Gesture Recognition Based Interactive Boxing Game

Ji-Young Park, and June-Ho Yi

School of Information & Communication Engineering
Sungkyunkwan University
300, Chunchun-dong, Jangan-gu Suwon 440-746, Korea

jiyp@ece.skku.ac.kr, jhyi@ece.skku.ac.kr

Abstract

This paper features a gesture recognition based interface for a PC action game called “O.J. Boxing”. We have developed an interactive version of the game where gesture recognition based input technologies are successfully applied. We have implemented the game as playable and enjoyable using simple vision algorithms. The recognition system is composed of server and client computers: the server recognizes the player’s movements, and the client the punch. Punch gestures detected by the client are fed into the server to control the game. We have confirmed that the interactive version of the boxing game using a gesture recognition based interface is more real and exciting to play than the keyboard based game.
Keyword: Gesture recognition, Interactive game, Human computer interface.

I. Introduction

Gesture recognition has a significant role to play in human-computer interaction of the future. The user will be no longer confined to the desktop for the access of computational resources using gesture recognition based input technologies. Significant research efforts have been made to replace conventional machine-centered interfaces such as keyboard, mouse and joystick by vision based interfaces [1-6]. This research features a gesture recognition based interface to a PC-based action game. We have used a flash-animated game called “O.J. Boxing” that is easily downloadable at several flash game sites. The purpose of this research is to show that the vision based interactions could make game interaction more enjoyable. We have developed interfaces for the “O.J. Boxing” game as illustrated in Figure 1 where the user is supposed to mimic the movements of a boxer by moving or punching. There are four actions to play: move left, move right, left punch and right punch. Please refer to the site http://vision.skku.ac.kr/ boxingdemo.zip for video clips of the game.
For interactive games, the response time should be very fast. The user should sense no appreciable delay of the computer’s response. On the other hand, expensive computation for accuracy is many times not necessary for interactive games because human’s perception is not precise. Thus, some fast

and simple vision algorithms can fit well with interactive games. Our system is initially designed for a network game. The first round of the implementation is finished for stand-alone games, and the extension of the system to a network game is under progress. The game is readily playable and enjoyable. We have confirmed that the game using a gesture recognition based interface based on simple vision algorithms is more real and exciting to play than using the keyboard interface. The performance of the system shows great promise that we will soon be able to substitute the conventional game interface by a gesture-recognition based interface.

This paper is organized as follows. The following section below briefly reviews an overview of our system. Section 3 and 4 present the proposed methods for movement and punch recognition of a player, respectively. Finally, experimental results are reported in section 5.

Figure 1. Gesture recognition based interactive game: “O.J. Boxing”

II. Overview of Our Approach

Let us briefly overview the entire system. As can be seen in Figure 2 it is composed of a large display device for projecting the game scene, two cameras for grabbing input images and two PCs. Camera 1 captures the player’s movements, and the other camera the punches. Two computers are connected to the internet and player’s action data are transferred through the internet. Two standard 2GHz Pentium 4 PCs are used with CCD cameras. We have employed the internet connection for the upgrade of the system to a networked game.

Figure 3 shows a block diagram of the entire operations. The system is divided into three parts. The server computer is responsible for the first part. It recognizes the player’s movement using camera 1. The second part is for the client computer that identifies punches using camera 2. The third part is for networking. Punches detected by the client computer are transferred to the server for control of the game.

