International Journal of Multimedia and Ubiquitous Engineering
Vol.8, No.5 (2013), pp.119-130
http://dx.doi.org/10.14257/ijmue.2013.8.5.12

Finger Gesture-based Three-Dimension Mobile User Interaction Using a Rear-facing Camera

Byung-Hun Oh and Kwang-Seok Hong

School of Electronic Electrical Engineering, Sungkyunkwan University, Suwon-Si, 440-746, Korea

sincelife@skku.edu, kshong@skku.ac.kr

Abstract

In this paper, we propose a 3D mobile user interaction for 3D space using a mono camera in mobile devices. The proposed system can support a UI using only the index finger of the hand holding the mobile device. Proposed fingertip detection algorithm was developed to overcome the limitation of the skin color segmentation of previous studies under varying lighting conditions and complex background using MMGC and AdaBoost algorithm. Also in proposed 3D mobile user interface, the estimated coordinates and area of the fingertip can be mapping 3D space of implemented application. The experimental result indicates the feasibility of the proposed algorithm for finger movement-based mobile user interfaces.

Keywords: *Mobile User Interface, 3D Mobile UI, Finger Detection, 3D Gesture Interface*

1. Introduction

The rapid growth of mobile devices including smartphones and tablet PCs has introduced multimedia content and web services into our daily lives. Users want to increase the efficiency of their interaction with mobile devices and applications. To satisfy this demand, user interfaces are increasingly gaining in importance [1]. Mobile user interfaces (UI) allow for interaction between a user and a mobile device. A user's ability to efficiently control the device is strongly related to the UI.

Research has been carried out on new types of input interface systems. Accordingly, the latest mobile devices can support a variety of UIs ranging from keypads to multi-touch screens and sensors. However, as the number of UI sensors increases, the sensors will become difficult to integrate into existing small form-factor mobile devices at hardware level [2]. In addition its function cannot be represented exactly in a simplified two-dimensional form. Therefore, vision-based 3D mobile UIs can serve as an important way to use camera-equipped mobile devices since no new hardware is necessary.

Vision-based human–computer interaction is a popular research topic, and computer vision has been increasingly used to control such interfaces. Interaction with mobile devices such as smartphones and personal digital assistants has also become a potential application area, especially for the wireless industry. Interest in developing new techniques is increasing. For example, mobile gaming is an important business field that could leverage advanced interaction techniques.

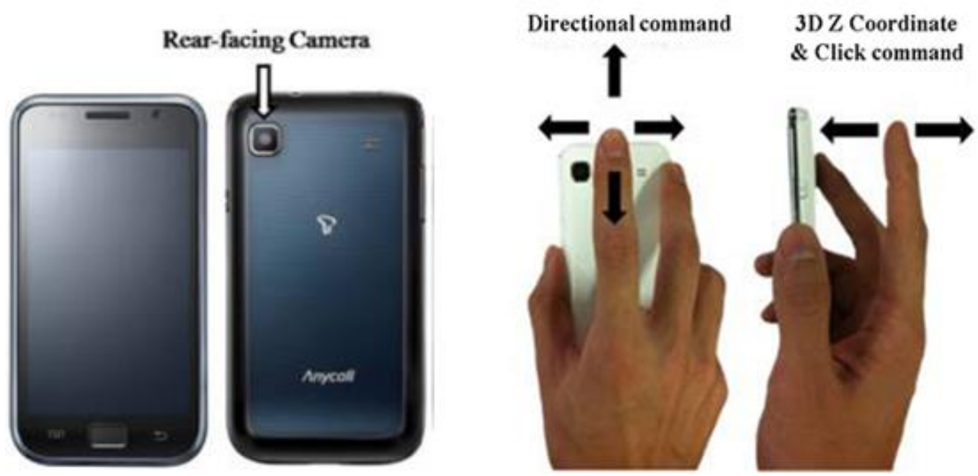
As Equipment for controlling a 3D space, a data glove, a 3D mouse, a 3D input device using an infrared sensor, *etc.*, have been developed. Development of the gaming industry has

ISSN: 1975-0080 IJMUE

required a new type of interface, which became an opportunity to greatly develop a 3D interface field.

