Wenjun Zeng
University of Missouri, zengw@missouri.edu

A New Writing Experience: Finger Writing in the Air Using a Kinect Sensor

Xin Zhang, Zhichao Ye, Lianwen Jin, Ziyong Feng, and Shaojie Xu
South China University of Technology

With the introduction of Microsoft Kinect, there has been considerable interest in creating various attractive and feasible applications in related research fields. Kinect simultaneously captures depth and color information and provides real-time, reliable 3D full-body human-pose reconstruction that essentially turns the human body into a controller.[1] Kinect has opened a new era for more advanced and natural human-computer interaction (HCI), and many exciting applications, from gaming to the medical field, have been developed. In this article, we present a finger-writing system that recognizes characters written in the air without the need for an extra handheld device. This application would allow for brand-new natural user interaction (NUI) experiences, especially for remote applications.

It is believed that HCI is becoming increasingly similar to the interaction among people. A traditional keyboard and mouse is the most common way to "talk with" machines. In recent years, more advanced remote controllers (such as the Nintendo Wii remote) and touchscreens have been widely used and enjoyed by users. However, a handheld device is still needed. We propose using a hand to write in the air by treating the fingertip as a virtual pen. Using Kinect system, users can input characters by moving their hands, enjoying a full-body-controlled experience. The remote input could enable several real-world services such as remote signatures. "Writing in the air with hands" can serve as a fun way to teach young students how to write. The finger-writing-in-the-air system based on Kinect allows the user to write in the air in a natural, unconstrained way that might be an essential component for the next generation of HCI.

By incorporating depth and visual information, we can directly track the hand-finger trajectory and recognize characters written in the air in real time. The first step is to segment the hand from the cluttered background by combining color and depth sequences. A depth-skin-background mixture model (DSB-MM) is proposed for hand segmentation to solve traditional problems associated with Kinect such as illumination variation, hand-face overlapping issues, and color-depth mismatch. Second, a dual-mode switching algorithm is used to accurately detect the fingertip from various hand poses. The trajectory of the fingertip is extracted and linked and then reconstructed as an inkless character. A state-of-the-art handwriting character recognition method is employed to generate the final output. Figure 1 shows the framework of our system. This article describes each component of the system in more detail.

Hand Segmentation Using Depth and Color Sequences

Hand segmentation is usually the first step of a hand-based application, and it directly affects the performance of the following procedures. A skin-based color model has been widely used because of the distinguishable color differences between a hand and the background.[2] This type of color model faces serious challenges in certain conditions such as when the illumination varies, in a cluttered environment, and when the hand and face overlap.

Editor's Note

The Kinect effect is transforming human-machine interaction in multiple industries. "Writing in the air with hands" is one such exciting example. This article unravels the enabling technologies behind this promising finger-writing system.