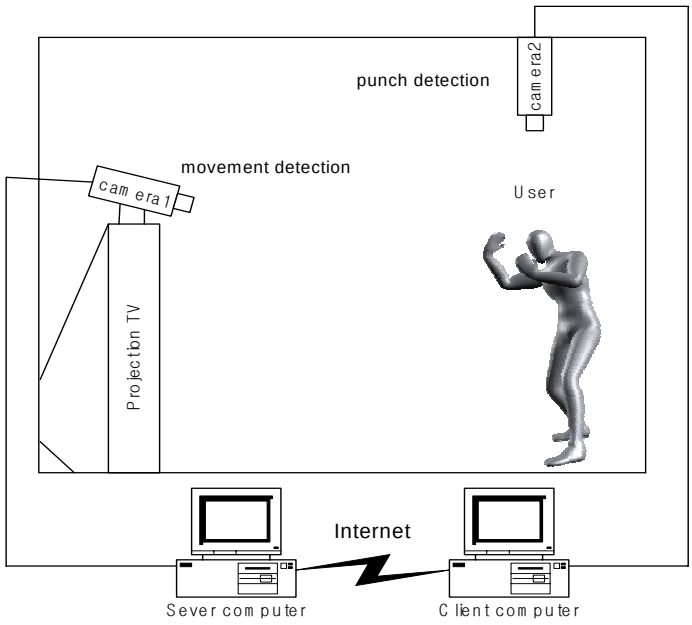


Figure 2. System framework

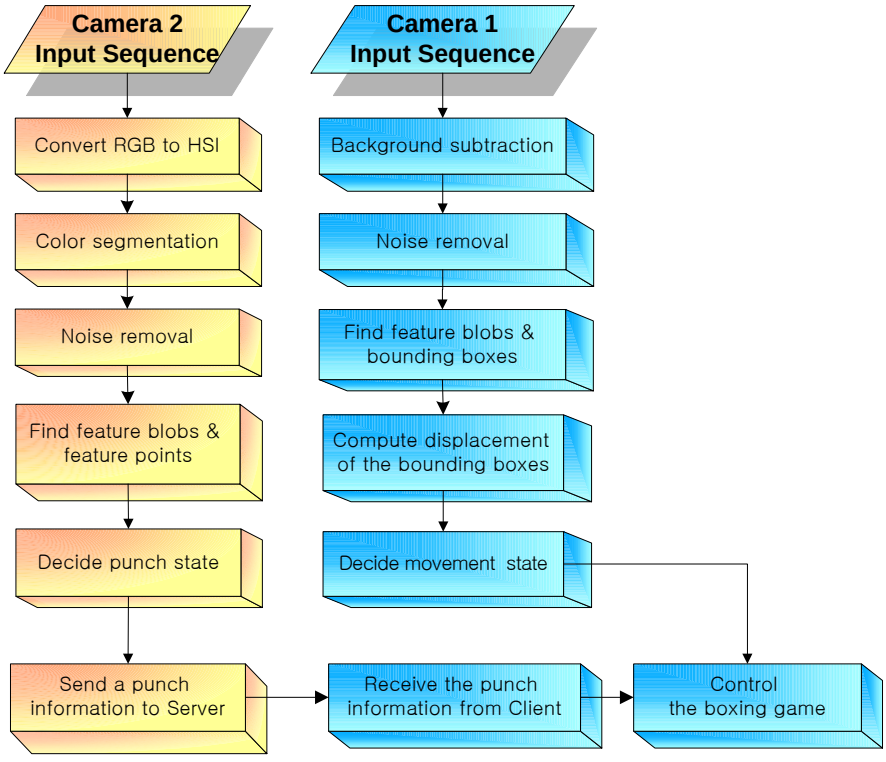


Figure 3. Recognition process of player's gesture

III. Movement recognition

We recognize player's movements that avoid the opponent's punches or approach the opponent for punching. We assume that these movements are gestures swinging the upper part of the body to the

left or to the right. The player's gesture is recognized by the system and its corresponding key event is invoked.

*A. **Noise removal***

In order to achieve reliable as well as fast recognition, we use stationary background. A simple background subtraction process gives several blobs in the input image. We use morphological operations and connected component labeling algorithm for noise removal and blob detection, respectively. Opening and closing with an isotropic structuring element are used to eliminate specific image details smaller than the structuring element, yet the global shape of the object is not distorted. Closing connects objects, which are close to each other, fill up small holes, and smooth the object outline by filling up the narrow gulf. Meanings of 'near', 'small', and 'narrow' are related to the size and shape of the structuring element. Opening is the result of eroding and dilating an image: Closing is vice versa. The erosion of binary image B by structuring element S is defined by

$$B \ominus S = \left\{ b \middle| b + s \in B \ \forall s \in S \right\} \tag{1}$$

The dilation of binary image B by structuring element S is defined as follows.

$$B \oplus S = \bigcup_{b \in B} S_b \tag{2}$$

A connected component labeling finds all connected components in an image, and assigns a unique label to all points in the same component [7]. It is used to compute the size and position of a blob having the same label. To reduce time-consuming computation, such as searching feature blobs, we delete the blob of which the size is smaller than a predetermined threshold.