To introduce the concept of the three-dimensional, Researchers discuss the interaction techniques for generic 3D tasks on the basis of the use of traditional 2D images. Markers are the key element for deriving three-dimensional structural information from images in a way that recognizes the existing methods [3, 4]. However, this marker-based system has obvious defects since markers always need to be included in the image or additional equipment is required for controlling objects, which results in reduced immersion.

We present a 3D mobile UI based on finger gesture estimation using the Adaboost algorithm [5, 6]. The proposed systems for mobile UIs require two-handed interaction and were developed for a front-facing camera. They also estimate the finger movement and gestures and operate 3D mobile applications using finger gesture images conveyed through the rear-facing camera. The basic idea of the proposed system relies on the finger gestures recognized from captured images of the mobile device's camera as depicted in Figure 1(a) and the basic control principle in illustrated in Figure 1(b).



(a) Mobile device and rear-facing camera (b) Basic control principle

Figure 1. Proposed 3D mobile user interface

2. System Architecture

Figure 2 shows the implementation details of the proposed architecture that can accurately detect a fingertip from a single image captured by a mobile phone camera. The detection module is composed of two pre-processing procedures which input image and fingertip area estimation as sequential steps using the Adaboost algorithm. The two pre-processing procedures used are based on gradient information and color information. An AND image between the morphological gradient combination image and the color-base pre-processed image is finally used to input the Adaboots algorithm in order to detect the fingertip area. The estimated coordinates and the area of the fingertip are then further refined and fed into the 3D gesture interface module, which determines the commands of mobile applications.