1070-986X/13/$31.00 © 2013 IEEE

Published by the IEEE Computer Society

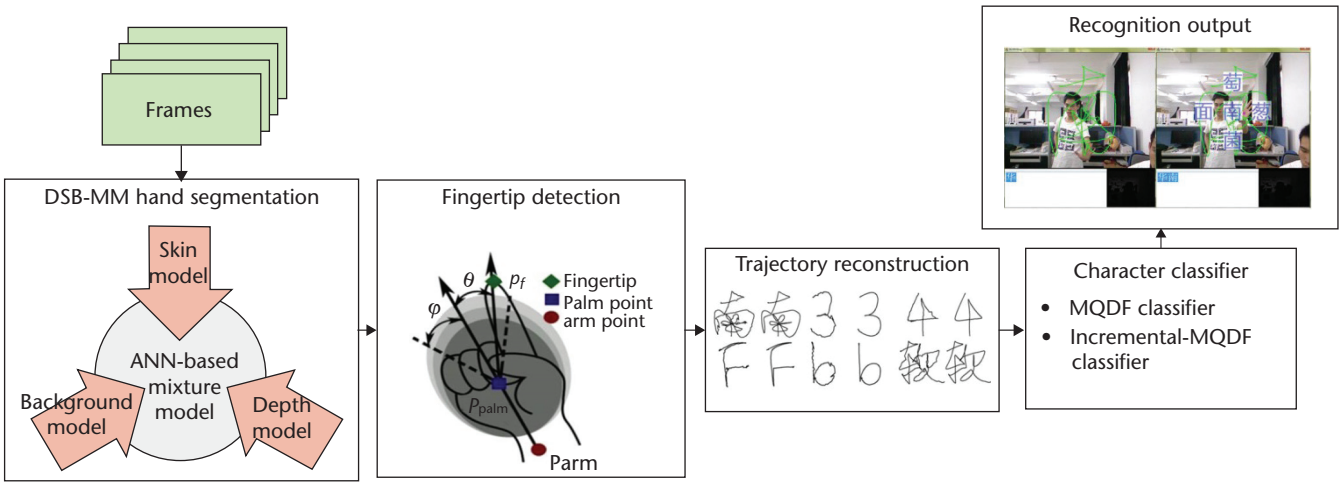

Figure 1. Framework of the finger-writing-in-the-air system.

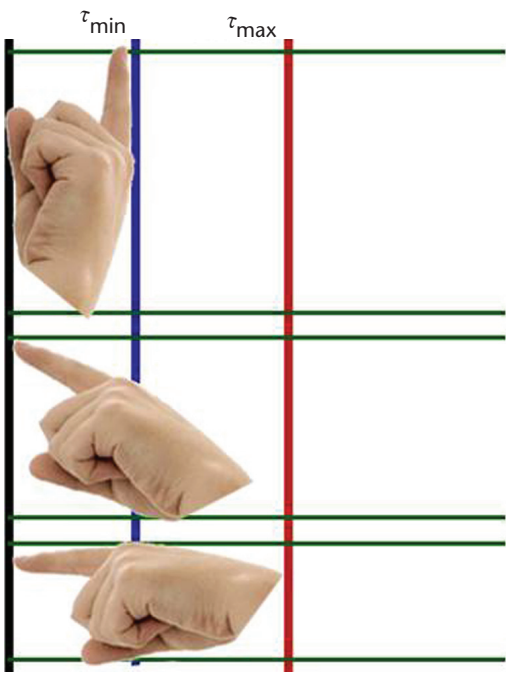

Figure 2. Relationship between the hand depth and the segmented region.

The frame-difference-based motion-cue method can also be applied to localize the hand.[3,4] This method strongly relies on the assumption that the hand should be the most distinct moving object, which limits its application.

Although the depth information provided by Kinect can account for background variation and color similarity issues, directly and solely applying a depth sequence does not return satisfactory results because of its low resolution and strong noise. Additionally, the color-depth nonsynchronization problem is a new challenge because color and depth sequences are not recorded and updated at the exact same time by Kinect. Therefore, we propose a depth-skin-background mixture model (DSB-MM) for fast and accurate hand segmentation.

Depth Model

The depth information provided by Kinect is a 640×480 grayscale image that encodes the distance of the scene object surfaces from the Kinect's viewpoint. With the 3D distance information, the depth image can partially solve the typical issues of the appearance-based segmentation model, such as color similarity, lighting variations, and a moving background.

The basic assumption of this depth-based segmentation model is that the hand is the closest part of the human body to the sensor during writing. Given a depth image, we first employ a "user map" offered by Kinect to extract a human body from the background. Second, we carry out preprocessing by applying morphology operations such as erode and dilate operations to remove the noise. The third step is to find the smallest depth value $d_{\min}$ within the body region, and a depth-based hand mask D is defined accordingly by applying an adaptive threshold, which is the key contribution in this work.