B. Determining the feature blobs

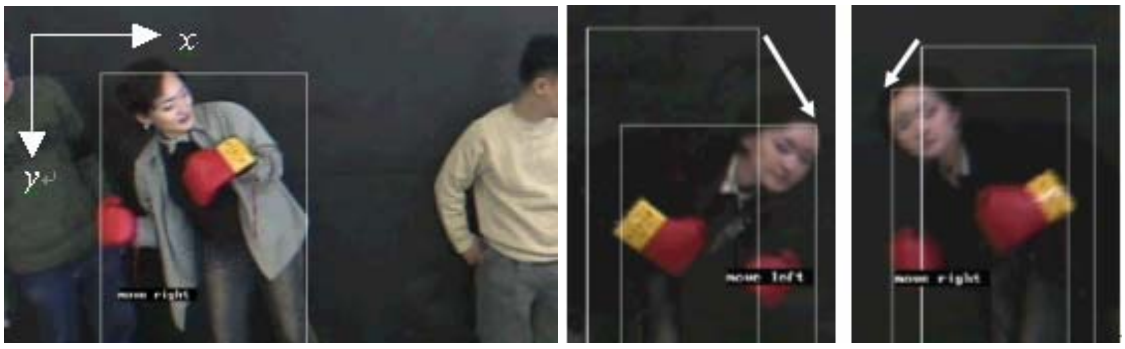
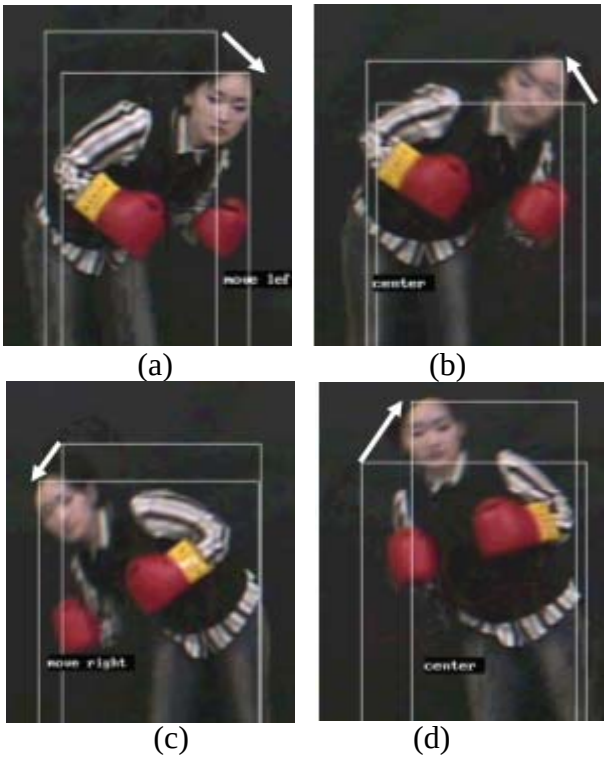


Figure 4. Examples of detection results: (a) Other people as well as the player are in the input image. (b) The player is dressed in colors similar to the background color.

To reduce time-consuming computation, such as searching feature blobs, we delete the blob of which the size is smaller than a predetermined threshold. When playing the game, a player cannot see the input image from the camera. There is possibility of existence of another person in the input image (illustrated in Figure 4 (a)). Moreover, in the case that the color of the player's clothes is similar to the background, the feature blob representing the player may be split as illustrated in

Figure 4 (b). The feature blobs are determined by their center coordinates. Their coordinates should be closer to the center of the image. The bounding box encompasses all feature blobs. It is reliably computed despite the fact that the player is dressed similar to the background color. Figure 4 (a) shows examples of detection results when another person as well as the player appears in the input image when the player is dressed in colors similar to the background color. Two rectangles in Figure 4 (b) represent bounding boxes of the previous and current frames.

The movement direction of the player is detected based on the information of the change of bounding boxes that encompass feature blobs. The bounding box is updated according to the movement of the player. Figure 5 shows two bounding boxes of the previous and current frames. The movement direction is computed using accumulated displacement of the bounding box.



