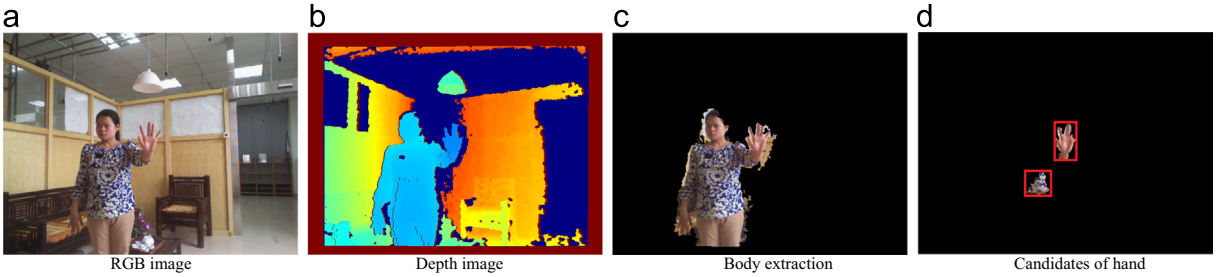

Fig. 3. Results of hand region detection.

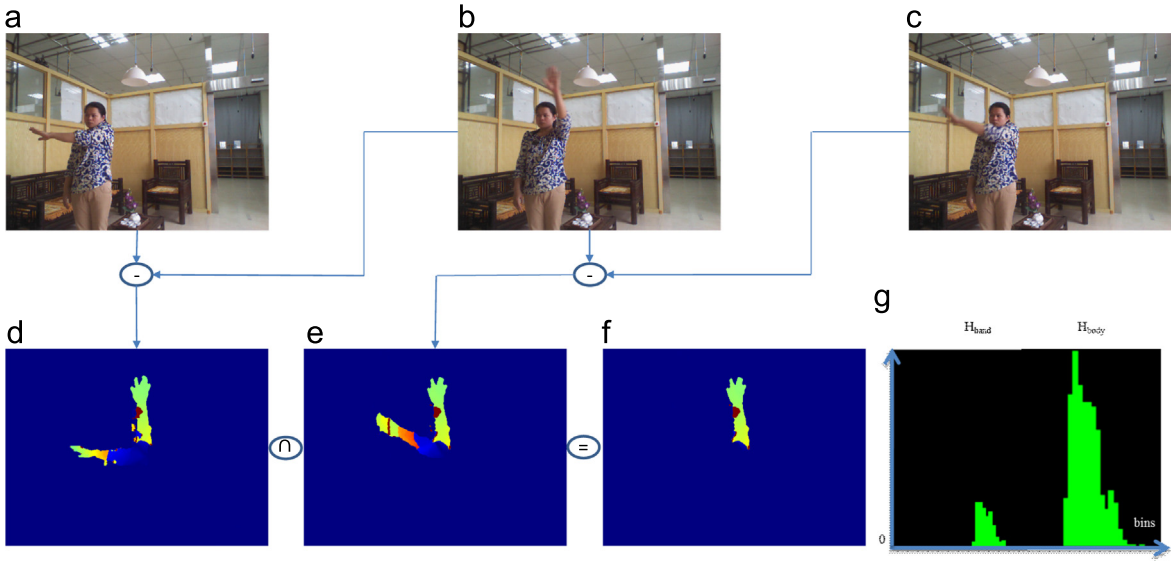

Fig. 4. Result of the learning distance parameter. (a–c) Three consecutive frames. (d) Results of subtracting two first frames. (e) Results of the subtracting two next frames. (f) Binary thresholding operator. (g) A range of hand (left) and of body (right) on the depth histogram.

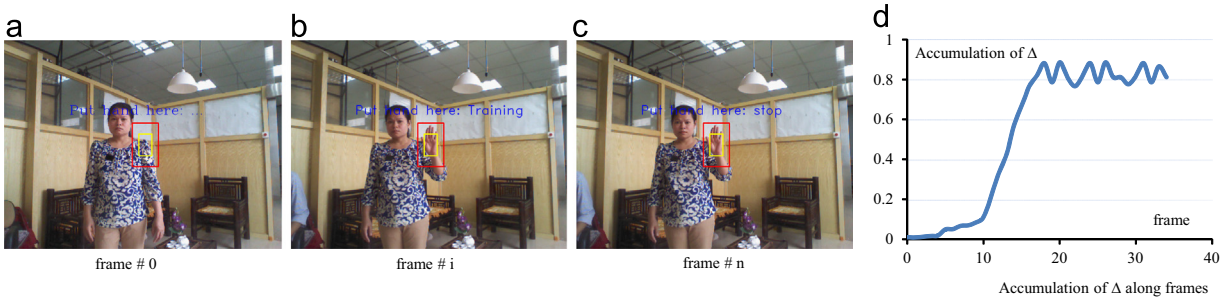

Fig. 5. Result of the training skin color model. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