Because the hand volume is fixed, an inverse proportion exists between the hand depth τ_d and the segmentation hand region R during the hand-pose variations. As Figure 2 shows, a larger value of τ_d should have a smaller segmented hand region and vice versa. Assuming $R(n)$ is the hand region at frame n, we use $\tau_d(n)$

as the threshold for the segmentation of frame $(n+1)$ and obtain the corresponding region $R(n+1)'$. τ_d is updated by

$$\tau_d(n+1) = \tau_d(n) + \left(\frac{R(n)}{R(n+1)'} - 1\right)\omega$$

where ω is the growth factor. τ_d should change within the range $[\tau_{\min}, \tau_{\max}]$, which is determined by experiments and statistics. Hence, the segmented result of the depth model at frame $(n+1)$ is then updated as $D(n+1)$ with the region $R(n+1)$ by applying $\tau_d(n+1)$ for resegmentation.

The depth model can roughly identify the hand position but not accurately determine the segmented region and clear edges. As Figure 3a shows, the depth-based segmented hand region has noisy edges. It also contains a large falsely segmented area for a moving hand in Figure 3b because of the color-depth nonsynchronization issue. Hence, other models are necessary.

(a) (b)

Figure 3. Zoomed-in depth-model segmentation result superimposed onto a color image. (a) A static hand and (b) a moving hand.

Skin Model

The skin model serves an important role in a range of hand-related research. Hence, we build a robust skin model characterizing both skin and nonskin distributions as single Gaussian distributions in the YCbCr color space. We have designed two strategies to save storage space and computational load. First, instead of building a skin model directly in 3D space, we quantify the Y component into three regions: bright $(170 \leq Y \leq 255)$, normal $(85 \leq Y \leq 169)$, and dark $(0 \leq Y \leq 84)$. Second, Mahalanobis-distance-based lookup tables are generated for the skin and nonskin models to reduce the computational load. The file size of the lookup table for the YCbCr space is 256 Mbytes, but our quantified Y-component-based CbCr tables is only 2 Mbytes.

As we discussed earlier, the combined depth-skin model helps to remove extra background pixels from the depth-model segmentation result and skin-like pixels from the skin-model segmentation result. However, the nonsynchronization issue in the color-depth sequences captured by Kinect leads to irreversible and incorrect segmentation, as shown in Figure 4. Note the thumb is missing in Figure 4d.

Background Model

The color-depth mismatch problem we mentioned earlier occurs because two images that are not recorded at the exact same time by Kinect—that is, they do not represent the same scene. Because the missing pixels belong to the moving foreground object, we believe a background model could help. We apply a codebook background model proposed in previous work[5,6] that aims to create a statistical model for the background and detect the object of interest as the foreground. In our case, we only consider the Cb and Cr channels to save computational load and memory storage. Instead of updating all the pixels that are recognized as the background, we only update the pixels detected as the nonhand region by the DSB-MM, which will be introduced in the next subsection. This target-oriented codebook model can immediately capture changes in the background scene and avoid the hysteresis phenomenon.[6] When the hand remains motionless for a long period of time, the pixels of the hand won't be "absorbed" or incorrectly learned as a new object.

Figure 5 illustrates some results using the codebook model. Figures 5c and 5d show that part of the hand is missing when it is around the face because the face color information has been identified as part of the background during previous frames. Hence, it is not enough to only use a background model.

Generally speaking, the depth, skin, and background models have their own advantages and limitations. The depth model is robust to illumination variation and skin similarity but is greatly influenced by strong noise, particularly along the hand boundary and in certain poses. Skin-model-based hand segmentation has been widely used, and some of its limitations can be successfully removed by the depth model. However, the depth-skin model combination fails when the depth and color sequences are mismatched because of hardware limitations.