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

Figure 5. Update of a bounding box for various swinging gestures. (a) center → left, (b) left → center, (c) center → right, (d) right → center

C. Decision of movement directions

A swinging gesture continuously invokes a ‘move’ event during the gesture and a standup gesture stops invoking the event without delay. Using these methods, a small movement of the player in consecutive frames can stop invoking the ‘move’ event. However, jitter can occur even though the player wants to remains still. To circumvent this problem, displacement for each direction is continuously accumulated. For example, when the player tilts to the left as shown in Figure 4 (a), the displacement of the top-left corner of the bounding box is accumulated. If it is larger than a predetermined threshold, the motion will create an event for moving to the left side. When the direction is changed to the right side, accumulated amount of displacement of the top-right rectangle corner is thresholded to detect a “move right” action. In this case, the amount of accumulated displacement is set to zero on the change of the movement direction.

IV. Recognition of Punch Gestures

Punch gestures are very fast. In order to guarantee realtime detection of punches, we have employed color segmentation. The player puts on red gloves. Wearing colored gloves not only help color segmentation but also greatly increase the reality of the boxing game. We convert the RGB input image into the hue image to detect the two glove regions efficiently. We obtain two feature blobs that correspond to the two glove regions. To clean up the noise, we have used a Gaussian filter. Then, using morphology operations, we merged small blobs in the glove regions and removed noise blobs in other regions.

We define a punch as a stretching gesture of the player's arm toward the front. We classify the state of a punch into "left" or "right". We decide a punch state using two feature points extracted from two feature blobs. In Figure 6, the black cross marks the feature points in each blob region located at the bottom pixel. The two feature points are used to compare the relative distance of two gloves.

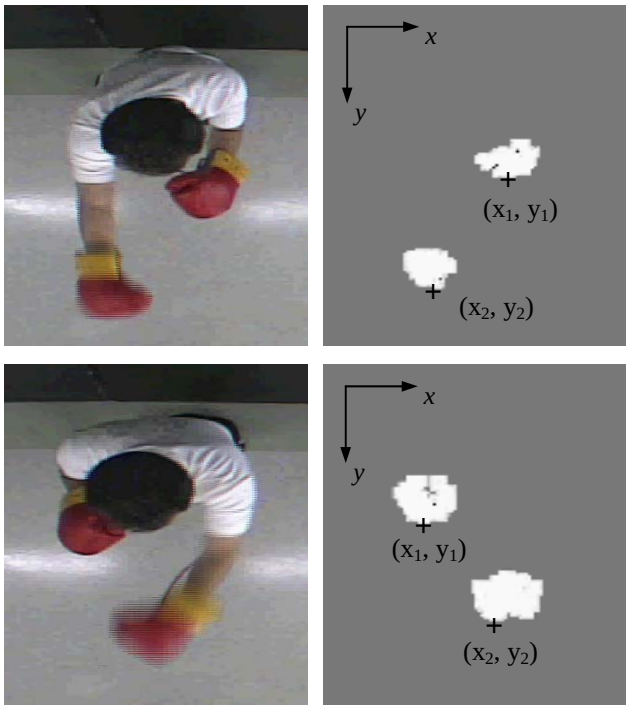


Figure 6. Segmentation of glove regions

When the distance between y coordinates of two feature points is above a predetermined threshold, we decide that there is a punch. y1 and y2 are the y coordinates of the two feature points. P denotes a Boolean value representing whether there is a punch or not.

$$\begin{aligned} &If \ \ |y_1 - y_2| > T, \ then \ P = True \\ &else \ P = False \end{aligned} \tag{3}$$

If P is true, there is a punch. We compare x coordinates of two feature points to classify the punch into the left or the right punch. x1 and x2 are the x coordinates of the feature points. S is a binary variable representing left or right punch.

$$If\ x_1 - x_2 > 0, S = 0 \text{ (right punch)}$$
$$If\ x_1 - x_2 < 0, S = 1 \text{ (left punch)} \quad (4)$$

Left or right punch events of the player take place as if its corresponding keyboard key is pressed. Unlike swinging gestures, a punch gesture invokes the punch event once during the punch. We compare the punch state in the current frame with that in the previous frame to check the change of the punch state. If the same state continues, no more punch event is invoked.