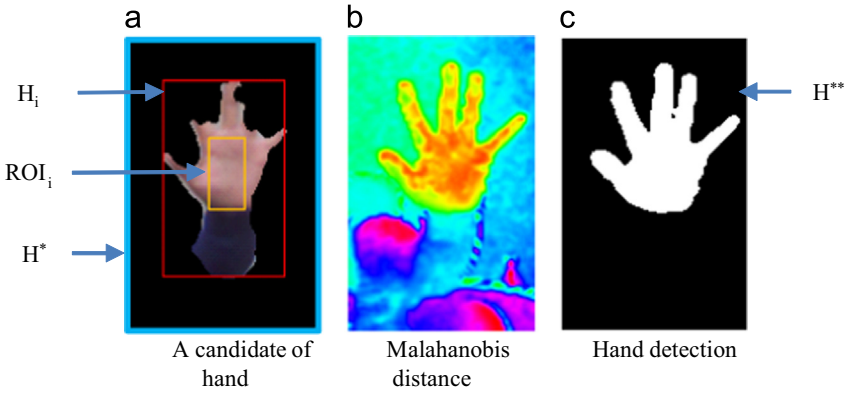

Fig. 6. Results of the hand segmentation. (a) A Candidate of hand. (b) Mahalanobis distance. (c) Refining the segmentation results using RGB features. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

3.4. *Estimating skin color parameters for pruning hand regions*

The hand detection results as shown in Fig. 3(d) often consist of contaminated regions. They come from objects having the same distance as hand to Kinect sensor. Moreover, because depth data has low resolution, the extracted hand regions do not precisely cover the whole area of hand, particularly at boundary of fingers. We propose to use the color distribution of hand skin to handle these issues.

There are many approaches that utilize color features to segment hand regions (e.g., Jones and Rehg, 1999; Pisharady et al., 2012). Our works are inspired from Jones and Rehg (1999) for learning a model of hand skin color. According to Jones and Rehg (1999), a skin model is characterized by $\Omega_c = (\mu_{skin}, \delta_{skin})$ where μ_{skin} and δ_{skin} are mean and covariance of color vector $[R, G, B]$. In order to estimate $(\mu_{skin}, \delta_{skin})$, we design a learning procedure as below. The main idea is to determine a hand region where skin color must not be confused with other colors.

Firstly, the end-user is required to raise his/her hand so that the hand is located approximately in a pre-determined rectangle box (red box as seen in Fig. 5(a)–(c)). Then the center region (marked in yellow box in Fig. 5(a)–(c)), considered as containing only hand skin colors, will be extracted to compute skin model parameters.

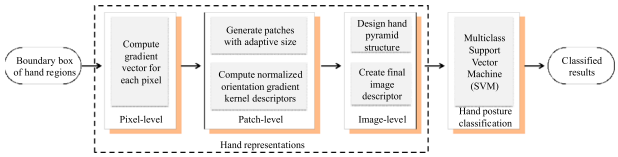

Fig. 7. The framework of the hand postures recognition.

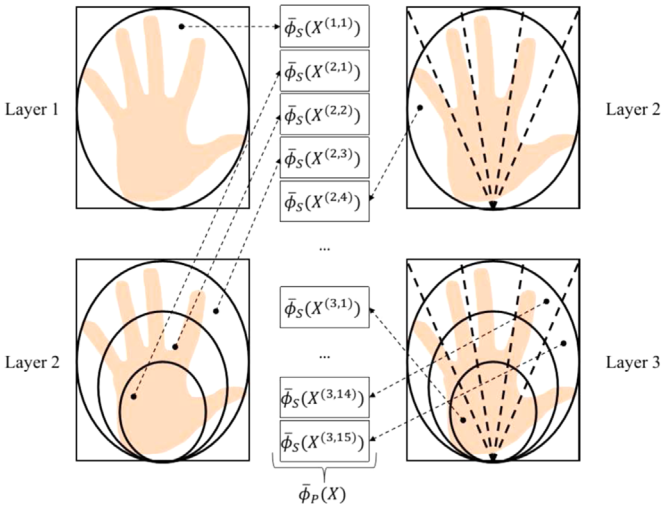

Fig. 8. Construction of image-level feature concatenating feature vectors of cells in layers of hand pyramid structure.

Given the extracted region, feature vectors $[R, G, B]$ of all pixels are transformed to HSV color space. We then apply a skin map as defined in Pisharady et al. (2012):

$$S_{skin(t)} = 1 - \sqrt{(H_t - H_{S0})^2 + \left(\frac{H_{Smax} - H_{Smin}}{S_{Smax} - S_{Smin}}\right)^2 (S_t - S_{S0})^2} \tag{9}$$

where H_t and S_t are values of Hue and Saturation channel at frame t, respectively. Other parameters are set by default as in Pisharady et al. (2012) ($H_{S0} = 0.1073, S_{S0} = 0.3515, H_{Smax} = 0.1892, H_{Smin} = 0.0122, S_{Smax} = 0.6250, S_{Smin} = 0.0558$).

As mentioned previously, in order to characterize as precise as possible skin model, the extracted region must contains only skin color. To this end, we ask the user to keep the hand at a fix position with open palm posture during a certain time. We then measure the stability of skin distribution and stop the training when no change appears.

