2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)
Oct. 20-23, 2013 in Kimdaejung Convention Center, Gwangju, Korea

Human Computer Interface Using the Recognized Finger Parts of Hand Depth Silhouette via Random Forests

Dinh Dong Luong[1], Sungyoung Lee[1], and Tae-Seong Kim[2]

[1]Department of Computer Engineering, Kyung Hee University, South Korea
(E-mail: luongdd@oslab.khu.ac.kr, sylee@oslab.khu.ac.kr)
[2]Department of Biomedical Engineering, Kyung Hee University, South Korea
(E-mail: tskim@khu.ac.kr)

Abstract: Hand gesture recognition provides an attractive option for Human Computer Interaction (HCI). In particular, vision-based recognition of finger and hand gestures can help humans to communicate with a computer more efficiently. In this paper, we present a novel approach of recognizing finger and hand parts from a hand depth silhouette using Random Forests (RFs), a multi-class classifier, and its use for a hand gesture HCI. We present how to train the RFs using our own database. Then, the trained RFs are used to recognize finger and hand parts, which are used to recognize hand gestures. We also present an HCI application of finger mouse in which the computer cursor is controlled with a recognized finger.

Keywords: Human computer interaction, random forests, hand gesture recognition, depth image.

1. INTRODUCTION

Recently, finger and hand gestures-based interface provides an attractive choice to interface between humans and computers. In general, gestures are defined as physical movements of human body parts such as hands, face, or other parts of the body which convey meaningful information. This gesture interface field has attracted lots of attention from researchers in computer engineering and robotics [6]. Among different body parts, finger and hand parts are getting more attention, since they provide a wide variety of gestures to be used in interfacing. The finger or hand based interface offers many applications in Human Computer Interaction (HCI) such as virtual reality, sign language, human-robot interaction, and human-computer interaction [9], [10], [13], [14], [15]. However, recognizing of finger and hand gestures is still an open topic with many challenges.

Previously, two kinds of approaches are commonly used to recognize hand gestures for HCI [6], [7], [11], [12]. The first approach is based on data gloves. This method employs the use of motion sensors attached to a glove that transduce finger or hand actions into electrical signals to recognize hand gestures. For example, gesture recognition for virtual reality applications using data gloves as an input device is done via the angles of measured different finger joints. The set of angular measurements is mapped to a set of pre-defined hand gestures. However, this method not only requires users to wear or carry a load of cables which are connected to a computer but also hinders naturalness of the user interaction. The second approach is based on machine vision. Machine vision-based hand gesture HCIs are based on cameras from which vision information such as color, illumination, boundary, and edge is obtained. In general, the vision-based approach requires complex calibration and setup procedures, since the quality of image captured from cameras is sensitive to noise, lighting conditions, and cluttered backgrounds [8], [10]. As an example, Wang et al. [9] proposed a method of hand tracking using a glove that is imprinted with a custom pattern: no sensors are embedded in or outside the glove to restrict movement. To recognize hand gestures, the glove is coded with a color pattern. This pattern of the glove facilitates faster and more robust gesture recognition. However, this approach is not as accurate as the traditional optical marker-based motion capture device.

To overcome the fore-mentioned limitations, in this work, we present a novel approach of recognizing finger and hand parts from a depth silhouette via Random Forests (RFs). In our approach, we first obtain a hand depth image from a depth camera. To recognize finger and hand parts from the hand depth image, a database of pairs of depth images and finger and hand parts-labeled maps is created by using a 3D hand modeling computer graphic tools. This database is used in training RFs. Finger and hand parts recognition is then performed based on the trained RFs. The recognized finger and hand parts are used for hand gesture recognition. Finally, an application of a finger mouse has been developed in which a computer mouse is controlled by recognized fingers.

The rest of the paper is organized as follows. Section II introduces the processes of the proposed methodology including synthetic database creation, hand detection and segmentation, finger and hand parts recognition via RFs, and finally an HCI application. Sections III presents experimental setups and results. Discussion and conclusion remarks are followed in Section IV.