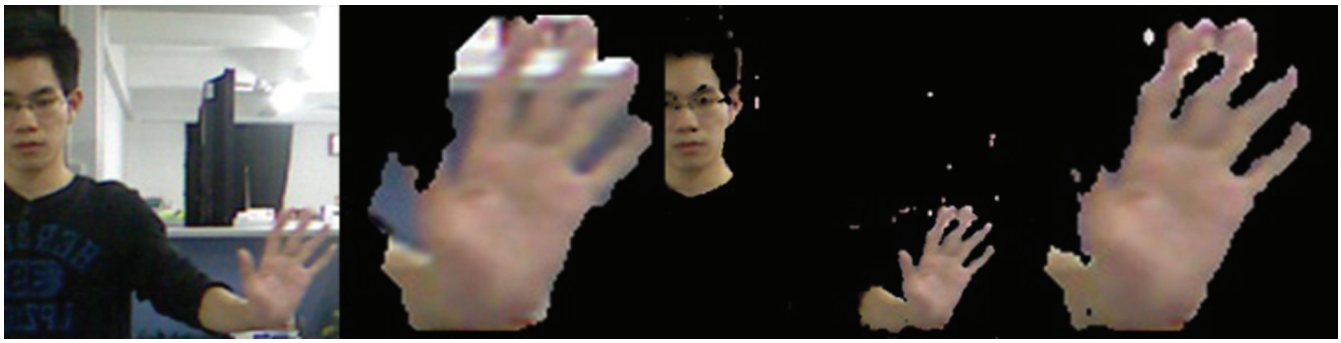

Figure 4. Color depth-model segmentation result. (a) Color image, (b) zoomed-in depth model result, (c) skin-model result, and (d) zoomed-in combination of the depth- and skin-model results.

Figure 5. Background-model segmentation result. (a) Color image and (b) and foreground codebook model result. Performance decreases (c) when the hand overlaps with the face image, (d) as the foreground codebook model result shows.

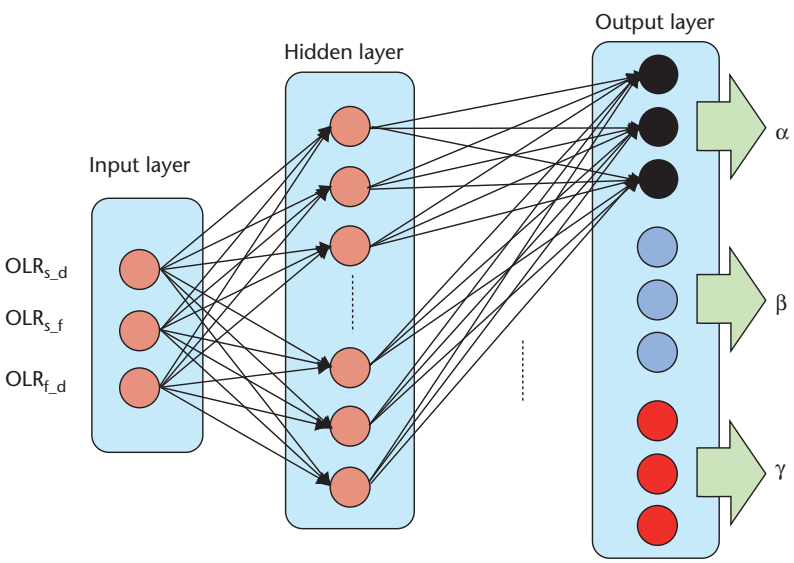

Figure 6. Architecture of the artificial neural network (ANN).

Hence, we introduce a background model to account for the depth-color nonsynchronization and noisy segmentation contour. The key issue here is how to merge the three models for the best final segmentation result.

DSB-MM Segmentation

The depth-skin-background mixture model (DSB-MM) combines the depth, skin, and background models to address their individual advantages and limitations. The DSB-MM functions similarly to an expert voting system. Instead of simply voting "yes" (the pixel of the segmentation result has value of 1) or "no" (value 0), we make the DSB-MM more adaptive because each model should have different reliabilities in different circumstances.