The stability of skin distribution is measured by normal cross correlation of two histograms computed on the extracted region at two contiguous frames:

$$\epsilon_t = \frac{\sum_i (S_{t-1}(i) - \overline{S}_{t-1})(S_t(i) - \overline{S}_t)}{\sqrt{\sum_i (S_{t-1}(i) - \overline{S}_{t-1})^2 \times \sum_i (S_t(i) - \overline{S}_t)^2}} \tag{10}$$

Let $\Delta_t = |\epsilon_t - \epsilon_{t-1}|$. Fig. 5(d) shows variation of difference accumulation $\frac{1}{N}\sum_{t=1}^{N} \Delta_t$ from frame 1 to N. We could see that at the beginning, this value increases significantly corresponding to the strong movement of hand raising to the pre-defined position. Until a certain time, this value oscillates around a point. That means the distribution of hand skin becomes stable and we could stop the training.

Fig. 5 (d) shows that the training procedure is converged after 35 frames. The samples pixels of the center regions from *frame*#1

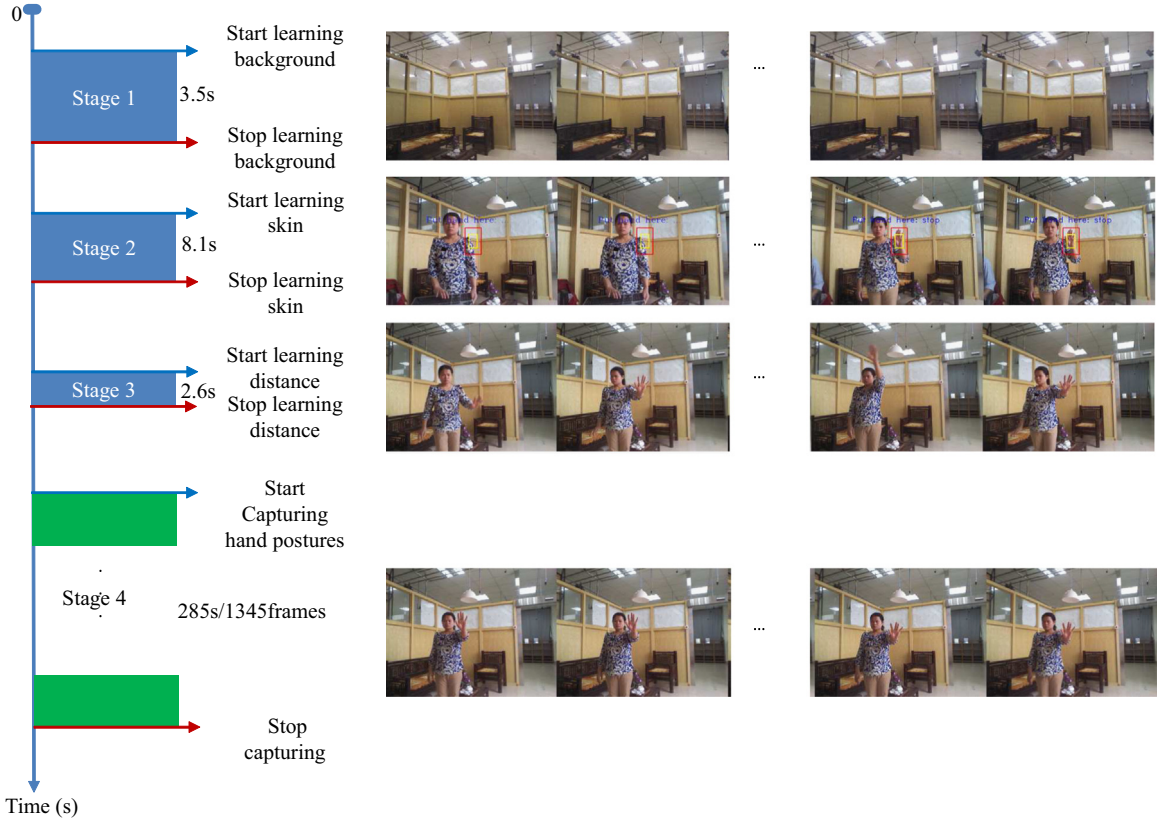

Fig. 9. Log activities of an evaluator who follows stages of the user-guide scheme and represents seven hand postures for preparing the posture dataset.

Table 1
The required time to learning parameters of the background model.

Subject ID	1	2	3	4	5	6	7	8	9	10
Time (s)	3.701	3.594	3.526	3.605	3.448	3.590	3.460	3.508	3.417	3.626
Avg. ± std	3.547 ± 0.086 s									

Table 2
The required time to learn parameters of the hand-skin color model.

Subject ID	1	2	3	4	5	6	7	8	9	10
∑ *Frames*	36	24	21	23	31	28	25	20	24	30
Time (s)	11.09	7.39	6.47	7.09	9.55	8.63	7.7	6.16	7.39	9.24
Avg. ± std	8.07 ± 1.54 s									

Table 3
The required time to learn the hand to Kinect distance.

