Journal of Computer and Communications, 2016, 4, 160-168
Published Online April 2016 in SciRes. http://www.scirp.org/journal/jcc
http://dx.doi.org/10.4236/jcc.2016.44014

A Vision-Based Fingertip-Writing Character Recognition System

Ching-Long Shih[1]*, Wen-Yo Lee[2], Yu-Te Ku[1]

[1]Department of Electrical Engineering, National Taiwan University of Science and Technology, Taiwan
[2]Department of Computer Network and Technology, Lunghwa University of Science and Technology, Taiwan
Email: *shihcl@mail.ntust.edu.tw, TristanWYLee@mail.lhu.edu.tw

Received 2 March 2016; accepted 2016

Abstract

This paper presents a vision-based fingertip-writing character recognition system. The overall system is implemented through a CMOS image camera on a FPGA chip. A blue cover is mounted on the top of a finger to simplify fingertip detection and to enhance recognition accuracy. For each character stroke, 8 sample points (including start and end points) are recorded. 7 tangent angles between consecutive sampled points are also recorded as features. In addition, 3 features angles are extracted: angles of the triangle consisting of the start point, end point and average point of all (8 total) sampled points. According to these key feature angles, a simple template matching K-nearest-neighbor classifier is applied to distinguish each character stroke. Experimental result showed that the system can successfully recognize fingertip-writing character strokes of digits and small lower case letter alphabets with an accuracy of almost 100%. Overall, the proposed fingertip-writing recognition system provides an easy-to-use and accurate visual character input method.

Keywords

Visual Character Recognition, Fingertip Detection, Template Matching, K-Nearest-Neighbor Classifier, FPGA

1. Introduction

As the demand for intelligent vision-based human and computer/machine interfaces continues to grow, more intuitive and cost-efficient alternatives to conventional human-machine interfaces such as mice, keyboards, and touch-screens are required. Examples of such interfaces are vision-based hand gesture recognition system and

*Corresponding author.

How to cite this paper: Shih, C.-L., Lee, W.-Y. and Ku, Y.-T. (2016) A Vision-Based Fingertip-Writing Character Recognition System. *Journal of Computer and Communications*, **4**, 160-168. http://dx.doi.org/10.4236/jcc.2016.44014

handwriting recognition systems [1]-[3]. In sign language recognition, hand gestures are interpreted as symbols and words. On-line handwriting recognition systems provide a natural, convenient, and touch-free interface for human-computer interaction. This opens the door to new wireless character-inputting methods. Human fingertip can be used as a pointer input interface in place of a pen or a mouse, making the interaction/interface more user-friendly. A fingertip writing recognition system has applications, such as virtual mouse, signature input device and application selector.

In general, a visual fingertip/handwriting recognition system consists of 3 modules: visual data acquisition, feature extraction, and handwriting recognition. Based on a user's body position, 3D trajectories of the human body, arm and/or hand can be reconstructed [4] [5]. An easier way for human hand or fingertip movement detection is based on image segmentation methods, such as skin color segmentation, background subtraction, etc. [6]-[8]. The connecting vectors between sampled points of the trajectory serve as features for the handwriting recognition system. The key-features selected are trajectory points, tangent angles, curvatures, etc. [8] [9]. Commonly used handwriting recognition techniques include template matching, DTW (dynamic time wrap) based KNN classifiers, hidden Markov models (HMM), neural networks, etc. [5] [7] [9] [10].

Most earlier handwriting or fingertip-writing character recognition interfaces are built on PC-based systems with image input from USB cameras or Microsoft Kinect RGB-D sensors. This work focuses on an entirely FPGA based implementation with input from a CMOS image sensor. The proposed system achieved recognition accuracy of 100% for digits and lower case alphabets, and it could deal with scaled finger motion. One requirement of the system is that a fingertip blue cover is mounted on the user's finger and is visible to the camera.

2. Main Approach

2.1. System Overview

The proposed vision-based fingertip-writing character recognition system, as shown in **Figure 1**, is built on the Altera DE2-115 FPGA development board. A LCD display with a camera module (VEEK_MT) is connected to the DE2-115. A fingertip blue cover is mounted on the top of the user's finger in order to simplify fingertip detection and to enhance the recognition accuracy. The main functional modules of this fingertip-writing character visual recognition system, as shown in **Figure 2**, consist of (1) fingertip detection, (2) fingertip tracking and recording, (3) character stroke feature angle extraction, and (4) template matching with a KNN classifier.