We present an artificial neural network (ANN) that contains three layers with three inputs, nine outputs (three as a group that determine which energy factor is chosen for each model), and a hidden layer including 30 neurons, as illustrated in Figure 6. The ANN is trained with the resilient back propagation algorithm (RPROP) with a sigmoid function as the activation function. The inputs of the ANN are three overlapping rates (OLRs) (skin versus depth, skin versus background, and background versus depth) that measure the consistency between the segmentation results of the two models. The outputs of the ANN are the

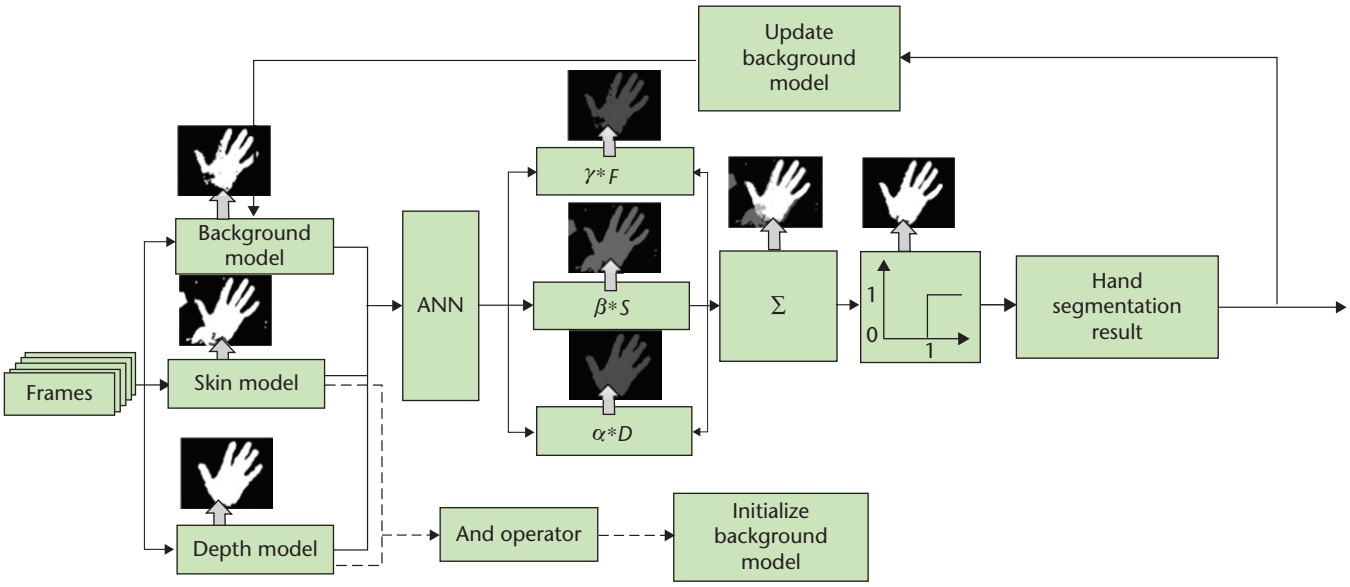

Figure 7. Flowchart of the DSB-MM segmentation algorithm.

confidence factors of the three models α, β, and γ. During the training process, the model that has a larger OLR with the other two models is assigned a higher confidence factor. There are two assumptions used in the ANN design:

- All the models contribute to the final result, which means that every model has a confidence factor larger than 0.
- None of the models is absolutely reliable (the confidence factor cannot be 1), which leads to the conclusion that a pixel finally treated as hand must be detected as a target (hand) by at least two models.

The final results are determined by the linear combination of these three models; the models' weights are their own confidence factors (having values from 0 to 1) provided by the ANN. The pixel with a combined value equal to or greater than 1 is considered a hand pixel.

Figure 7 shows the flowchart for DSB-MM segmentation. First, the background, skin, and depth models are employed to segment the hand regions individually. The overlapping rates of every two models are computed as the inputs of the trained ANN model. The outputs of the ANN are the three confidence factors, which are weights of the linear combination of the three models. The background model is initialized by the depth-skin model result of the first N frames (we set N to 15). In the following frames, the background model is updated using the nonhand region determined by the DSB-MM to avoid the hysteresis phenomenon.

Figure 8 illustrates the results. We show the original images and superimposed results of the depth model, skin model, background model, and final DSB-MM. The figure clearly shows that none of the models alone can outperform the proposed model. The depth model results have larger and mismatched regions, while the skin model results always include the face region and unwanted background. Furthermore, the background model has serious difficulties when attempting to distinguish a hand that is near the face. The proposed DSB-MM obtains the best performance.