Subject ID	1	2	3	4	5	6	7	8	9	10
∑ *Frames*	32	37	35	40	32	38	36	28	34	30
Time (s)	2.4	2.8	2.7	3.0	2.5	2.9	2.7	2.1	2.6	2.3
Avg. ± std	2.6 ± 0.27 s									

to *frame#N* are collected. Parameters $(\mu_{skin}, \delta_{skin})$ are calculated from such sample data.

3.5. Hand detection

Once all needed parameters have been determined, we could move to the second phase of hand detection and posture recognition for controlling smart devices. At this time, an user starts to raise his/her hand and performs a hand posture. The system will firstly detect human body using background subtraction with background model that has been learnt previously. Then hand candidates are extracted using distance threshold. As mentioned above, these regions often are contaminated (see Fig. 6(a)). We then apply skin color constraint to remove false alarms and pruning hand regions for a better segmentation.

Given a candidate of hand, we determine its bounding box H_i (as marked in red in Fig. 6(a)) and take a ROI_i (Region of Interest) at center of H_i (as marked in yellow in Fig. 6(a)) that is taken by a certain scale $\delta > 1$: $ROI_i = H_i/\delta$. We utilize the skin color model that we have learnt to verify if that it is true hand region or not. A Mahalanobis distance on [R,G,B] data between ROI_i region and skin model $(\mu_{skin}, \delta_{skin})$ is calculated. If the number of pixels whose Mahalanobis distances are large enough in comparison with total number of pixels in the ROI_i, the candidate is decided as a true detection.

For pruning the segmentation results, we create a larger region than original H_i by $H^* = H_i \times \delta$ (as marked in the light-blue box in Fig. 6(a)) on RGB image. Then a Mahalanobis distance between H^* and skin model is calculated in order to extract fully hand skin pixels Fig. 6(b). Obviously, hand pixels were updated from original true hand shown in Fig. 6(a). Efficiency of this procedure is shown in Fig. 6(c), in which a true hand H^{**} region is extracted.

3.6. Hand posture recognition

We utilize a hand representation based on kernels to recognize hand postures. The detail of this method can be found in Nguyen et al. (2015). However, to make this paper more self-contained and to assure its continuity, we will briefly review this method as follows. The framework of the hand posture recognition using kernels is presented in Fig. 7. The framework consists of two main steps: Hand representation and Hand posture classification.

Hand representation composes of three sub-steps that are Pixel-level feature extraction, Patch-level feature extraction, and Image-level feature extraction.

- *Pixel-level feature extraction*: At this level, a normalized gradient vector is computed for each pixel of the image. The normalized gradient vector at a pixel p is defined by its magnitude $m(p)$ and normalized orientation $\omega(p) = \theta(p) - \overline{\theta}(P)$, where $\theta(p)$ is orientation of gradient vector at the pixel p, and $\overline{\theta}(P)$ is the dominant orientation of the patch P that is the vector sum of all the gradient vectors in the patch. This normalization will make patch-level features invariant to rotation. In practice, the normalized orientation of a gradient vector will be:

$$\tilde{\omega}(p) = [\sin(\omega(p)) \cos(\omega(p))] \tag{11}$$

- *Patch-level feature extraction*: Firstly, a set of patches is generated with adaptive size. The size of the patches is directly proportional to image size. On one hand, this ensures the number of patches to be considered unchanged. On the other hand, it makes the patch descriptor more robust to scale change. For each patch, we compute patch-level features based on a given definition of match kernel. The gradient match kernel is constructed from three kernels that are gradient magnitude kernel $k_{\tilde{m}}$, orientation kernel k_o and position kernel k_p.

$$K_{gradient}(P,Q) = \sum_{p \in P} \sum_{p' \in Q} k_{\tilde{m}}(p,p') k_o(\tilde{\omega}(p), \tilde{\omega}(p')) k_p(p,p') \tag{12}$$

where P and Q are patches of two different images that we need to measure the similarity. p and p' denote the 2D position of a pixel in the image patch P and Q respectively. Let $\varphi_o(.)$ and $\varphi_p(.)$ the feature maps for the gradient orientation kernel k_o and position kernel k_p respectively. Then, the approximate feature over image patch P is constructed as:

$$\overline{F}_{gradient}(P) = \sum_{p \in P} \tilde{m}(p) \phi_o(\tilde{\omega}(p)) \otimes \phi_p(p) \tag{13}$$

where $\otimes$ is the Kronecker product, $\phi_o(\tilde{\omega}(p))$ and $\phi_p(p)$ are approximate feature maps for the kernel k_o and k_p, respectively. The approximate feature maps are computed based on a basic method of kernel descriptor. The basic idea of representation based on kernel methods is to compute the approximate explicit feature map for kernel match function (Maji et al., 2013; Vedaldi and Zisserman, 2012; Bo and Sminchisescu, 2009; Bo et al., 2010).
- *Image-level feature extraction*: At this step, a pyramid structure *specific to hand postures* is used to combine patch features. This specific pyramid structure makes the descriptor more suitable for hand representation. We can see the proposed hand pyramid structure in Fig. 8. Based on the structure of the hand, the ellipses and the lines are used to divide the hand region into parts that contain different components of the hand such as palm and fingers. We remark that the regions at images corners often do not contain hands. For this reason, we only consider the area inside the inscribed ellipse of the hand image rectangle bounding box. The hand pyramid structure has 3 layers.