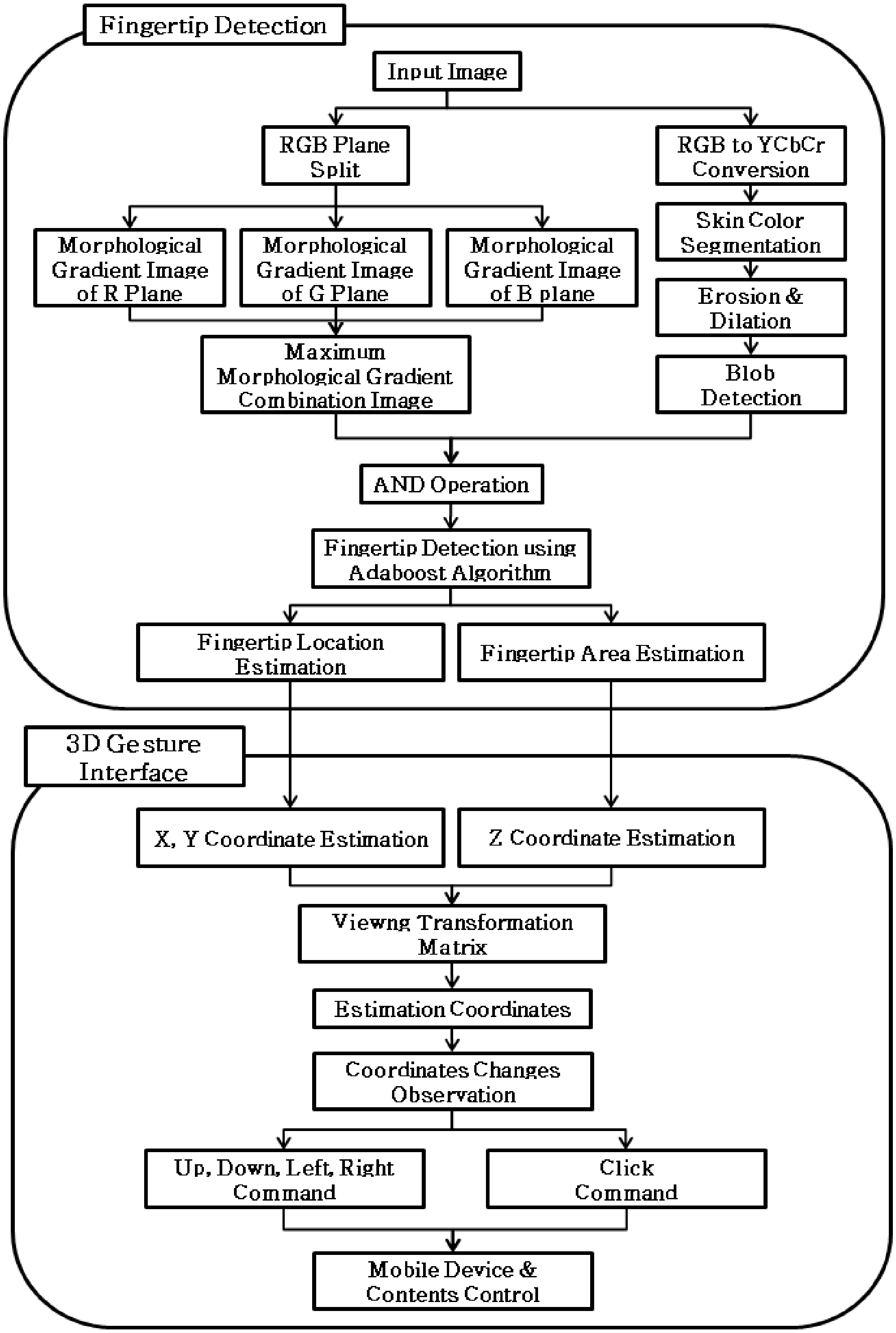


Figure 2. Proposed system architecture

3. Fingertip Detection

3.1. Fingertip Detection Method

Figure 3 shows the results of the pre-processing procedure of the fingertip detection method. In computer vision research, color information is important and widely used in image processing and analysis. There are a variety of color spaces, for example RGB, normalized RGB, LUV, LAB, XYZ, YUV for color coding, YIQ, HSV, HIS, and GLHS from computer graphics [7]. The most common color space is RGB. However, it is not suitable for constructing an accurate skin color model since a high correlation exists between the three components, R, G and B. Thus we use YCbCr space for skin color segmentation [8]. There are non-linear relations between chrominance components (Cb, Cr) and a luminance component (Y). The chrominance components are almost independent of the luminance component in the space. Hence, lots of skin color researches operate only on the Cb and Cr plane. For the skin color segmentation, each pixel is classified as being either skin or non-skin and is converted into a new binary image with the threshold value analysis defined as follows:

$$SkinColor(x,y) = \begin{cases} 1 & if(77 \leq C_b \leq 127) \cap (133 \leq C_r \leq 178) \\ 0 & otherwise \end{cases} \quad (1)$$

To reduce the effects of small background objects in the binary image, two basic morphological operations are performed; non-skin color objects that are larger than the 3 by 3 structuring element still exit. To remove large objects except for the finger region, we labelled each blob. In the blob detection, connected component labeling was used to divide and label the blobs [9].

An obvious gradient value cannot be acquired by using only a gray image because of the equalizing effects of red, green, blue(RGB) to gray conversion. Thus, we devise the maximum morphological gradient values in the split R, G, and B planes and combine them into a single image. This allows for clearer gradient values than those of a gray image. The Maximum Morphological Gradient Combination(MMGC) [10] image is defined in the following equation:

$$MMGC = \sum_{j}^{height} \sum_{i}^{width} \max(MG_R(i,j),\ MG_G(i,j),\ MG_B(i,j)) \quad (2)$$

Finally, we obtained an AND image between MMGC and the resulting image of the skin color segmentation and blob detection. The fingertip detection from the AND image, which includes both the clear gradient information and the non-skin color subtraction, has a higher performance than that derived from the original image.

