FINGER-WRITING-IN-THE-AIR SYSTEM USING KINECT SENSOR

Zhichao Ye, Xin Zhang, Lianwen Jin, Ziyong Feng, Shaojie Xu

School of Electronic and Information Engineering, South China University of Technology, China
dante.ye.2011@gmail.com, xin.zhangcn@gmail.com, lianwen.jin@gmail.com

ABSTRACT

A novel Kinect based finger-writing character recognition system called finger-writing-in-the-air is proposed in this paper. This system allows people to input characters by writing in the air virtually. We propose a mixture model for hand segmentation, which takes advantage of depth, color, and motion information. This model helps overcome problems arisen with traditional vision-based methods like illumination variation or hand-face overlapping, also solves the fatal color-depth mismatch problem accompanied by Kinect. A dual-mode switching algorithm is presented for accurate fingertip detection. It is less sensitive to noise along the segmented hand contour and can handle various hand poses. The positions of fingertip are linked and reconstructed as inkless character strokes. The recovered trajectory is recognized by the state-of-the-art handwriting character recognition method. Experiments show that user can write freely in the air inputting Chinese characters, English letters (upper and lower case) and digits in real time with over 90% accuracy rate for the first five candidates.

Index Terms— Writing system, hand segmentation, fingertip detection, Kinect

1. INTRODUCTION

So far most of writing systems still rely on extra devices such as keyboard or touch screen. These technologies have not met the essential goal of HCI: making interaction between user and computer more natural. Hence, using bare-hand for the input and interaction is the ultimate goal. One kind of bare-hand-writing system is introduced in [1], where finger is writing on a flat surface with a digital camera above to catch the scene. This vision-based system faces a few restrictions like relative clear and static background and strict constraint on the hand pose. The introduction of Microsoft Kinect could lead to a new solution in the computer vision field by effectively combining depth and color data. Here, we propose a finger-writing-in-the-air system based on Kinect to provide a user-friendly and unconstrained bared-hand writing system.

Hand segmentation is usually the first step of hand-based application. In existing work, skin model has been widely used and various models have been proposed to characterize skin color, like single Gaussian model (SGM) [2], Gaussian mixture model (GMM) [2] etc. These methods suffer from varying illumination and overlapping between hand and face. Frame-difference based motion-cue method [3] is also applied. It relies strongly on the assumption that hand should be the moving object. Researchers take advantages of the depth information from Kinect which is robust against environmental changes and overlapping for segmentation using threshold method [4] or Gaussian model [10]. But solely applying these low quality depth sequences often results in unsatisfied performance. RGB and depth information are combined in [5]. However, the color-depth mismatch problem becomes a new challenge because they are not updated at the same time. Therefore, segmentation still faces significant challenges.

After hand segmentation, fingertip detection algorithm is used to track trajectory of writing. Previous research can be divided into two kinds: 2D vision-based model and 3D model. In the case of 2D methods, local maximum curvature [6] and template matching [1] are the most commonly used methods. They require clean hand segmentation, and tend to fail when finger is pointing to the camera since the fingertip contour is unavailable. 3D hand models have achieved nice results, but required special equipment [7]. Also, high computational cost makes it infeasible for real-time application. 3D clustering in [10] may suffer from noise and data missing as it use depth-cue from Kinect solely.

In this paper, we propose a novel finger-writing-in-the-air system in which the user can write characters in the air with their fingertip in real-time. Hand is segmented from background firstly. The trajectory of writing is extracted by a physics-based fingertip detection algorithm, and linked and reconstructed as an inkless character. The state-of-the-art handwriting character recognition method is employed for the final output. Our main contributions lie in three aspects: (1) We propose the idea of finger-writing-in-the-air which is a bared-hand HCI with much more easy and user-friendly writing experience. (2) A depth-skin-background mixture model (DSB-MM) for hand segmentation solves the problems like illumination variation, hand-face overlapping and the color-depth mismatch problem accompanied by Kinect. (3) A dual-mode switching algorithm accurately detects the fingertip from various hand poses.

2. DSB-MM HAND SEGMENTATION

2.1. Depth model

Illumination variation and hand-face overlapping are the most challenging problems of appearance-based hand segmentation. They could be partially solved by introducing the depth information. User map offered by Kinect also helps extract human body from cluttered background.