Given an image, the final representation is built based on

Fig. 10. Seven type of the postures recognized in the proposed system. (a) The first row: original images with results of the hand detections (in red boxes). (b) The second row: zoom-in version of the hand regions without segmentation. (c) The third row: the corresponding segmented hand.

Table 4
The required time to hand segmentation.

Subject ID	1	2	3	4	5	6	7	8	9	10
$\sum$ *Frames*	1132	1154	1496	1462	1288	1543	1427	1402	1295	1254
1/fps (ms)	256	286	368	394	299	274	284	365	324	241
Avg. $\pm$ std	285.45 $\pm$ 48.97 m									

features extracted from lower levels using efficient match kernels (EMK) proposed in Bo et al. (2010). First, the feature vector for each cell of the hand pyramid structure is computed. The final descriptor is the concatenation of feature vectors of all cells (see Fig. 8).

Hand posture recognition: We use Multi-class SVM classifier with the input is hand descriptor vector computed in the previous step. We observe that the accuracy of the classifier is strongly affected by background in the hand image. For this reason, we propose to use the segmented hand regions to improve the performance of hand posture recognition.

4. Experimental results

We evaluate the proposed method in term of the learning time versus the robustness and real time performances. The proposed framework is warped in a C++ program on a PC Core i5 3.10 GHz CPU, 4GB RAM. A MS Kinect sensor (http://www.microsoft.com/en-us/kinectforwindows) is mounted on a tripod at fixed position. The Kinect sensor captures data at 20 fps. We setup the evaluations in two different scenarios. The main purposes are to show effectiveness of the proposed method in different lighting conditions and/or appearances of complex backgrounds. Ten volunteers (5 males, 5 females) are asked to implement the evaluations. We obtained a hand posture dataset of 10 subjects including equal number postures for each environment. The subjects are asked to shot their postures in either standing or siting positions in order to incorporate the natural variations which may appear during the learning stages. The hand posture dataset is available at http://www.mica.edu.vn/perso/Doan-Thi-Huong-Giang/MICA_HandSet. Activities of the participants are recorded in log-files. They are analyzed to measure computational time costs of the proposed user-guide scheme, and to evaluate accuracy of the hand segmentation, as well as the hand posture recognition algorithms.

Fig. 9 shows a full evaluation scenario of an end-user who interacts with the proposed system. In this example, the experimental room is a smart-room with consumer electronic devices such as Television, air-condition, and a smart-lighting system. The end-user will follow a series guides on the PC's screen. Firstly, without appearances of the end-user in the experimental room, the system is activated to learn parameters of the background model, as described in the Stage 1. Then the end-user entrances the experimental room, stands in front of the device, and follows guides in the Stage 2 to learning skin model. Similarly, in Stage 3, the end-user is asked to raise her hand to learn distances from the hand to Kinect. Besides three stages of the proposed user-guide scheme, fourth stage captures seven postures of the end-user in order to evaluate hand posture recognition algorithms.

4.1. The required learning time for end-users

We calculate the required learning time for each stage. Details are reported below.

- *Stage* 1: To learn parameters of the background model. It is the same computational time for all evaluators because this procedure runs once only. However, we still implement this procedure for all evaluators to calculate an average time. Table 1 shows the computational time of this stage.
- *Stage* 2: The most consuming time is to learn parameters of the skin color model. We calculate a duration of each evaluator that

Table 5
The required time to hand posture recognition.

Subject	1	2	3	4	5	6	7	8	9	10
$\sum$ *Frames*	289	273	287	277	279	335	310	314	327	318
Time (fps)	0.15	0.141	0.153	0.139	0.145	0.174	0.161	0.142	0.14	0.149
Avg. $\pm$ std	0.149 ± 0.01 fps									

Table 6
Results of the JI indexes without/with learning scheme.

Subject	1	2	3	4	5	6	7	8	9	10
$\sum$ *Frames*	102	121	157	144	147	149	141	142	125	135
	Without learning scheme									
JI (%)	55.4	53.4	71.7	56.8	68.2	73.5	58.2	63.1	64.8	61.3
Avg. $\pm$ std	$62.6 \pm 6.5\%$									
	With learning scheme									
JI (%)	86.7	87.6	89.5	88.9	90.4	84.8	87.8	92.4	88.1	83.6
Avg. $\pm$ std	$87.98 \pm 2.58\%$									