Fingertip Detection

After hand segmentation, a fingertip detection algorithm is applied to track the writing trajectory. We propose a dual-mode (side and frontal) switching algorithm, which covers all the possible hand poses during writing with different fingertip-detection approaches.

The side mode indicates that the finger is not pointing toward the camera, as shown in the outer circle in Figure 9. Although the finger is usually distinguishable in the segmented 2D region in this mode, previous vision-based fingertip detection algorithms such as local maximum curvature[7] and template matching[8] were sensitive to the hand-pose orientation and segmentation noise. Hence, we assume the fingertip is the farthest point from the arm point in the segmented hand region when considering

Figure 8. Segmentation results of the three single models and proposed DSB-MM. (a) Original images, (b) depth model, (c) skin model, (d) background model, and (e) mixture model.

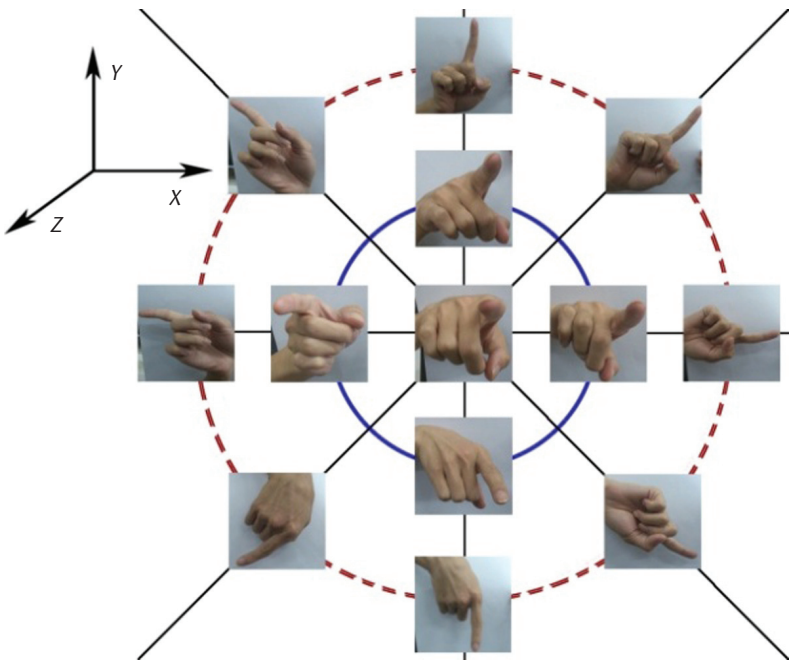

Figure 9. Various hand poses for writing. The outer dashed circle illustrates the poses of the side mode, and the inner circle illustrates the poses of the frontal mode.

the depth information and the physical relationship of the finger, hand, and arm.

The frontal mode indicates that the finger is almost pointing at the camera, as shown in the inner circle in Figure 9. The frontal mode can handle a set of specific poses when visual cues are invalid. In this mode, the fingertip may not always be the farthest point from the arm in the segmented area, but it is definitely the nearest point in the hand region to the camera—that is, the point with the smallest depth value.

Based on our analysis and experimental observations, the hand pose belongs to the side mode if the following two criteria are satisfied. First, intuitively, the farthest point p_f from the arm in the segmented 2D hand region should not be in the palm area. As Figure 10 shows, the palm area is obtained by applying an ellipse-fitting technique for three iterations, and the central point of the final ellipse is regarded as the palm point. To determine the arm point, we increase the depth threshold in the depth model, and the newly included pixels belong to the arm. The arm point can be located by calculating the center of the increased region.

Second, considering the physical limitations of a human, the angle θ between the hand direction (the line connecting palm and arm points) and the finger direction (the line connecting p_f and the palm point) is less than a certain value (we set it to 30°). If the hand pose belongs to the side mode, the fingertip is p_f.