2.2. Character Stroke Feature Angles

For each fingertip stroke, 9 feature points are extracted. The stroke's starting point is denoted by (x_0, y_0), and the end point by (x_7, y_7). Another 6 curve feature points (x_i, y_i), $i = 1, 2, \cdots, 6$, are then uniformly sampled along the stroke curve between start to end. The last feature point, (x_c, y_c), is the average point of the above 8 sampled points,

$$(x_c, y_c) = \frac{1}{8}\left(\sum_{i=0}^{7} x_i, \sum_{i=0}^{7} y_i\right) \tag{1}$$

Each stroke curve is encoded with 10 key feature angles, as shown in **Figure 3**. The first seven feature angles, θ_i, $i = 1, 2, \cdots, 7$, are tangent angles of vectors between consecutive sample points,

$$\theta_i = \arctan 2\left(y_i - y_{i-1}, x_i - x_{i-1}\right), \ i = 1, 2, \cdots, 7. \tag{2}$$

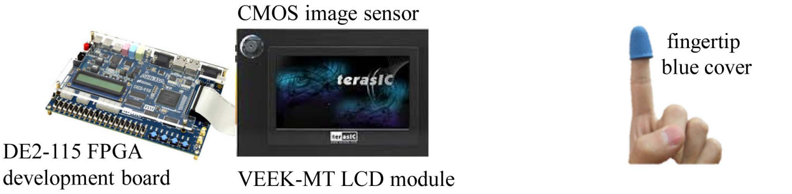


Figure 1. A fingertip writing character recognition system setup.

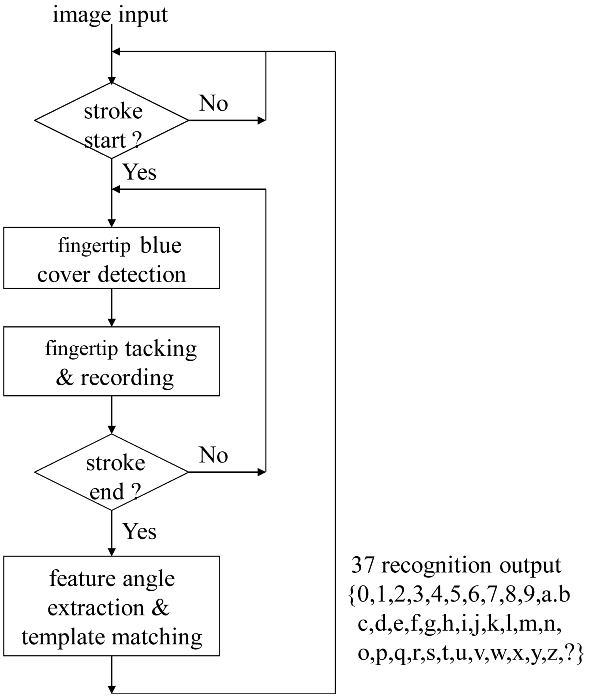


image input
stroke start ?
No
Yes
fingertip blue cover detection
fingertip tacking & recording
stroke end ?
No
Yes
feature angle extraction & template matching
37 recognition output
{0,1,2,3,4,5,6,7,8,9,a.b
c,d,e,f,g,h,i,j,k,l,m,n,
o,p,q,r,s,t,u,v,w,x,y,z,?}

Figure 2. Functional modules of the visual character recognition system: (1) fingertip detection, (2) fingertip tracking and recording, (3) character stroke feature angle extraction, and (4) template matching.

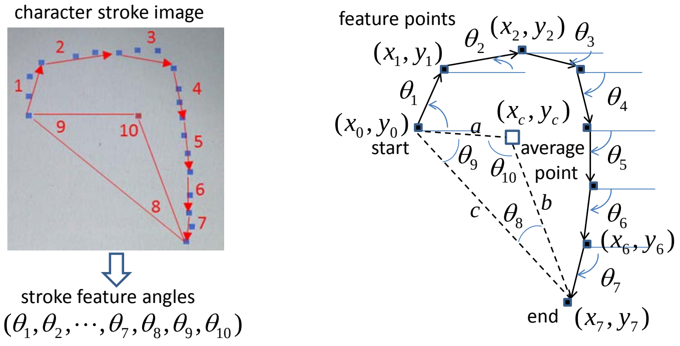


Figure 3. Illustration of a character stroke's 10 feature angles.