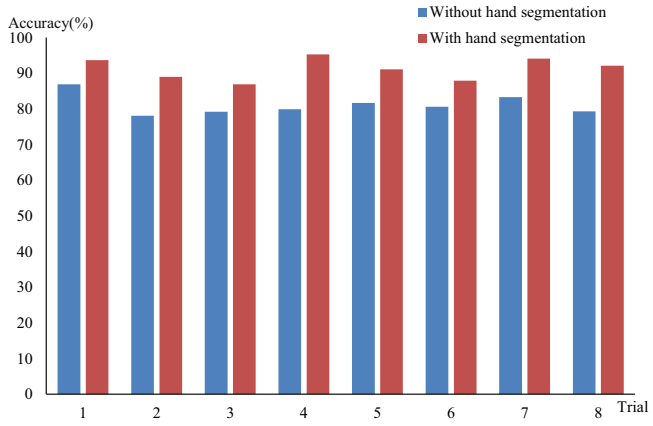

Fig. 11. Results of the kernel-based descriptors for hand posture recognition without/with segmentation.

is from starting time when their hands are located to stopping time when the system stops due to the convergence of the learnt parameters. Table 2 reports the required time of this step.

- *Stage* 3: To learn the hand to Kinect distances. The evaluators raise their hands in a few times until the system gives a stop message. Table 3 reports the required time of this step.

The required learning time for an end-user is averaged from the data shown in Tables 1–3. It is 14.21 ± 1.89 s per user for entire learning procedures.

4.2. The computational time for hand segmentation and recognition

After training procedures, we ask the participants to implement an additional step, that is Stage 4. In this stage, seven postures of each evaluator are shot, as shown in Fig. 10. We then analyze the log activities to calculate computational time for the segmentation and recognition procedures.

Table 4 reports the required time for detecting and segmenting hand regions from a captured image. The computational time for recognizing a hand posture is reported in Table 5. Consequently, by utilizing the proposed user-guide scheme, computational time to cover whole the proposed flow-works averagely is 285 ± 45 ms, or equal 3 fps. These costs are significantly lower than the hand detection methods in Mittal et al. (2011), and the hand posture recognition algorithms in Pisharady et al. (2012), respectively.

As shown in Figs. 10 and 9, the hand postures were shot in two different experimental rooms against complex natural backgrounds, and in different lighting conditions. For instance, Posture 6 in Fig. 10 is captured in relatively low lighting condition, whereas Posture 1 is captured in enough lighting condition. Furthermore, hand to Kinect distances are changed when we collect the dataset. Posture 3 is captured at relatively closed distance, whereas Posture 5 is shot relatively far one. These conditions made the hand posture dataset captured with various hand shapes and sizes.

4.3. Performance of the hand region segmentations

Fig. 10 (c) illustrates several segmented hands extracted the images captured at different view points. As shown, our proposed method segment correctly hand regions in different lighting conditions, against various natural complex backgrounds. For quantitative evaluation, we use Jaccard Index (McGuinness and O Connor, 2010) that is calculated by

$$JI = \frac{HDT \cap HGT}{HDT \cup HGT} \tag{14}$$

where *HDT* is regions of the hands segmented by the proposed method, *HGT* is ground-truth one. The segmentation result is better if the *JI* index is more closed by 100%. Table 6 shows *JI* indexes without/with using the proposed user-guide scheme. Obviously, by paying a cost for learning parameters, *JI* indexes are significantly increasing from 62.6% to 87.98%.

4.4. The hand posture recognition results

We evaluate performance of the proposed kernel-based descriptors method with seven hand postures as shown in Fig. 10(c). To avoid over-fitting by similar frames, we simply sampling at a rate of 1/3 from the captured image sequences. For preparing training and testing data, we follow *Leave-p-out-cross-validation* method, in which p equals 5. An accuracy of the recognition is a ratio which is calculated by the numbers of true detection per total postures testing. The accuracy for each testing set is given in Fig. 11. The horizontal axis denotes the trial implemented to validate the proposed algorithms. A trial means a validation with a test-set that is generated in scheme of the *Leave-p-out-cross-validation*. The vertical axis presents the average of the accuracy of the proposed hand posture recognition algorithms. Averaging on whole trials, the proposed method obtains the accuracy rate at $91.2 \pm 2.9\%$. This performance is comparable with results of the current state-of-art works (e.g., Pisharady et al., 2012 reports 93%). Moreover, Fig. 11 also confirms that by utilizing kernel-based descriptors, use of the segmented hands from complicated backgrounds outperforms a scheme of without hand segmentations for all of the testing set.

We continue evaluating the robustness of the kernel-based descriptors on a public hand posture dataset, that is NUS II (Pisharady et al., 2012). This evaluation is to confirm that the kernel-based descriptors are particularly improving the recognition rate for the segmented hand regions. NUS dataset consists of 10 type of postures, captured by 40 subjects, 5 images per class per