Otherwise, we switch to the frontal mode. From our experiments, we notice that there is a black area around the fingertip in the depth image because the infrared light is scattered by the fingertip and depth values are incorrectly set to 0. We employ the inpainting technique[9] to fill the hole using nearby pixels. The fingertip is the point with the smallest depth value of the recovered hand region.

We use one finger-writing sequence to demonstrate mode switching between the frontal and side modes in Figure 11. Additionally, the effectiveness of the proposed algorithm is illustrated by showing the frame-wise pixel distance between the detected and manually marked fingertips. The side-mode-only plot (red line) significantly increases when the finger is pointing directly at the camera (frames 40 to 60). On the other hand, the frontal-mode-only plot (blue line) has large errors when the finger is pointing to the side (frames 110 to 120). Obviously, our method can intelligently choose the proper mode for different hand poses to attain the best result.

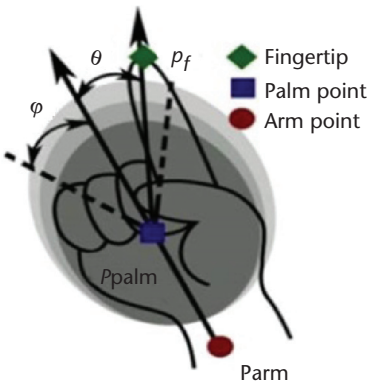

Figure 10. Physical model of a hand for fingertip detection.

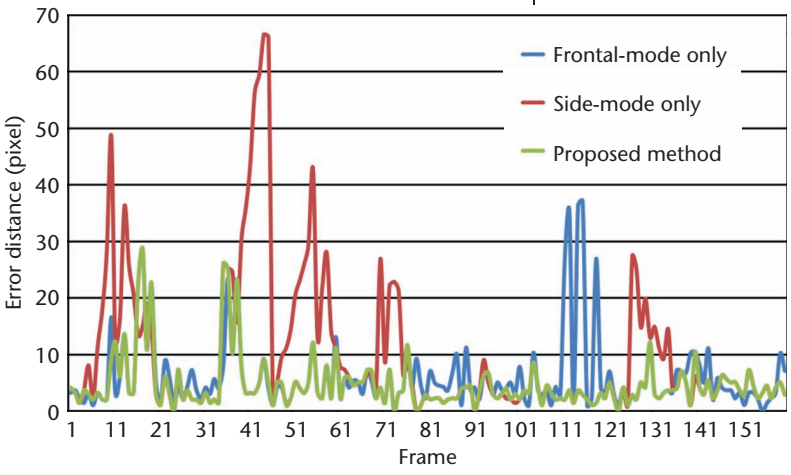

Figure 11. Error distance plots of one video sequence using the frontal-mode-only method, side-mode-only method, and proposed dual-mode switching algorithm.

Finger-Writing Character Recognition

The finger-writing trajectory is generated by linking all detected fingertip positions from continuous frames together, as shown in Figure 12. The linked trajectory is then passed into a mean filter to remove noise and jitter caused by incorrect fingertip detection and larger hand movement. Figure 13 shows some examples of reconstructed written trajectories and the smoothed filtered results.

We use a compact modified quadratic discriminant function (MQDF) character classifier[10] for finger-writing trajectory recognition. After extracting the modified eight-direction features of 1,024 dimensions, the original feature is reduced to 160 dimensions by linear discriminant analysis (LDA) and recognized by an MQDF classifier, which outputs the final result. The classifier can recognize 6,763 frequently used Chinese characters, 26 English letters (both uppercase and lowercase), and 10 digits.

A preliminary character recognition experiment was conducted on 375 videos with a total of 44,522 frames. We successfully recognized 6,763 frequent Chinese characters, all English characters (lowercase and uppercases), and all digits. As summarized in Table 1, we achieved an accuracy rate of more than 90 percent for the first five candidates. The recognition rate for Chinese characters is slightly lower because it usually contains a more complex structure. The finger-writing-in-the-air system was tested using a PC with an Intel Core i5-2400 CPU running at 3.10 GHz and 4 Gbytes of RAM at 20 frames per second (fps). In general, our system achieves satisfactory and promising results for real-time applications.