The function $\arctan 2(y, x)$ is a four quadrant arctangent function, such that

$$\arctan 2(y,x) = \begin{cases} \arctan\left(\dfrac{y}{x}\right) & y, x \geq 0 \\ \pi - \arctan\left(\dfrac{y}{-x}\right) & y > 0, x < 0 \\ \pi + \arctan\left(\dfrac{-y}{-x}\right) & y, x \leq 0 \\ 2\pi - \arctan\left(\dfrac{-y}{x}\right) & y < 0, x > 0 \end{cases} \tag{3}$$

The last three features angles $(\theta_8, \theta_9, \theta_{10})$ are angles of the triangle consisting of the start point (x_0, y_0), end point (x_7, y_7) and average point (x_c, y_c), and are obtained using the cosine theorem,

$$\theta_8 = \arccos\frac{b^2+c^2-a^2}{2bc} \tag{4}$$

$$\theta_9 = \arccos\frac{a^2+c^2-b^2}{2ac} \tag{5}$$

and

$$\theta_{10} = \arccos\frac{a^2+b^2-c^2}{2ab} \tag{6}$$

where (a,b,c) are edge lengths of the triangle.

2.2. Template Matching and Kth-Nearest-Neighbor Classifier

The proposed character recognition is based on template matching using a K-near-neighbor (KNN) classifier. Feature angles of reference characters (in the template *i*-th *j*-th feature angle table) are denoted by $\lambda_{i,j}$, $1 \le i \le 36$, $1 \le j \le 10$, where *i* stand for the *i*-th character in the alphabet sequence of $\{0,1,\cdots,9,a,b,\cdots,z\}$, as shown in **Appendix**. The match distance between input stroke, encoded with feature angles $(\theta_1,\theta_2,\cdots,\theta_{10})$, and each character template *i* is then computed by

$$d_i = \sum_{j=1}^{10} e_{ij}, \quad i = 1,\ 2,\cdots,36. \tag{7}$$

where

$$e_{ij} = \begin{cases} \left|\theta_j - \lambda_{i,j}\right| & \left|\theta_j - \lambda_{i,j}\right| \le 180^\circ \\ 360^\circ - \left|\theta_j - \lambda_{i,j}\right| & \left|\theta_j - \lambda_{i,j}\right| > 180^\circ \end{cases} \tag{8}$$

Let i^* denote the reference character satisfying $d_{i^*} = \min_{1 \le i \le 36}\{d_i\}$, then the KNN classifier output is

$$\text{output} = \begin{cases} i^* & d_{i^*} \le 400 \\ 0 & \text{otherwise} \end{cases}, \tag{9}$$

zero stands for an unknown character stroke.

3. Fingertip Detection and Tracking

The fingertip detection image processing module captures real-time images with a CMOS image sensor, image capture is followed by fingertip detection, character stroke tracking, and feature points recording. **Figure 4** shows the functional block diagram of fingertip detection image processing module. The processing pipeline includes color space transformation, histogram equalization, color detection, filtering, object tracking and recording.

The RGB to YCbCr color transform is defined as

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \tag{10}$$

where $0 \le R,G,B \le 255$ and $0 \le Y,Cb,Cr \le 255$. Color space transform is followed by histogram equalization for the Cb component. Let histogram density function of $r_k = Cb$ be denoted by

$$P_{rob}(r_k) = \frac{n_k}{MN}, \quad k = 0,1,2,\cdots,L-1, \tag{11}$$

where $M \times N$ is the total number of pixels, then the histogram equalization function is

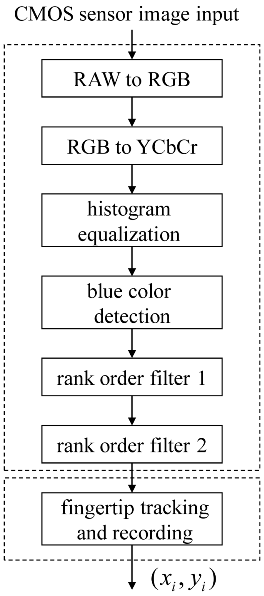


Figure 4. The image processing of fingertip detection and tracking.