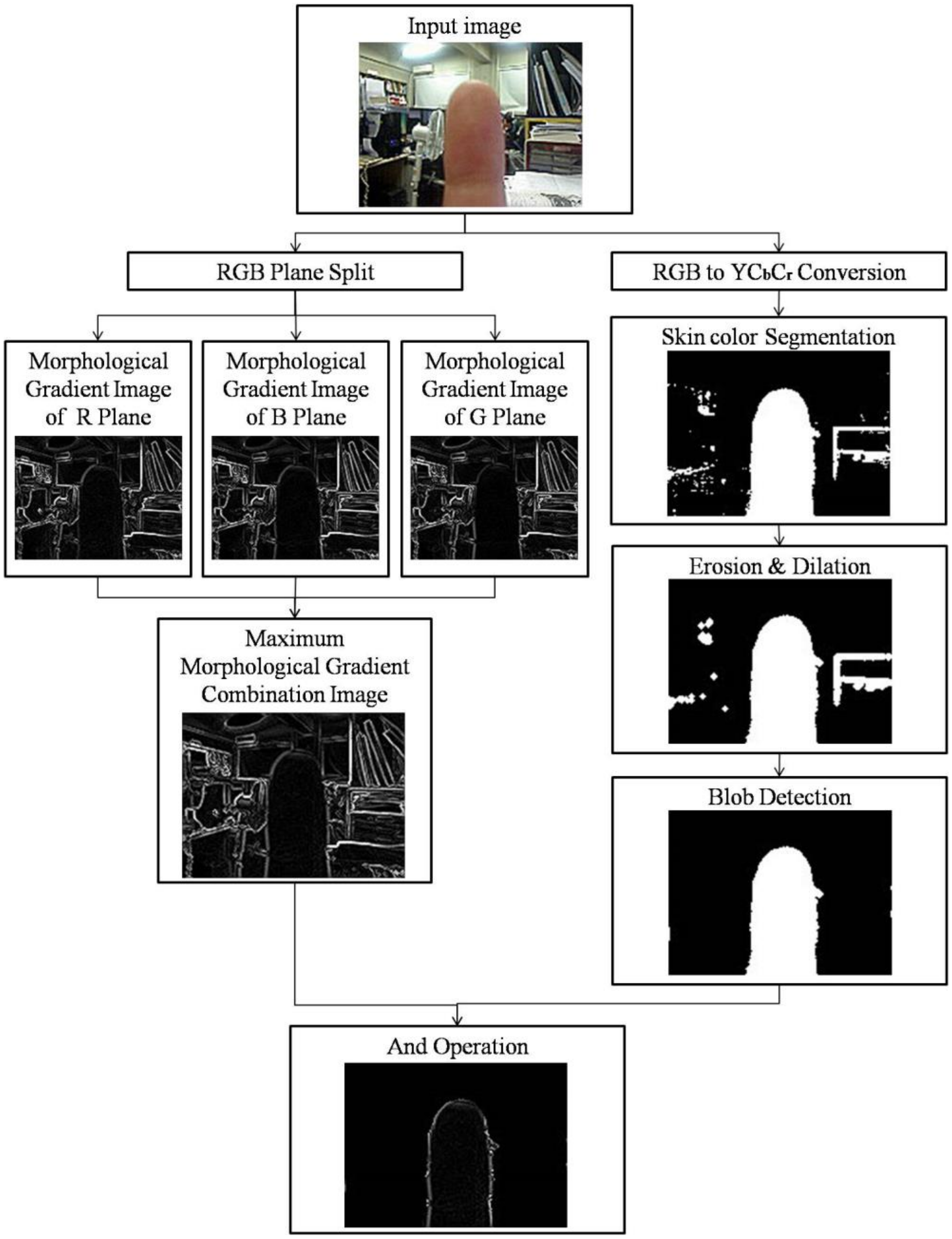
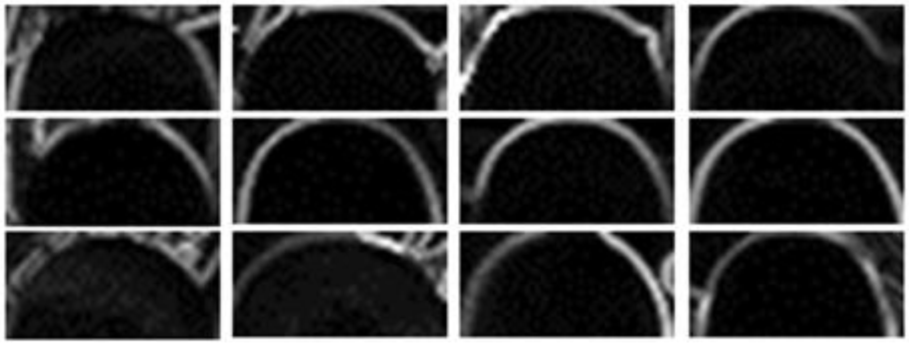