2. METHODS

As illustrated in Fig. 1, our proposed framework is divided into two main steps: training and recognition processes. In the training step, to train RFs, a database (DB)

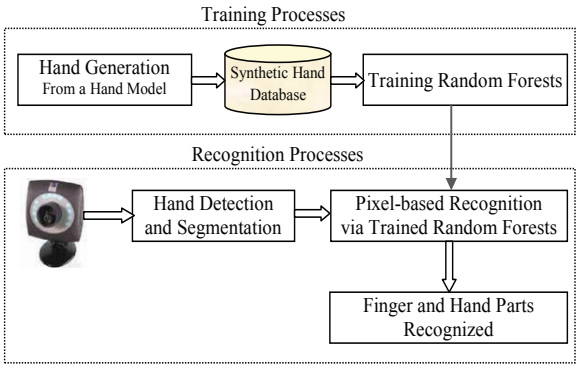


Fig. 1 Flows of finger and hand parts recognition via RFs.

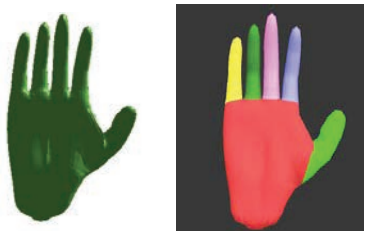
is first created in which pairs of depth images and their corresponding finger and hand parts pre-labeled maps are included. The synthetic DB is generated from a hand model created by using computer graphic tool, such as Autodesk 3Ds Max. In the recognition step, each depth image is first captured directly from a depth camera and then the hand in the depth image is detected and segmented. The finger and hand parts in the image are recognized by the trained RFs.

2.1 Depth database

To recognize finger and hand parts from a depth image, RFs need to be trained using a set of data containing pairs of depth images and finger and hand parts pre-labeled maps. In this work, we build such a DB using Autodesk 3Ds Max [16]. To generate the synthetic hand DB, three main steps are involved. In the first step, a synthetic hand model is built including palm part, thumb, index, middle, ring, and pinky fingers. In the second step, the 3D finger and hand model consisting of hand bone and skin parts is rendered as presented in Fig. 2(a). Each finger or hand part is assigned a different color. The results are shown in Fig. 2(b). In the final step, from the color labeled finger and hand parts of the hand model, pairs of depth image and finger parts-labeled maps are created. Some demonstrative finger and hand poses are shown in Fig. 3. Figs. 3(a) and 3(b) illustrate depth images and finger-parts pre-labeled maps respectively. The synthetic hand DB from the hand model consists of a total of 800 images covering the four different gestures. Each type of gestures in DB is presented with 200 poses that consist of depth images and finger and hand pre-labeled maps. The resolution of each image is 320 x 240 pixels.

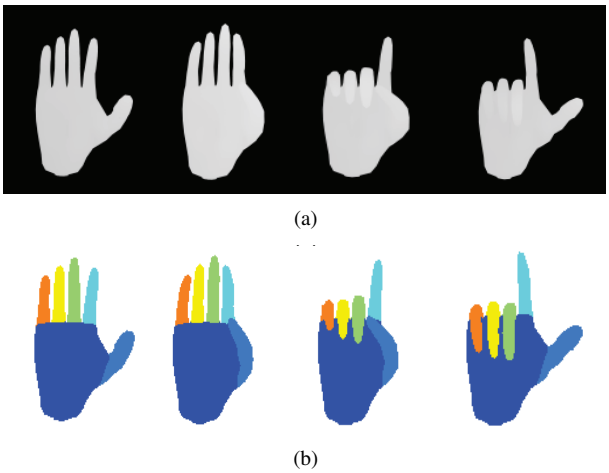
2.2 Hand detection and background removal

To detect the hand area and remove background, we use an approach using an adaptive depth threshold. The value of threshold was determined based on a specific distance from a depth camera to the hand objects. The detected and segmented hand is shown in Fig. 4.



(a) (b)

Fig. 2 A 3D Max hand model: (a) 3D finger and hand model and (b) labeled finger and hand parts.



(a)

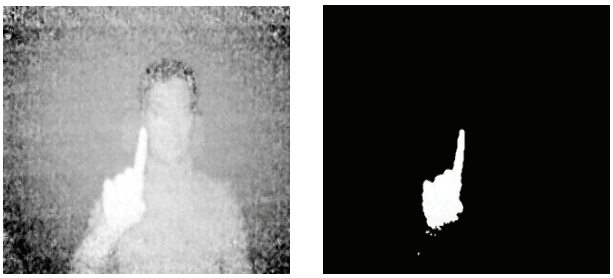
(b)

Fig. 3 Samples of four types of hand poses in training database: (a) sample depth images, (b) finger and hand parts-labeled maps

2.3 Random forests for pixel-based classification

In this part, we describe a method of pixel-based classification utilizing RFs. Each class corresponds to one in the six defined finger and hand parts. The details of this pixel-based classification algorithm are presented the following part.