$$s_k = T\left(r_k\right) = \left(L-1\right)\sum_{j=0}^{k} P_{rob}\left(r_j\right) = \frac{L-1}{MN}\sum_{j=0}^{k} n_j,\ \ k = 0,1,2,\cdots,L-1. \tag{12}$$

Then, the fingertip blue-cover color segmentation is performed as

$$P = \begin{cases} 1 & s_k > 252 \\ 0 & \text{otherwise} \end{cases}, \tag{13}$$

where s_k denotes the Cb component after histogram equalization. Finally, a rank order filter in a 5×5 window,

$$P_{13} = \begin{cases} 1 & \sum_{i=1}^{25} P_i = 25 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

is applied twice to generate binary image, where pixels in the 5×5 window are numbered from 1 to 25 from left to right and then top to bottom, with P_{13} being the center pixel. True pixels indicate potential fingertip positions. The detected fingertip position is recoded and tracked at a frequency of 7 times per seconds.

4. FPGA Implementation and Experiment

The proposed fingertip-writing visual character recognition system is implemented as a dedicated logic circuit on a FPGA chip. Real-time image input is feed to the FPGA chip line by line. Up to 5 rows of a image are stored in line-buffers (FIFO) in a pipelined fashion. The image processing pixel clock is 96 MHz. The raw image data is captured by a color camera with 800×480 resolution and a frame rate of 7 fps (frames per second). The raw image is converted to a 800×480 24-bit RGB image (by a local neighborhood of every four scanned pixels). **Figure 5** shows the image processing pipeline for fingertip detection and tracking.

Two mathematical functions, arccos(x) and arctan2(y,x), are required when computing a character stroke's feature angles. The arccosine function is implemented using a lookup-table with a resolution of 1.0 degrees. CORDIC algorithm [11] with 12-bit input data is applied to compute tangent angles. **Figure 6** shows the pipelined CORDIC design for computing first-quadrant inverse tangent on the FPGA chip. The four-quadrant inverse tangent function can then be obtained from Equation (3). The synthesis results for all character recognition system architectures are shown in **Table 1**. The overall percentages of total FPGA resource are enclosed in parenthesis.

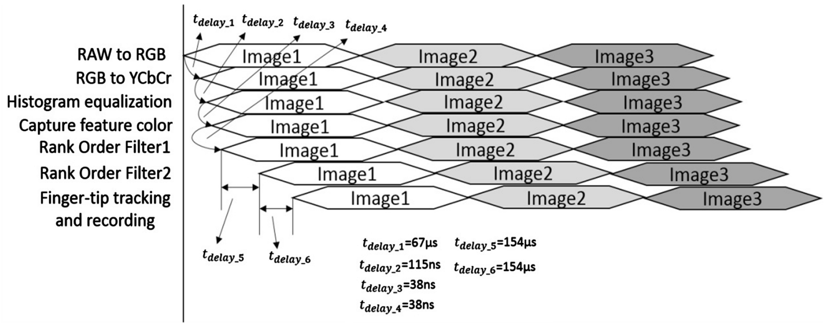


Figure 5. The image processing pipeline of fingertip detection and tracking.