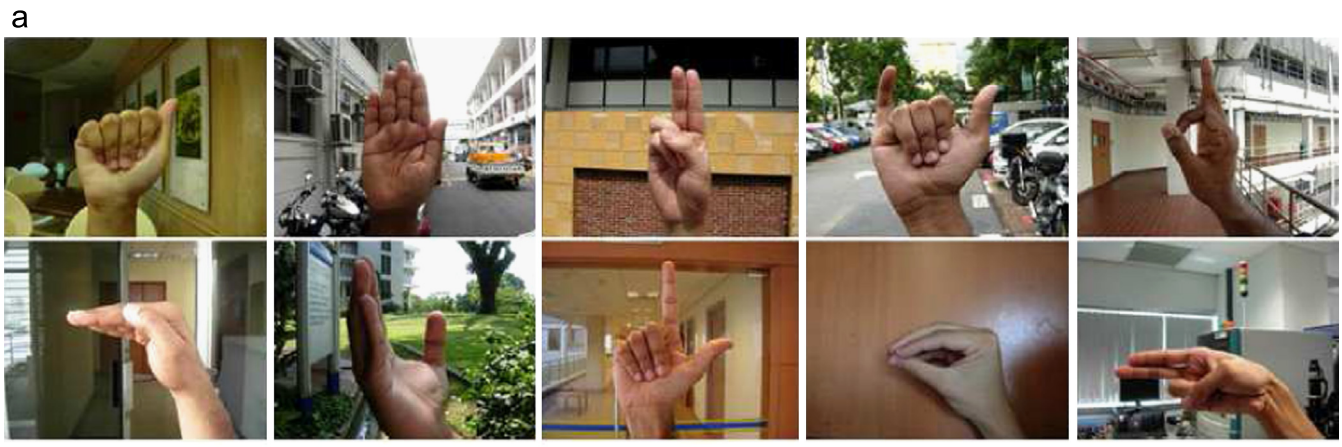

Fig. 12. Sample images from NUS II hand posture dataset. (a) Posture class of NUS II dataset from 1 to 10 (Pisharady et al., 2012) and (b) samples of cropped whole hand region images from NUS II dataset.

subject. Original examples of the NUS dataset are shown in Fig. 12 (a), whereas the corresponding hand regions are manually extracted, as shown in Fig. 12(b). By using the proposed kernel-based descriptors on RGB features, we obtain the best recognition rate at 97.4%. These results outperform the results reported in the relevant works (Pisharady et al., 2012). They also are better than the results reported in Fig. 11. The main reasons are that hand postures defined in NUS dataset are more discriminated than ours, and NUS hand dataset is captured at higher resolutions.

5. Discussions and conclusions

Discussions: According to the experimental results, we have demonstrated the efficiency of the proposed user-guide scheme for hand posture recognitions. It confirms that segmenting hand regions and hereby recognizing a hand posture are more accuracy. The fact that we have tried to setup the evaluations in different experimental rooms against various natural complex background. However, it is much more difficult to confirm that the proposed method could still work well due to the complicated lighting conditions, which may appear in indoor environments. While estimating the hand-Kinect distances to be more stable with lighting conditions (thanks to utilizing depth data), the parameters of the skin color model can be changed. As results, the hand regions will be over/or under pruned.

In current experimental setup, we asked subjects stand/or sit in front of Kinect sensors. This request may be not reasonable in the practices. In order to ensure natural behavior of the system, an end-user can stand in any direction to control a home alliance device. To solve this issue, a system utilizing multiple Kinects may be more appropriated. Fusing the hand regions from multiple Kinects could also resolve issues of the clustered backgrounds, particularly, when the background and hand skin colors are identical. In such cases, evaluating the Kernel-based descriptors from multiple view-points of the hand postures need to be implemented.

Feedbacks from end-users, who participated in the evaluations, have been not reported in our experimental results. For instance, a question can be raised: “Is it easy to do a learning stage?”. Obviously, an end-user could be failed in a learning phase. He/she is requested to implement again in that case. A survey on user's attitudes suggests us directions to improve the current system to be more convenient and efficient.

Conclusions: This paper described a robust and real-time vision-based hand posture recognition system. To achieve this goal, we designed an efficient user-guide learning scheme and represented a compactness of hand postures. We also reported that kernel-based descriptors significantly increases accuracy of the recognition rate when the hand regions are separated from the background. Performances of the proposed method are comparable with results of the stage-of-the art methods. However our system was relatively faster than those works. Consequently, the proposed method is feasible to deploy practical application, such as to control smart TV or smart lighting system in indoor. In the future, we will continue research on learning parameters for the dynamic hand gesture recognition as well as evaluating relevant feedbacks of the user-guide scheme.

Conflict of interest

None declared.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number FWO.102.2013.08.

References

⟨http://www.microsoft.com/en-us/kinectforwindows⟩, 2014.

Bergh, M., Bosché, F., Koller-Meier, E., Van-Gool, L., 2009. Haarlet-based hand gesture recognition for 3D interaction. In: Proceedings of the Workshop on Applications of Computer Vision.

Bergh, M., Van-Gool, L., 2011. Combining RGB and ToF cameras for real-time 3D hand gesture interaction. In: Proceedings of the Workshop on Applications of Computer Vision.

Bo, L., Ren, X., Fox, D., 2010. Kernel descriptors for visual recognition. In: NIPS, pp. 1–9.

Bo, L., Sminchisescu, C., 2009. Efficient match kernel between sets of features for visual recognition. In: NIPS, pp. 135–143.