Conclusion

Using a new ANN-based DSB-MM for hand segmentation and a dual-mode switching algorithm that can deal with all possible hand poses for writing, the proposed state-of-art handwriting character recognition method

Figure 12. Examples of finger-writing trajectories for different characters.

Figure 13. Reconstructed written trajectories. (a)–(c) Each set of characters shows first the reconstructed trajectories and then the filtered results.

Table 1. Recognition rates of characters.

	Within the first *N* candidate characters (%)		
Character	***N* = 1**	***N* = 3**	***N* = 5**
Chinese	78.46	89.23	90.77
Uppercase English letter	94.62	98.46	99.23
Lowercase English letter	86.15	96.92	98.46
Digit	92.00	100	100

facilitates inkless character recognition. Our experiments show that the user can write freely in the air, and the real-time recognition rates for the input of Chinese characters, English letters (upper and lower case), and digits are all greater than 90 percent for the first five candidates. In the future, we plan to collect more written trajectories to train an incremental character recognition model and design a few intuitive hand gestures for the system control and interaction. **MM**

Acknowledgments

This work is sponsored in part by Microsoft Research Asia Kinect Theme Fund (No. FY12-RES-THEME-067), by the National Natural Science Foundation of China (Grant No. 61075021 and No. 61202292), by the National Science and Technology Support Plan (No. 2013BAH65F01-2013BAH65F04), by Doctoral Fund of Ministry of Education of China (No. 20120172120026), by Foundation of Guangdong Key Lab of DSIP (No. 201201), and by Fundamental Research Funds for the Central Universities No. 2012ZM0022.

References

1. Z. Zhang, "Microsoft Kinect Sensor and its Effect," *IEEE MultiMedia,* vol. 19, no. 2, 2012, pp. 4–10.
2. P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A Survey of Skin-Color Modeling and Detection Methods," *Pattern Recognition,* vol. 40, 2007, pp. 1106–1122.
3. J. Alon et al., "A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no.9, 2009, pp. 1685–1699.
4. C. Shan, T. Tan, and Y. Wei, "Real-Time Hand Tracking Using a Mean Shift Embedded Particle

Filter," *Pattern Recognition,* vol. 40, no.7, 2007, pp. 1958–1970.
5. G.R. Bradski and A. Kaehler *Learning OpenCV,* O'Reilly Media, 2008.
6. K. Kim et al., "Real Time Foreground-Background Segmentation Using Code Book Model," *Real-Time Imaging,* vol. 11, no.3, 2005, pp. 172–185.
7. D. Lee and S. Lee, "Vision-Based Finger Action Recognition by Angle Detection and Contour Analysis," *Electronics and Telecomm. Research Inst. J.,* vol. 33, no. 3, 2011, pp. 415–422.
8. L. Jin et al., "A Novel Vision Based Finger-Writing Character Recognition System," *J. Circuits, Systems, and Computers* (JCSC), vol. 16, no. 3, 2007, pp. 421–436.
9. A. Telea, "An Image Inpainting Technique Based on the Fast Marching Method," *Proc. J. Graphics Tools,* vol. 9, no. 1, 2004, pp. 25–36.
10. T. Long and L. Jin, "Building Compact MQDF Classifier for Large Character Set Recognition by Subspace Distribution Sharing," *Pattern Recognition,* vol. 41, no. 9, 2008, pp. 2916–2926.

Xin Zhang is a lecturer in the School of Electronic and Information Engineering at the South China University of Technology. Contact her at eexinzhang@scut.edu.cn.

Zhichao Ye is a graduate student in the School of Electronic and Information Engineering at the South China University of Technology. Contact him at dante.ye.2011@gmail.com.

Lianwen Jin (corresponding author) is a professor in the School of Electronic and Information Engineering at the South China University of Technology. Contact him at eelwjin@scut.edu.cn.

Ziyong Feng is a PhD student in the School of Electronic and Information Engineering at the South China University of Technology.

Shaojie Xu is a graduate student in the School of Electronic and Information Engineering at the South China University of Technology.