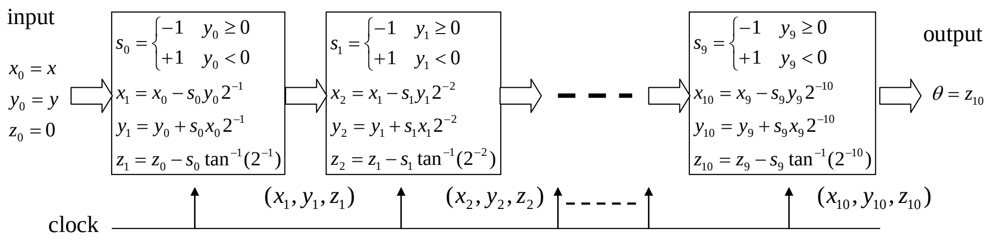

input
$x_0 = x$
$y_0 = y$
$z_0 = 0$
$s_0 = \begin{cases} -1 & y_0 \ge 0 \\ +1 & y_0 < 0 \end{cases}$
$x_1 = x_0 - s_0 y_0 2^{-1}$
$y_1 = y_0 + s_0 x_0 2^{-1}$
$z_1 = z_0 - s_0 \tan^{-1}(2^{-1})$
$s_1 = \begin{cases} -1 & y_1 \ge 0 \\ +1 & y_1 < 0 \end{cases}$
$x_2 = x_1 - s_1 y_1 2^{-2}$
$y_2 = y_1 + s_1 x_1 2^{-2}$
$z_2 = z_1 - s_1 \tan^{-1}(2^{-2})$
$s_9 = \begin{cases} -1 & y_9 \ge 0 \\ +1 & y_9 < 0 \end{cases}$
$x_{10} = x_9 - s_9 y_9 2^{-10}$
$y_{10} = y_9 + s_9 x_9 2^{-10}$
$z_{10} = z_9 - s_9 \tan^{-1}(2^{-10})$
output
$\theta = z_{10}$
clock
(x_1, y_1, z_1)
(x_2, y_2, z_2)
(x_{10}, y_{10}, z_{10})

Figure 6. Pipelined CORDIC design for computing first quadrant inverse tangent.

Table 1. Synthesis results and FPGA device utilization.

Cyclone IV EP4CE115F29	Logic elements	Registers	Memory bits
total elements	114,480 (bits)	114,480 (bits)	3,981,312 (bits)
used elements	43,773 (38%)	13,770 (12%)	55,804 (2%)

Functional module	Logic elements	Registers	Memory bits
FPGA used elements	43,773 (bits)	13,770 (bits)	55,804 (bits)
RAW to RGB	100 (<1%)	53 (<1%)	19,152 (34%)
RGB to YcbCr	104 (<1%)	53 (<1%)	0
histogram equalization	11,634 (27%)	9728 (70%)	0
blue color detection	15 (<1%)	0	0
rank order filter 1	82 (<1%)	37 (<1%)	3990 (7%)
rank order filter 2	82 (<1%)	37 (<1%)	3990 (7%)
fingertip tracking and recording	8987 (21%)	1505 (11%)	0
feature angle extractions	8580 (20%)	0	0
sdram control	991 (2%)	696 (5%)	28,672 (52%)
template matching	1172 (3%)	177 (1%)	0
others	12,026 (27%)	1484 (11%)	0

Figure 7 shows a typical fingertip detection and character recognition execution. The start of the fingertip writing process is triggered by moving the fingertip to a start button box shown on the LCD screen. The recognition system then starts to track and record a character stroke after 4 seconds. The end of a stroke is signaled by holding the fingertip stationary for 2 seconds. The recognition result is shown on the LCD screen thereafter. The entire process takes about a few seconds. **Figure 8** shows a 2-character writing recognition experiment, in which the recognition of the second character starts right after the first character is recognized by 3 seconds.

The following character stroke pairs are similar and are more difficult to distinguish, hence the requirement for special distinction mechanisms.

(1) Digit 0 and alphabet o: digit 0 is written clock-wise, and alphabet o counter clock-wise.

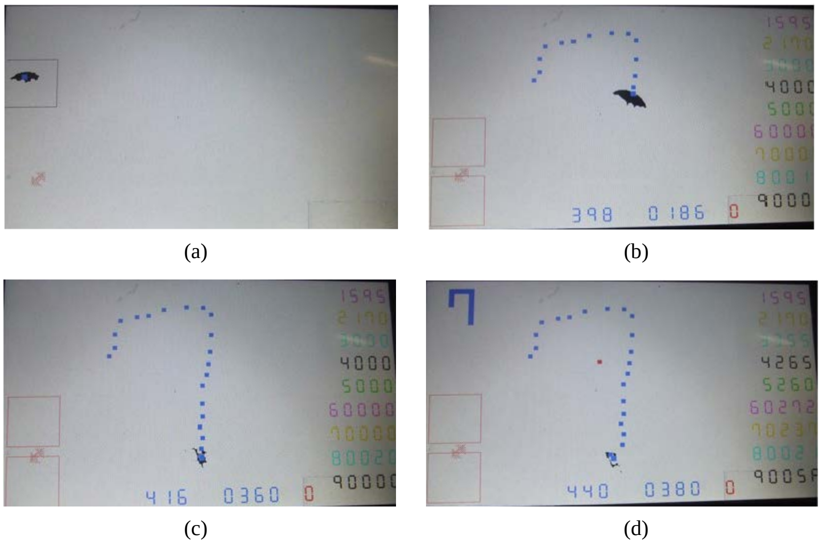
(2) Digit 2 and alphabet z: alphabet z is written with an additional up-tick stroke.