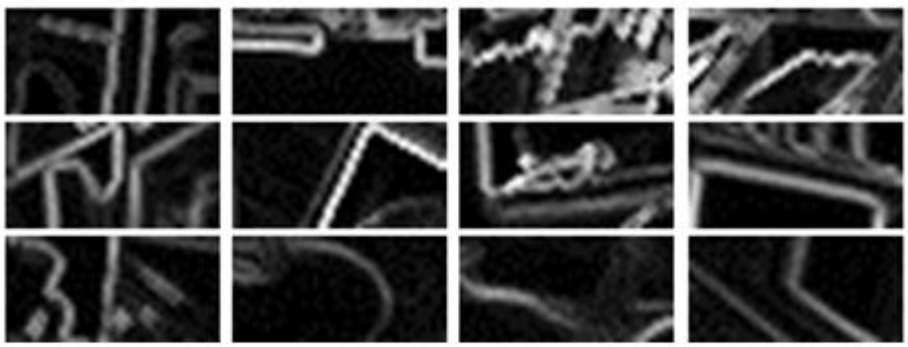
Figure 3. Result of pre-processing procedure of the finger detection

3.2. Fingertip Detection Based on Adaboost Algorithm

For our fingertip detection system, we collected the half-circle area images of the fingertip from the results of pre-processing as positive samples. The samples were collected under various illumination conditions. The number of the collected positive samples is 2240, and 4500 image from the results of pre-processing are collected as the negative samples for the training process. The fingertip cascade classifier is 13-stage cascade that is 20 × 10 in size. Figure 4 shows some positive and negative samples.



(a) Positive samples



(b) Negative Sample

Figure 4. An example of some positive and negative samples

4. Three-Dimension Mobile User Interface

4.1. Three-Dimension Coordinate Estimation

The X and Y coordinate are generated by changing of the point value of the fingertip. In the proposed system, the center pixels of the detected fingertip can be substituted for the X and Y coordinate. The Z coordinate is generated by changing of the area of the fingertip. The changing of the area of the fingertip can be mapping 3D space of implemented application.

4.2. Direction Command

In the proposed system, finger gestures can be performed for the operations of click, up, down, left and right. They can play the same role as a directional keypad and mouse. All of the directional commands except click are defined by chessboard distance. The moved distance, direction and the instant speed of the finger point between two frames determines the directional commands.

Assume that we have an N x M digital image I with $\mathrm{i}[\mathrm{i},\mathrm{j}] \in 0 \leq \mathrm{i} \ \leq \mathrm{N}-1, 0 \leq \mathrm{j} = \mathrm{j} \leq \mathrm{M}-1$ the current pixel position of the finger point is at $(\mathrm{i}_2, \mathrm{j}_2)$, while the previous position was at. $(\mathrm{i}_1, \mathrm{j}_1)$T he chessboard distance is defined as:

$$d_{chess} = \max(|i_2 - i_1|, |j_2 - j_1|) \tag{3}$$

Table 1 shows each of the directional commands. In the table, the variable σ can be change according to the frame rate of the camera properly. In our system σ=4 is used with the frame rate of 10fps, 24bit color , and a resolution of 320 x 240 pixels.