- *Feature extraction*: features are one of key factors to decide an accuracy of classification. In our work, by using a large quantity of features to improve the accuracy of pixel-based classification, we extract the features based on the depth value difference among pixels. Specifically,



(a) (b)

Fig. 4 Hand detection and segmentation: (a) the captured depth image and (b) the segmented hand shape using an adaptive threshold.

to extract features f from a depth image I at pixel x as [1], herein, each pixel x of depth image I is considered to extract one or many features. For each feature that is extracted based on the depth value difference of a pixel pair which is a neighborhood relation with the considered pixel x by two terms u and v. The neighbor pixel pairs of the considered pixel belong to range of a defined region named a window region. The size of the window region is defined by user based on the size of subject in image I. According to [3], [4], at a given pixel x of depth image I, the feature is computed as follows:

$$f_\theta(I,x) = \left[d_I\left(x + \frac{u}{d_I(x)}\right) - d_I\left(x + \frac{v}{d_I(x)}\right)\right]^2 \quad (1)$$

where $d_I(x)$ is the depth value at pixel x in image I, and parameters $\theta = (u, v)$ describe offset u and v. The normalization of the offset by $\frac{1}{d_I(x)}$ ensures that the features are distance invariant.

- *RFs for classification*: RFs are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently with the same distribution for all trees in the forest [2]. The tree predictors in tree forests are decision trees, so RFs can be called randomized decision forests.

Suppose a randomized forest is an ensemble of T decision trees. Each tree T_t in RFs is trained on a set of N samples drawn randomly from a training data set which includes a feature vector set and a corresponding ground truth vector extracted from all images in synthetic DB. The set of N samples (S^*) is chosen randomly, it has to ensure a distribution of samples over considered subject in the image I. To solve this work, a bootstrap replacement method is used to draw N samples from the training data set. At each test node of the tree T_t, optimal split is derived by searching a random subset of m candidate attributes which are selected without replacement from the candidate attributes. Based on [1], [2], [3], [5], the detail of this RFs algorithm for classification is presented in Algorithm 1.

Algorithm 1 RFs for pixel-based classification

A. Training:

I. Input: A training data set

Let N_t be the number of used trees to build

FOR each t of N_t iterations

- Draw a bootstrap sample S^* size N from training data set
- Grow a random forest tree T_t on the bootstrapped samples

Repeat

– At each internal node of the tree, randomly select a set of m variables to splitting candidates.

– Pick the best variance among m selected variables

– Split the node into two sub-nodes

Until stopping condition is met one of minimum information gain, minimum node size, or maximum depth in tree.

END FOR

II. Output: The built ensemble of tree $\{\hat{\mathrm{T}}_t\}_{t=1}^{N_t}$

B. Classification: To classify a new point p based on N_t built trees from the training data set

- Let $\hat{\mathrm{C}}_t(p)$ be the class prediction of the t^{th} random-forest tree.
- Then $\hat{\mathrm{C}}_{RFs}^{N_t}(p)$=majority vote $\{\hat{\mathrm{C}}_t(p)\}_{t=1}^{N_t}$

2.4 Finger and hand parts recognition

Using the trained RFs, each pixel is classified to assign a corresponding label. In our work, the six considered labels correspond to the six parts of the hand model including the palm and the fingers of thumb, index, middle, ring, and pinky. From the recognized finger and hand parts, instead of computing all pixels that have the same label value in the hand depth image, we only determine the centroid point of each part based on the region where the distribution of the same label pixels is the largest. Each centroid point corresponds to the recognized finger or hand part. The information of the centroid point includes a label index, a 3D coordinate of x, y and z. This information is used later in an HCI application.

2.5 A finger mouse

Instead of using the traditional pointing devices such as mouse, or a track ball to control computers, an user is looking for an intuitive and cost-efficient interaction device. In this work, we have developed an application of HCI in which a cursor is controlled by the recognized finger named finger mouse. Using the information of centroid point of the recognized finger and hand parts, we move a cursor and some operations at the cursor position such as a single-click and double-click of left mouse button. We present two demonstrations corresponding to the main operations of the finger mouse.