(3) Digit 5 and alphabet s: alphabet s is written with a smoother curvature.

(4) Digit 9 and alphabet g: alphabet g is written with an additional up-tick stroke.

(5) Alphabets f and t: alphabet f is finished with an left-up tick stroke.

Figure 9 shows the experimental sample character strokes of digits 0 - 9 and lower case alphabets a - z. The



(a)

(b)

(c)

(d)

Figure 7. Experimental fingertip writing and recognition process. (a) Start of fingertip writing; (b) Stroke curve tracking; (c) End of character stroke; (d) Character recognition result.

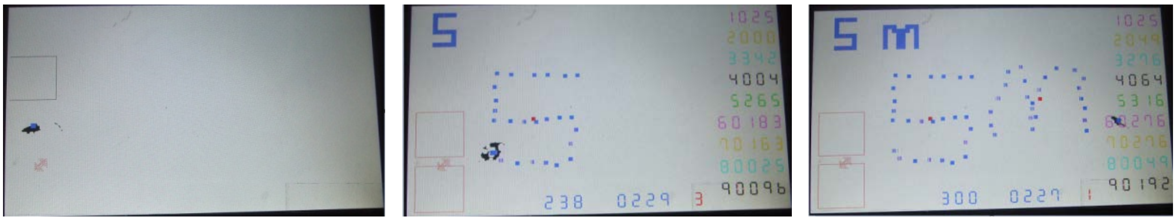


Figure 8. Experimental 2-character writing recognition.

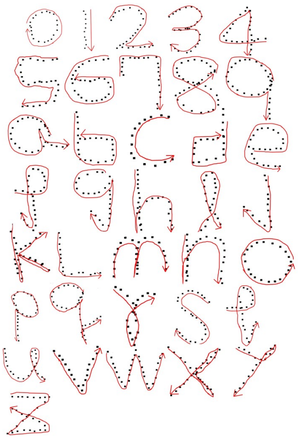


Figure 9. Experimental 36 character strokes of 0 - 9 and a - z.

proposed system demonstrated 100% recognition accuracy in our experiments. We tested each sample character 10 times (a total of 360 character strokes), and no recognition error occurred.

5. Conclusion

We have presented a simple but effective vision-based fingertip writing character stroke recognition system. The proposed system is implemented on a single FPGA chip with input from a CMOS image sensor. The recognition system achieved accuracy 100% for digits and lower case alphabets, and it could deal with scaled finger motion. One additional requirement of the system is that a blue cover is mounted on the user's fingertip and is visible to the camera. In future work, we would like to perform recognition of words with a few characters.

Acknowledgements

This work is supported by Taiwan Ministry of Science and Technology grants MOST-103-2221-E-011-101 and MOST 104-2221-E-011-035.

References

[1] Jin, L., Yang, D., Zhen, L. and Huang, J. (2007) A Novel Vision Based Finger-Writing Character Recognition System. *Journal of Journal of Circuits, Systems, and Computers*, **16**, 421-436. http://dx.doi.org/10.1142/S0218126607003757

[2] Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H. and Presti, P. (2011) American Sign Language Recognition with the Kinect. *ICMI*'11 *Proceedings of the* 13*th International Conference on Multimodal Interfaces*, 279-286. http://dx.doi.org/10.1145/2070481.2070532

[3] Mitra, S. and Acharya, T. (2007) Gesture Recognition: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics*, *Part C*: *Applications and Reviews*, **37**, 311-324. http://dx.doi.org/10.1109/TSMCC.2007.893280