The basic assumption of this model is that the hand is the closest object of human body to sensor during writing. We first remove unmeasured pixels (with value 0) of depth image (16 bits) and pixels not belonging to body region ($u(x,y)=0$) by converting them as the farthest points:

$$d(x,y)=65535 \text{ if } d(x,y)=0 \text{ or } u(x,y)=0, \quad (1)$$

where $d(x,y)$ and $u(x,y)$ indicate the pixel of depth image and user map at point $\mathbf{p}=(x,y)$. Here, $1\le x\le w$ and $1\le y\le h$ where w and h are the width and height of image. By finding the smallest depth value d_{min} of body region, a mask of hand D is defined accordingly by applying an threshold:

$$D(x,y)=\begin{cases}1 & \text{if } d(x,y)\le d_{min}+\tau_d\\ 0 & \text{otherwise,}\end{cases} \quad (2)$$

where τ_d indicates the depth of hand decided by statistics.

The depth information is unreliable around the edges of hand, as shown in Fig. 1 (b). Also, the color-depth mismatch problem becomes severe when hand is moving relative fast (thumb is missing). Hence, other models are needed.

2.2. Skin model

Skin model plays an important part in wide range of hand-based researches. In this paper, we build a robust skin model characterizing both skin and non-skin distributions as single Gaussians in the *YCbCr* color space. Our work is much similar to the Gaussian classifier mentioned in [2]. Instead of building a skin model directly in 3D space as [2] did, we quantify Y component into three regions to reduce the storage size: bright ($170\le Y\le 255$), normal ($85\le Y\le 169$) and dark ($0\le Y\le 84$). A pixel $\mathbf{I}(x,y)=(Cb(x,y),Cr(x,y))$ is classified by calculating the difference between squared Mahalanobis distances of skin class and non-skin class:

$$S(x,y)=\begin{cases}1 & \text{if } dis_s^i-dis_{ns}\le\tau_s^i\\ 0 & \text{otherwise,}\end{cases} \quad (3)$$

$$dis_s^i=(\mathbf{I}(x,y)-\mathbf{m}_s^i)^T{\mathbf{C}_s^i}^{-1}(\mathbf{I}(x,y)-\mathbf{m}_s^i), \quad (4)$$

$$dis_{ns}=(\mathbf{I}(x,y)-\mathbf{m}_{ns})^T{\mathbf{C}_{ns}}^{-1}(\mathbf{I}(x,y)-\mathbf{m}_{ns}), \quad (5)$$

where $i\in\{1,2,3\}$ is the index of skin class depending on the Y component of inputted pixel. $\mathbf{m}_s^i$ and $\mathbf{C}_s^i$ are the mean vector and covariance of the i-th skin class, $\mathbf{m}_{ns}$ and $\mathbf{C}_{ns}$ are the mean vector and covariance of non-skin class. τ_s^i is a threshold of the i-th region. Once skin and non-skin Gaussian models are created, Mahalanobis distances-based lookup tables are generated to save computational load.

Previous work [5] tried to simply combine results of depth and skin model. As illustrated in Fig. 1, the combined depth-skin model helps to remove redundant background pixels from depth model segmentation result and skin-like background pixels from skin model segmentation result. However, the problem of color-depth mismatch still exists. Only the overlapping part of these two models is segmented.

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

Fig. 1 (a) Color image, (b) zoomed-in depth model result, (c) skin model result, (d) zoomed-in combination of (b) and (c).

2.3. Background model

The color-depth mismatch problem mentioned above is arisen because two images are not recorded at the exact same time by Kinect, that is, they do not represent the same scene. Since missed pixels belong to the moving foreground object, we believe a background model could help. We apply the codebook background model in [8] and make some improvements pertinently. The codebook model detects object of interest as foreground (hand in our case) by:

$$F(x,y)=\begin{cases}0 & \text{there is matched code word for } \mathbf{I}(x,y)\\ 1 & \text{otherwise,}\end{cases} \quad (6)$$

During background model updating, we update pixels detected as non-hand region by the DSB-MM which will be introduced in the next subsection. This target-oriented codebook model can capture changes of the background scene immediately and avoid the hysteresis phenomenon in [8]. Even if hand keeps still for a long time, pixels of it won't be "absorbed" or wrongly learnt as new interest.