Table 1. Directional commands by finger gesture recognition

Directional Command	Table Column Head
UP	$\lvert j_2 - j_1\rvert > \lvert i_2 - i_1\rvert, j_2 > j_1, d_{chess} > M/\sigma$
Down	$\lvert j_2 - j_1\rvert > \lvert i_2 - i_1\rvert, j_2 < j_1, d_{chess} > M/\sigma$
Left	$\lvert i_2 - i_1\rvert > \lvert j_2 - j_1\rvert, i_2 < i_1, d_{chess} > N/\sigma$
Right	$\lvert i_2 - i_1\rvert > \lvert j_2 - j_1\rvert, i_2 > i_1, d_{chess} > N/\sigma$

4.3. Click Command

The instantaneous rate of change of the fingertip area is applied to the click command by finger gestures, and it is defined as the fingertip area of the current frame over the fingertip area of the previous frame. When a user moves his finger back and forth towards the camera, similar to the gesture of clicking a mouse, a click command is triggered. The threshold value is determined experimentally. A double-click command is generated when a click command occurs twice continuously over a defined frame duration. The length of the frame for a double-click command can be properly selected by the system frame rate.

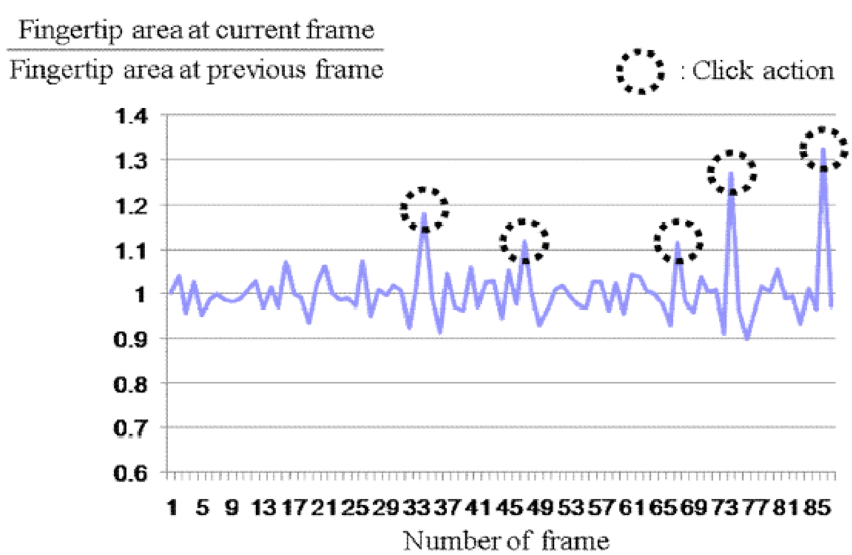


Figure 5. Example of click action

5. Experiment Result and Application

5.1 Fingertip Extraction Evaluation

We used a light meter to evaluate the performance of fingertip tracking under varying lighting conditions. Five lighting conditions were set as shown in Table 2 including average lux values.

Table 2. Average lux values in the five different lighting conditions

	Places	Intensity of Illumination (lux)
Condition 1	Public areas with dark surroundings	20
Condition 2	Office	400
Condition 3	Corridor	1,000
Condition 4	Outdoor in shade	2,000
Condition 5	Outdoor in sunny side	60,000

To evaluate the performance of the proposed fingertip detection, approximately 1,000 images including 100 images of each condition (five lighting conditions with simple backgrounds and five lighting conditions with complex backgrounds) were used in the real-time fingertip detection program. The trained cascade classifier for fingertip detection is a 13-stage cascade classifier with the required false alarm rate set at 1×10-6 and its size is 20×10 pixels. For the detection results, we measured the number of hits, misses, false pictures, and the detection time and detection rate (Table 3).