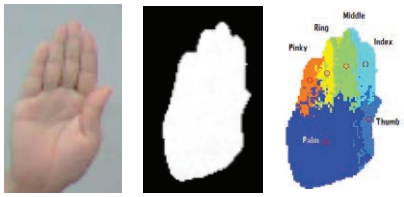
3. EXPERIMENTAL RESULTS

3.1 Experimental setups

In our training hand DB, the total number of training image pairs is 800. Each image size is 320 x 240 with the 8 bit depth value. In the feature extraction, at each pixel in the hand depth image, 100 features were extracted based on 100 randomly selected u, v offset pairs. The maximum offset value of u,v pairs was 40 pixels. RFs parameters were set as follows: the number of tree N_t was 20, and the subsets of candidate features m was 14. The testing depth data was captured in real-time using a depth camera, Zcam [17].

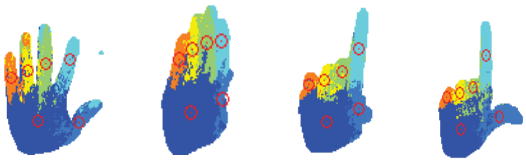
3.2 Recognition of hand and finger parts

In this section, we have evaluated our techniques with synthetic data and real data. In the first experiment, we



(a) (b) (c)

Fig. 5 A recognition example of finger and hand parts from an input depth image of a hand: (a) RGB image of the hand, (b) depth image of the hand, and (c) the recognized finger and hand parts and their centroids in circles.



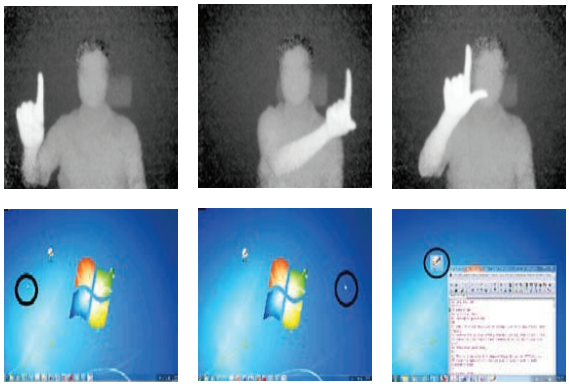
(a) (b) (c) (d)

Fig. 6 The sample results of the recognized finger of hand parts from their corresponding hand depth images via the trained RFs.

used two different synthesized DB for training and testing RFs with the defined parameters in experimental setups. Average pixel-based recognition rate achieved 97.10% for 200 testing images and 800 training images. The recognition results of the hand parts are given in a confusion matrix in Table 1. In the second experiment, we have experimented on real data. Input data was captured from a depth camera. Herein, the hand depth silhouette was obtained by the hand detection algorithm using an adaptive threshold as illustrated in Fig. 4. The finger parts in the hand depth image were recognized by the trained RFs. Each centroid point correspond to a recognized finger part. The qualitative assessment of recognition results are given in Figs. 5 and 6. The information of the centroid points includes a label and a 3D coordinate of x, y, and z value. In Fig. 6, the experimental results of the recognized finger parts in the depth image of four gestures are shown. The points in the red circles indicate the recognized finger and hand parts. The six centroid points represent the palm part, thumb, index, middle, ring, and pinky fingers respectively.

3.3 A finger mouse

With the recognized finger and hand parts, we built an application in which a recognized finger was used to control a computer cursor. In this application, two basic operations were implemented: one operation moves a cursor on a display screen of computer and the other performs double-click. The two chosen gestures in this experiment were the gesture in Fig. 6(c) and Fig. 6(d) corresponding to move the cursor and to open an application at the cursor position. To control the movement of the cursor, we need to recognize the pose and then tracke the centroid of the index finger. Since, the coordinate of the centroid point of index finger in image was mapped to the coordinate of the cursor pointer on a computer screen. Figs. 7 (a) and (b) illustrate the results of moving the cursor from left to right. The cursor positions were indicated with the black circle. At the position of the cursor, the "winedit" program was open as illustrated in Fig. 7(c).



(a) (b) (c)

Fig. 7 A demonstration result of the figure mouse: (a)-(b) dragging a mouse cursor from the left to the right, and (c) opening a Winedit program.