Fig. 2 shows some results of codebook model. It collapses when hand is moving around face since face color information has been learnt as background before.

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

Fig. 2 Color image ((a)(c)) and, foreground results ((b)(d)).

2.4. DSB-MM segmentation

Since three models mentioned above have their own advantages and limitations, we combine them intelligently and propose a depth-skin-background mixture model (DSB-MM). It works like an expert voting system. Instead of simply voting "yes" (pixel of segmentation result has value 1) or "no" (value 0 correspondingly), we make it more adaptive, for each model should have different reliabilities in different circumstances. Results of these three models are weighted by their own confidence factors. The final DSB-MM segmentation result depends on the linear combination of them. A pixel is kept as hand pixel by:

$$H(x,y)=\begin{cases}1 & \text{if } \alpha*D(x,y)+\beta*S(x,y)+\gamma*F(x,y)\ge 1\\ 0 & \text{otherwise,}\end{cases} \quad (7)$$

where α, β and γ stand for confidence factors of each model.

Inputs of ANN are three Overlap Rates (*OLR*). *OLR* measures consistency of pair-wise models' segmentation results. For example, OLR_{s_d} is defined as:

$$OLR_{s\_d} = \frac{\sum_{x=1}^{w}\sum_{y=1}^{h} S(x,y) \wedge D(x,y)}{\sum_{x=1}^{w}\sum_{y=1}^{h} D(x,y)}, \quad (8)$$

OLR_{s_f} and OLR_{f_d} have the similar form. Outputs of ANN are confidence factors of three models. The more one model agrees with the other two, the higher confidence factor should be given. There are two assumptions: (1) All the models contribute to the final result, which means every model has a confidence factor larger than 0. (2) None of them is absolutely reliable which leads to the conclusion that a pixel finally treated as hand must be detected as target by at least two models. In our case, the confidence factor is quantified as three values: α, β, $\gamma \in \{1/3, 1/2, 2/3\}$. This setting covers all possible confident factor combinations based on the above two assumptions.

This ANN contains 3 layers with 3 inputs, 9 outputs (3 as a group labeled as "1 0 0", "0 1 0" or "0 0 1" representing 1/3, 1/2 or 2/3 while training) and the hidden layer including 30 neurons. During training stage, 3,659 video frames are used. The one with better details and less over-segmentation is chosen manually from 27 (3*3*3) factor combinations selections. Fig. 3 shows the architecture of ANN.

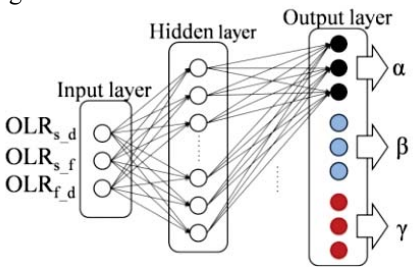
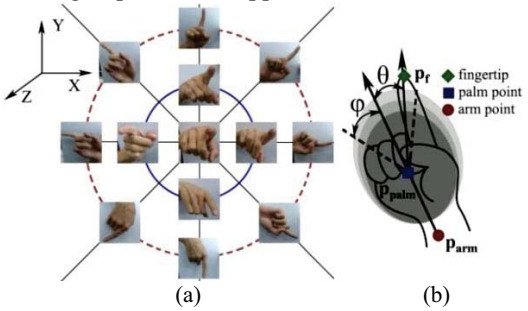


Fig. 3 The architecture of artificial neural network (ANN).

3. FINGERTIP DETECTION

After hand segmentation, a fingertip detection algorithm is applied to track the trajectory of writing. We propose a dual-mode switching algorithm: the ***side-mode*** and ***frontal-mode***. These two modes cover all possible hand poses but adopt different fingertip detection approaches.



(a) (b)

Fig. 4 (a) Writing hands of different poses, belong to two modes, (b) A physical model of hand for the fingertip detection.

(1) The ***side-mode*** indicates the finger is not pointing to the camera, as shown in the outer circle (red dotted line) in Fig. 4 (a). In the segmented hand region, the fingertip is supposed to be the farthest point from the arm point. Hence, we adopt the finger-hand-arm physical constraint for the fingertip detection, which is less sensitive to noise.