Table 3. The performance of fingertip detection classifier

		Hit	Missed	False	Detection Rate	Detection Time
Simple background	Cond.1	98	2	1	97%	97 ms
	Cond.2	100	0	1	99%	112 ms
	Cond.3	99	1	0	99%	104 ms
	Cond.4	98	2	1	97%	99 ms
	Cond.5	100	0	2	98%	101 ms
Complex background	Cond.1	96	4	1	95%	109 ms
	Cond.2	98	2	0	98%	102 ms
	Cond.3	97	3	2	95%	99 ms
	Cond.4	99	1	2	97%	109 ms
	Cond.5	95	5	2	93%	104 ms
Average		98	2	1.2	97%	103 ms

By analyzing the detection results, we found that the detection rate is 97% and the detection time is 0.103 seconds. The proposed fingertip detection algorithm showed good robustness against lighting variance and complex background including skin-like color. As a result, this indicates the feasibility of the proposed algorithm for finger movement-based 3D mobile user interface.

5.2 Performance Evaluation on Gesture Commands Recognition

To evaluate the finger interface recognition performance, five participants used a directional command test program. Each finger interface input command was tested 100 times. The overall recognition rate was 92.6%.

Table 4. Finger interface command recognition

Input commands	The number of finger interface recognition	Recognition rate
Click	81/100	81%
Up	87/100	87%
Down	99/100	99%
Left	98/100	98%
Right	97/100	97%
Total	463/500	92.6%

5.3 System Configuration

The proposed system is implemented on the Samsung Galaxy S2 smart phone, which is provided with a 1.2GHz dual core processor, 1GB MB of memory, and a 8MP rear-facing camera.

The operating system is Android with Java support. Since Android applications based on Java are slower than applications written in native C/C++ languages, we use the Android NDK and the JNI interface [11, 12]. The Android NDK is a companion tool to the Android SDK that allows developers to build performance-critical portions of applications in native code. It also provides headers and libraries that allow developers to build activities, handle user input, use hardware sensors, and access application resources when programming in C or C++. If Android developers write native code, applications are still packaged into an .apk file and still run inside a virtual machine on the device; the fundamental Android application model does not change

5.4 Application

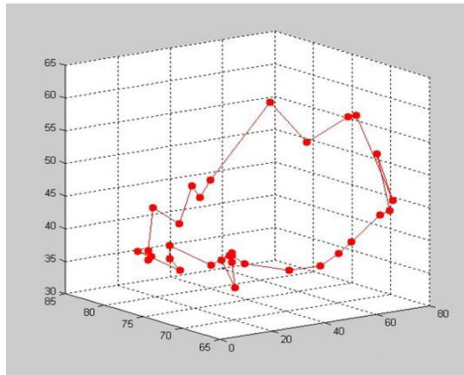
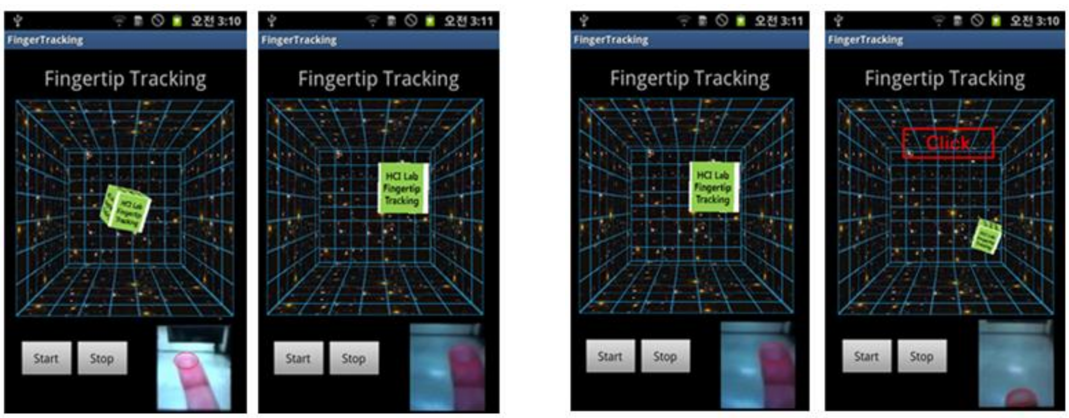


Figure 6. 3D trajectory for the fingertip tracking test

Proposed system can estimated the 3D position accurately. As shown in Figure 6, the line is drawn according to the fingertip 3D positions.