Chen, F.-S., Fu, C.-M., Huang, C.-L., 2003. Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vis. Comput. 21 (August (8)), 745–758.

Company, P., 2013. PointGrab brings gesture control to home appliances. [Online]. Available: ⟨http://www.cnet.com/news/pointgrab-brings-gesture-control-to-home-appliances/⟩.

De Silvaa, L., Morikawab, C., Petraa, I., 2012. State of the art of smart homes. Eng. Appl. Artif. Intell. 25, 1313–1321.

Dipietro, L., Sabatini, M., Dario, P., 2008. A survey of glove-based systems and their applications. IEEE Trans. Syst. Man Cybern.–Part C 38, 461–482.

Doliotis, P., Athitsos, V., Kosmopoulos, D., Perantonis, S., 2012. Hand shape and 3D pose estimation using depth data from a single cluttered frame. In: Proceedings of the International Symposium on Visual Computing.

Erol, A., Nicolescu, M., Boyle, R., Twombly, X., 2007. Vision-based hand pose estimation: a review. Comput. Vis. Image Underst. 108, 52–73.

Herrera, D., Kannala, J., Heikkila, J., 2012. Joint depth and color camera calibration with distortion correction. IEEE Trans. Pattern Anal. Mach. Intell.

Ionescu, B., Coquin, D., Lambert, P., Buzuloiu, V., 2007. Dynamic hand gesture recognition using the skeleton of the hand. In: Proceedings of Annual Conference on Communication by Gaze Interaction.

Jones, M., Rehg, J., 1999. Statistical color models with application to skin detection. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.

Joslin, C., El-Sawah, A., Chen, Q., Georganas, D., 2005. Dynamic gesture recognition. In: Proceedings of IEEE on Instrumentation and Measurement Technology Conference.

Keskin, C., Erkan, L., Akarun, A., 2003. Real time hand tracking and gesture recognition for interactive interfaces using HMM. In: Proceedings on ICANN/ICONIP.

Lamberti, L., Camastra, F., 2011. Real-time hand gesture recognition using a color glove. In: Proceedings of Springer 16th International conference on Image Analysis (ICIAP), pp. 365–373.

Maji, S., Berg, A.C., Malik, J., 2013. Efficient classification for additive kernel SVMs. IEEE Trans. Pattern Anal. Mach. Intell. 35 (1), 66–77.

McGuinness, K., O Connor, N.E., 2010. A comparative evaluation of interactive segmentation algorithms. Pattern Recognit. 43 (February (2)), 434–444.

Mittal, A., Zisserman, A., Torr, P., 2011. Hand detection using multiple proposals. In: Proceedings of International Conference on British Machine Vision Conference.

Nguyen, T.T., Nguyen, D.B., 2008. An active boosting-based learning framework for real-time hand detection. In: Proceedings of International Conference on Automatic Face and Gesture Recognition.

Nguyen, V. T., Le, T. L., Tran, T. T. H., Remy, M., Vincent, C., 2015. A new hand representation based on kernels for hand posture recognition, In: Proceedings of International Conference on Face and Gesture (FG).

Park, J.H., Choi, Y.K., 1996. On-line learning for active pattern recognition. IEEE Signal Process. Lett.

Park, M., Hasan, M., Kim, J., Chae, O., 2012. Hand detection and tracking using depth and color information. In: Proceedings of IPC.

Pisharady, P.K., Vadakkepat, P., Loh, A.P., 2012. Attention based detection and recognition of hand postures against complex backgrounds. Int. J. Comput. Vis. (August).

Qifan, P., Gupta, S., Gollakota, S., Patel, S., 2013. Whole-home gesture recognition using wireless signals. In: Proceedings of the 19th Annual International Conference on Mobile Computing and Networking.

Rautaray, S., Agrawal, A., 2012. Vision based hand gesture recognition for human computer interaction: a survey. Artif. Intell. Rev. (November).

Ren, Z., Yuan, J., Zhang, Z., 2011. Robust hand gesture recognition based on finger-earth movers distance with a commodity depth camera. In: Proceedings of the 19th ACM International Conference on Multimedia.

Ren, Z., Yuan, J., Meng, J., Zhang, Z., 2013. Robust part-based hand gesture recognition using kinect sensor. IEEE Trans. Multimed. 15 (5), 1110–1120.

Stauffer, C., Grimson, W., 1999. Adaptive background mixture models for real-time tracking. In: Proceedings of Computer Vision and Pattern Recognition.

Vedaldi, A., Zisserman, A., 2012. Efficient additive kernels via explicit feature maps. IEEE Trans. Pattern Anal. Mach. Intell. 34 (March (3)), 480–492.

Viola, P., Jones, M., 2001. Robust real-time object detection. In: Proceedings of IEEE Workshop on Statistical and Computational Theories of Vision.

Yang, H.-D., 2015. Sign language recognition with the kinect sensor based on conditional random fields. Sensors 15, 135–147.

Zabulis, X., Baltzakis, H., Argyros, A., 2009. Vision-based Hand Gesture Recognition for Human Computer Interaction. Lawrence Erlbaum Associates, Crete, Grece.