(2) The ***frontal-mode*** indicates the finger is almost pointing to the camera, as shown in the inner circle (blue solid line) and center one in Fig. 4 (a). In this mode, fingertip is not always the farthest point from the arm in the segmented 2D region, but is definitely the nearest point of hand region to the camera along Z-axis. So the fingertip is the point with the smallest depth value in the hand region. The ***frontal-mode*** can handle specific poses when visual cues are invalid.

3.1. Side-mode fingertip detection

We build a physical model for the ***side-mode*** fingertip detection, shown in Fig. 4 (b). When the hand is writing, the finger is usually holding out straight. Hence, the fingertip is not only the farthest point from the arm in 2D image, but also satisfied finger-hand-arm angle relationship.

The palm point in Fig. 4 (b) can be obtained by applying the ellipse fitting technique. Central point of the ellipse is regarded as the palm point. Recalling the depth threshold segmentation in Section 2.1, here we increase the threshold a little so that more region is included. These extra pixels definitely belong to the arm. The arm point can be located as the center of the increased region. Given the position of palm and arm point, the direction of hand is got.

Firstly, we calculate the point with the farthest distance to the arm point in the 2D segmented hand region $\mathcal{R}_{hand}$:

$$\mathbf{p}_f = \underset{\mathbf{p}\in\mathcal{R}_{hand}}{\operatorname{argmax}} \|\mathbf{p} - \mathbf{p}_{arm}\|, \quad (9)$$

$$\mathcal{R}_{hand} = \{(x,y) | H(x,y) = 1\}, \quad (10)$$

where $\|\cdot\|$ represents the L2 norm.

To make sure we identify the fingertip point in the right mode, we need another criterion: the angle between the hand direction and the line connecting fingertip and palm point:

$$\theta = 180° - \left| \cos^{-1} \frac{\overrightarrow{\mathbf{p}_{palm}\,\mathbf{p}_f} \cdot \overrightarrow{\mathbf{p}_{palm}\,\mathbf{p}_{arm}}}{\left|\overrightarrow{\mathbf{p}_{palm}\,\mathbf{p}_f}\right| \left|\overrightarrow{\mathbf{p}_{palm}\,\mathbf{p}_{arm}}\right|} \right|. \quad (11)$$

Based on our analysis and experimental observations, if the following two conditions are satisfied, the hand pose belongs to the ***side-mode*** and fingertip is $\mathbf{p}_f$ defined in (9).

(1) $\theta \le \varphi$, $\varphi = 30°$ considering human physical limitation;

(2) $\mathbf{p}_f \notin \mathcal{R}_{ellipse}$ (the region of fitted ellipse).

Otherwise, we switch to the ***frontal-mode.***

3.2. Frontal-mode fingertip detection

This mode defines the fingertip as the point with the smallest depth value in the segmented hand region (that is, the nearest point to the Kinect senor along Z-axis):

$$\mathbf{p}_f = \underset{\mathbf{p}\in\mathcal{R}_{hand}}{\operatorname{argmin}} d(\mathbf{p}), \quad (12)$$

where $d(\mathbf{p})$ is the depth value in the depth image at point $\mathbf{p}$.

4. FINGER-WRITING CHARACTER RECOGNITION

The finger-writing trajectory is generated by linking all detected fingertip positions together. The linked trajectory is

then passed into a mean filter to remove noise. Some examples of reconstructed characters are given in Fig. 5. We use a state-of-the-art compact MQDF character classifier [9]. It can recognize 6,763 frequent Chinese characters, 26 English letters (upper and lower case) and 10 digits.

(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 5 Reconstructed character trajectories ((a)(c)(e)(g)) and filtered results ((b)(d)(f)(h)).

5. EXPERIMENTAL RESULTS AND DISCUSSIONS

5.1. Experiments on DSB-MM segmentation

We manually marked out hand contours of 1,828 frames and regarded them as ground truth. Two rates, indicating percentage of right segmentation (*TP*/(*TP*+*FN*)) and over segmentation (*FP*/(*TP*+*FN*)), are applied to quantitatively measure the segmentation accuracy where *TP*, *FN* and *FP* stand for the true positive, false negative and false positive. As shown in Table. 1, the right segmentation rate of [3] is the lowest because this kind of vision-based method may suffer from the hand-face overlapping problem. Our model reaches very nice performance as it does not merely use unreliable depth information like [4], and overcomes the color-depth mismatch problem which affects [5].