[4] Schick, A., van de Camp, F., Ijsselmuiden, J. and Stiefelhagen, R. (2009) Extending Touch: Towards Interaction with Large-Scale Surfaces. *Proceedings of ACM International Conference on Interactive Tabletops and Surfaces*, 117-124. http://dx.doi.org/10.1145/1731903.1731927

[5] Phung, S.L., Bouzerdoum, A. and Chai, D. (2005) Skin Segmentation Using Color Pixel Classification: Analysis and Comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligent*, **27**, 148-154. http://dx.doi.org/10.1109/TPAMI.2005.17

[6] Schick, A., Morlock, D., Amma, C., Schultz, T. and Stiefelhagen, R. (2012) Vision-Based Handwriting Recognition for Unrestricted Text Input in Mid-Air. *ICMI*'12 *Proceedings of the* 14*th International Conference on Multimodal Interfaces*, Santa Monica, October 2012, 217-220. http://dx.doi.org/10.1145/2388676.2388719

[7] Ye, Z., Zhang, X., Jin, L., Feng, Z. and Xu, S. (2013) Finger-Writing-in-the-Air System Using Kinect Sensor. *IEEE International Conference on Multimedia and Expo Workshops* (*ICMEW*), San Jose, July 2013, 1-4.

[8] Lee, D. and Lee, S. (2011) Vision-Based Finger Action Recognition by Angle Detection and Contour Analysis. *Journal of ETRI*, **33**, 415-422. http://dx.doi.org/10.4218/etrij.11.0110.0313

[9] Vikram, S., Li, L. and Russel, S. (2013) Handwriting and Gestures in the Air, Recognizing on the Fly. *Proceedings of the CHI*'13, Paris, April 2013, 1179-1184.

[10] Plamondon, R. and Srihari, S.N. (2000) On-Line and Off-Line Handwriting Recognition: a Comprehensive Survey. *Transactions on Pattern Analysis*, **22**, 63-84.

[11] Hu, Y.H. (1992) CORDIC-Based VLSI Architectures Digital Signal Processing. *IEEE Signal Processing Magazine*, **9**, 16-35. http://dx.doi.org/10.1109/79.143467

Appendix

Table 2. Template of character stroke feature angles.

$\lambda_{i,j}$	1	2	3	4	5	6	7	8	9	10
0	50	343	292	237	191	147	85	87	91	0
1	274	261	275	262	265	276	261	37	37	180
2	43	359	290	232	212	2	4	26	39	151
3	25	309	205	11	304	221	140	46	46	99
4	179	183	50	42	302	274	265	44	74	64
5	182	232	334	343	264	175	175	45	29	160
6	181	232	292	334	57	161	180	57	22	130
7	83	0	309	265	276	219	288	39	44	118
8	150	245	320	244	129	39	43	104	77	7
9	137	212	302	25	293	265	281	19	66	109
a	163	234	295	29	85	295	4	33	85	69
b	262	267	281	355	43	162	168	88	12	86
c	148	196	223	259	307	358	14	68	40	75
d	262	275	278	182	161	56	2	95	10	82
e	3	78	164	220	277	331	1	16	77	95
f	4	92	219	275	290	265	122	42	55	87
g	178	220	355	29	265	244	127	39	34	180
h	264	274	265	179	53	355	276	42	35	147
i	52	71	96	182	274	295	307	78	62	43
j	288	264	265	260	198	127	108	66	26	97
k	264	274	81	320	129	101	32	65	61	54
l	265	265	274	355	4	4	355	42	37	131
m	67	348	281	71	43	290	295	42	43	112
n	265	267	87	65	3	289	265	30	44	156
o	124	181	223	288	334	42	85	103	79	0
p	55	330	230	140	327	278	281	14	76	98
q	147	244	4	251	265	79	25	39	63	82
r	309	274	244	95	78	43	26	53	49	81
s	136	193	281	327	294	196	123	39	33	173
t	2	64	180	258	267	304	56	38	38	143
u	262	309	25	65	85	258	316	39	34	166
v	305	306	341	57	64	65	71	48	40	106
w	288	1	68	308	295	78	55	38	42	128
x	307	309	124	50	212	230	223	52	46	88
y	327	348	57	230	233	219	223	37	60	88
z	359	245	219	244	11	55	140	138	23	54