4. CONCLUSION AND DISCUSSION

In this paper, we have presented a novel work for depth image based finger and hand parts recognition using RFs and its application as a finger mouse in which a cursor is controlled by the recognized finger. Our proposed system have developed in Matlab takes an average of 100 ms (10 fps) on a 2.5 GHz Inter Core 2 Duo Processor to classify each frame. The current work is mainly dedicated to improve the robustness of the system. For the future work, we plan to extend the number of the recognized finger and hand parts of the hand. To improve the rate of finger and hand parts recognition on real data, we will increase not only the size of training DB but also the number of extracted features. Based on the 3-D information of the centroid points such as their directions and positions, we can improve the hand gesture recognition.

ACKNOWLEDGMENTS

This research was supported by the MSIP(Ministry of Science, ICT&Future Planning), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Pro-

Table 1. A confusion matrix for hand parts recognition

	$Palm$	$Pinky$	$Ring$	$Middle$	$Index$	$Thumb$
$Palm$	**0.9887**	0.0022	0.0031	0.0012	0.0039	0.0009
$Pinky$	0.0051	**0.9915**	0.0013	0.0004	0.0012	0.0004
$Ring$	0.0065	0.0008	**0.9905**	0.0008	0.0003	0.0012
$Middle$	0.0135	0.0011	0.0002	**0.9732**	0.0027	0.0093
$Index$	0.0305	0.0021	0.0029	0.0009	**0.9639**	0.0005
$Thumb$	0.0619	0.0041	0.0071	0.0029	0.0052	**0.9189**

motion Agency)" (NIPA-2013-(H0301-13-2001)). This work also was supported by the Industrial and Strategic technology development program, (10043977, Development of a Sustainable and Practical Wellness System using Emotion Mechanism and Smart Media for the Elderly) funded by the Ministry of Knowledge Economy(MKE, Korea).

REFERENCES

[1] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," Neural Computationo, Vol. 9, No. 7, pp. 1545–1588, 1997.

[2] L. Breiman, "Random forests," ML Journal, Vol. 45, No. 1, pp. 5–32, 2001.

[3] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," In Proc, CVPR, 2005.

[4] P. Kontschieder, S. R. Bulo, H. Bischof, and M. Pelillo, "Structured Class-Labels in Random Forests for Semantic Image Labelling," In Proc. ICCV, 2011.

[5] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," In Proc. CVPR, 2011.

[6] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," International Journal of Information Technology and Knowledge Managenment, Vol. 108, pp. 52–73, 2007.

[7] G. R. S. Murthy, and R. S. Jadon, "A rewiev of vision based hand gesture recognition," Computer Vision and Image Understading, Vol. 2, No. 2, 2009.

[8] E. Dente, A. A. Bharath, Jeffrey-Ng, A. Vrij, S. Mann, and A. Bull, "Tracking hand and finger movements for behavior analysis," Pattern Recognition Letters, Vol. 27, pp 1797–1808. 2006.

[9] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," ACM Trans. Graph, Vol. 328, No. 3, 2009.

[10] X. Yin and M. Xie, "Finger identification and hand posture recognition for human robot interaction," Image and Vision Computing, Vol. 25, pp. 1291–1300, 2007.

[11] J. R. Parker and M. Baumback, "Finger Recognition for Hand Pose Determination," In Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics, pp. 2492–2497, 2009.

[12] M. K. Bhuyan, Debanga-Raj-Neog, and Mithun-Kumar-Kar, "Fingertip Detection for Hand Pose Recognition," IJCSE, Vol. 4, No. 3, 2012.

[13] Zhou-Ren, Jingjing-Meng, and Junsong-Yuana, "Depth Camera based Hand Gesture Recognition and its Applications in Human-Computer-Interaction," In Proc, ICICS, 2011.

[14] A. Antonis Argyros and Manolis I.A. Lourakis, "Vision-Based Interpretation of Hand Gestures for Remote Control of a Computer Mouse," LNCS Vol. 3979, pp. 4051, 2006.

[15] Ali-Erol, George-Bebis, Mircea-Nicolescu, Richard D. Boyle, and Xander Twombly, "Vision-based hand pose estimation: A review," Computer Vision and Image Understanding, Vol. 108, pp. 52–73, 2007.

[16] http://docs.autodesk.com/3DSMAX/15/ENU/3ds-Max-help/index.html.

[17] http://www.3dvsystems.com.