We implemented a few test applications to demonstrate the proposed system's strengths and limitations. The proposed system can accurately estimate the 3D position and it is applicable to 3D user interface as shown in Figure 7(a). Also, Figure 7(b) shows the click command examples by finger movement.



(a) 3D user interface (b) Click command examples

Figure 7. 3D application

6. Conclusion

We propose a 3D mobile user interface for 3D space using a mono camera in mobile devices. The proposed system for mobile UIs requires two-handed interaction and was developed for a front-facing camera. The proposed system further estimates the finger movement and gestures and operates 3D mobile applications using the finger gesture images through the rear-facing camera. As a result, this indicates the feasibility of the proposed algorithm for finger movement-based mobile user interfaces. And the proposed finger detection and 3D mobile interface system using a rear-facing camera was developed on an Android platform for directional commands and 3D applications. Furthermore the system is implemented on Android platform and Android NDK, Native C and Java Native Interface(JNI) to reduce calculation time for a real time finger gesture recognition.

Acknowledgements

This work was supported by Priority Research Centers Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012-0005861) and by MKE, Korea under ITRC NIPA-2012-(H0301-12-3001).

References

[1] A. Mulder, “Hand gestures for HCI”, Technical Report 96-1, Simon Fraster University, **(1996)**.
[2] E. Koh, J. Won and C. Bae, “On-premesie skin color modeling method for vision-base hand tracking”, Consumer Electronics, ISCE IEEE 13 International Symposium, **(2009)**.
[3] F. Zhou, H. B. Duh and M. Billinghurst, “Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR”, IEEE/ACM International Symposium on Mixed and Augmented Reality, **(2008)**, pp. 193-202.
[4] M. Fiala, “ARTag, a fiducial marker system using digital techniques”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, **(2005)**, pp. 590-596.
[5] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, **(2001)**, pp. 511-518.
[6] Y. Freund and E. Robert, “A decision theoretic generalization of on-line learning and an application to boosting”, Computational Learning Theory: Eurocolt ., vol. 95, **(1997)**, pp. 23-37.
[7] Y. -W. Wu and X. -Y. Ai, “Face Detection in Color Images Using AdaBoost Algorithm Based on Skin color Information”, Knowledge Discovery and Data Mining, **(2008)**, pp. 339–342.
[8] V. Vezhnevets, V. Sazonov and A. Andreeva, “A survey on pixel based skin color detection techniques”, GraphiCon, Moscow, Russia, **(2003)**.
[9] M. B. Dillecourt, H. Samet and M. Tamminen, “A general approach to connected component labeling for arbitrary image representations”, Journal of the ACM, vol. 39, no. 2, **(1992)**, pp. 253-280.
[10] J. -H. An, B. -H. Oh and K. -S. Hong, “A Robust Fingertip Detection Algorithm for Complex Backgrounds Using AdaBoost Algorithm”, ICCCIT 2011, **(2011)** December.
[11] Android NDK | Android Developers, http://developer.android.com /sdk/ndk/index.html, Java Native Interface, Wikipedia,http://en.wikipedia.org.
[12] M. B. Dillecourt and H. Samet, “Tamminen Java Native Interface”, Wikipedia,http://en.wikipedia.org.

Authors

Byung-Hun Oh received the B. S. degrees in electronic engineering from the Eulji University, in 2011, respectively. He is presently a M.S candidate at the department of Information and Communication Engineering Sungkyunkwan University. His current research focuses on digital image processing and pattern recognition.

Kwang-Seok Hong received his B.S., M.S., and Ph.D. in Electronic Engineering from Sungkyunkwan University, Seoul, Korea in 1985, 1988, and 1992, respectively. Since March 1995, he has been a professor at Sungkyunkwan University, Suwon, Korea. His current research focuses on human-computer interaction, five-sense recognition, interaction, and representation.