Table. 1 Comparison of hand segmentation methods.

Model	*Right-Seg* *TP*/(*TP*+*FN*)	*Over-Seg* *FP*/(*TP*+*FN*)
Skin + motion cues [3]	69.57%	14.23%
Depth [4]	84.20%	39.48%
Skin + depth [5]	80.84%	**7.99%**
DSB-MM	**88.85%**	10.02%

5.2. Experiments on fingertip detection

We have collected over 3,027 video frames and manually labeled the position of fingertip on each frame. We calculate the Euclidean distance between real and detected fingertip position as error distance. Error within a specified distance is treated as a correct detection. As shown in Table. 2, our method obviously achieves the highest detection accuracy in both cases. Template matching [1] and maximum curvature [6] methods heavily rely on the visual shape features; while our model considers finger-hand-arm physical constraints. Also these two methods cannot handle certain extreme poses like fingertip pointing directly to the sensor. 3D clustering [10] may suffers from noise and data missing.

Table. 2 Experimental results of fingertip detection.

Method		Template matching [1]	Maximum curvature [6]	3D clustering [10]	Proposed method
Error distance (pixels)	≤10	79.48%	78.10%	90.63%	**94.19%**
	≤5	55.24%	52.96%	60.32%	**75.78%**

5.3. Experiments on character recognition

Character recognition experiment is conducted on 375 videos. For the first five candidates, we can reach the accuracy rate of 90.77%, 99.23%, 98.46% and 100% for Chinese, upper case English, lower case English and digit respectively. The system is tested using PC with Intel®Core™ i5-2400 CPU, 3.10GHz at 20fps.

6. CONCLUSION

A novel finger-writing-in-the-air system is proposed which allows user to write freely in the air with fingertip. Experiments show the superior performance and great potential on a natural and user-friendly writing experience.

ACKNOWLEDGEMENT

This research is supported in part by the National Natural Science Foundation of China (Grant No: 61075021), MSRA Research Fund (No. FY12-RES-THEME-67), the National Science and Technology Support Plan (2013BAH65F01-2013BAH65F04), the National Science Foundation (No. 61202292), the Guangdong Natural Science Funds (Grant No: S2011020000541), the GDSTP (No. 2012A010701001), and the Fundamental Research Funds for the Central Universities of China (no.2012ZP0002, D2116320, 2012ZM0022).

REFERENCES

[1] L. Jin, D. Yang, L. Zhen, and J. Huang. A novel vision based finger-writing character recognition system. *Journal of JCSC*, 16(3):421–436, 2007.

[2] S. L. Phung, A. Bouzerdoum, and D. Chai. Skin segmentation using color pixel classification: Analysis and comparison. *IEEE Trans. on PAMI*, 27:148–154, 2005.

[3] Jonathan Alon, Vassilis Athitsos, Quan Yuan and Stan Sclaroff. A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation. *IEEE Trans. on PAMI*, 31:1685–1699, 2009.

[4] X. Liu and Kikuo Fujimura. Hand Gesture Recognition using Depth Data. In *Proc. of IEEE FGR,* 2004.

[5] Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proc. of ACM Multimedia*, 2011.

[6] D. Lee and S. Lee. Vision-based finger action recognition by angle detection and contour analysis. *Journal of ETRI*, 33(3):415–422, 2011.

[7] Martin de La Gorcel, Nikos Paragios and David J. Fleet. Model-Based Hand Tracking with Texture, Shading and Self-occlusions. In *Proc. of IEEE CVPR, 2008.*

[8] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis. Real time foreground-background segmentation using code book model. *Real-Time Imaging*, 11:172–185, 2005.

[9] T. Long and L. Jin. Building Compact MQDF Classifier for Large Character Set Recognition by Subspace Distribution Sharing. *Pattern Recognition*, 41(9):2916-2926, 2008.

[10] Ziyong Feng, Shaojie Xu, Xin Zhang, Lianwen Jin, Zhichao Ye and WeixinYang. Real-time Fingertip Tracking and Detection using Kinect Depth Sensor for a New Writing-in-the Air System. In *Proc. of IEEE ICIMCS, 2012.*