V. Experimental Results

We applied our methods to analyze a player's gesture for controlling the game 'O. J. Boxing'. There are a few operations for controlling the game; moving right or left and punching O. J. We confirmed that the game in the computer vision interface became more real and exciting than that in keyboard interface. Table 1 summarizes the experimental results. The accuracy measures the frequency of correct recognition of the player's motion. Detection of some actions were missed, however, the player did not perceive the misses. We have observed that most players controlled the game well on their first try. Moreover, they seemed to become naturally engaged in the game itself.

Table 1. Correct detection accuracies of a player's motion

	# of total actions	# of recognized actions	# of missed actions	Accuracy (%)
Move right	300	289	11	96.3
Move left	300	281	14	95.3
Punch right	300	280	20	93.3
Punch left	300	276	24	92
Total	1200	1,131	69	94.3

The processing rate of the algorithm itself is 29.5 frames per second. However, the rate of the total system slowed 16.4 frames per second due to the performance of the frame grabber used in the system. With this speed, the player could enjoy the game without any noticeable delays.

VI. Conclusions and Future Work

We have developed an interactive game based on a PC based boxing game called "O.J. Boxing" using a gesture recognition based interface. Further extension to the current system is under progress to be capable of the network game using avatars. We will make some modifications to the current display of O.J. Simpson. The O.J's face will be replaced by opponent's face and the face will be displayed distorted when hit by the opponent. Figure 7 illustrates the network version of the game.
The performance of our system shows great promise that the methodology will replace the conventional machine-centered interfaces by natural human-centered interfaces in near future. In addition, more efforts should be made to develop various attractive game contents that suit vision technologies well.

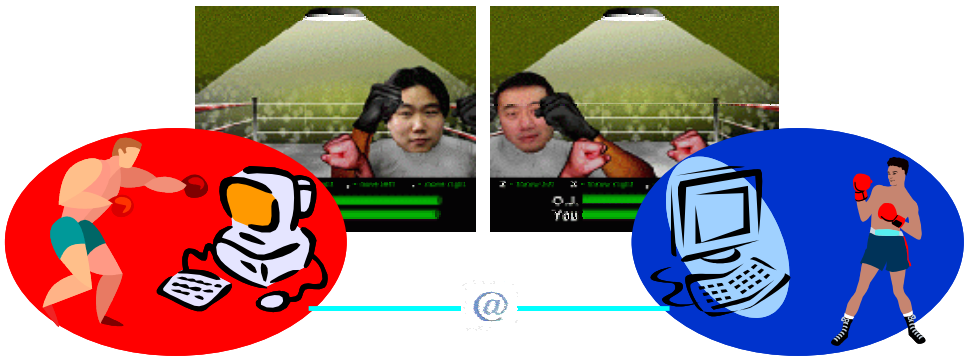


Figure 7. Network based boxing game

Acknowledgment

This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) through the Biometrics Engineering Research Center (BERC) at Yonsei University and the intelligent robotics frontier R&D program funded by the Ministry of commerce, industry and energy of Korea.

References

[1] A W.T. Freeman, D. Anderson, P. Beardsley, C. Dodge, H. Kage, K. Kyuma, Y. Miyake, M. Roth, K. Tanaka, C. Weissman and W. Yerazunis, “Computer Vision for Interactive Computer Graphics,” *IEEE Computer Graphics and Applications*, Vol. 18, No. 3, pp. 42-53, 1998.

[2] C. Wren, F. Sparacino, A. Azarbayejani, T. Darrell, James W. Davis, T. Starner, Kotani A, C. Chao, M. Hlavac, K. Russell, Aaron Bobick, and Pentland A. “Perceptive spaces for performance and entertainment,” *Proceedings of ATR Workshop on Virtual Communication Environments*, 1998.

[3] S.S. Intille, J.W. Davis, A.F. Bobick, “Real-Time Closed World Tracking,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 697-703, 1997.

[4] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, “Pfinder: Real-Time Tracking of the Hman Body,” *IEEE Transaction on Pattern Analysis Machine Intelligence*, Vol. 19, pp. 780-785, 1997.

[5] Andrew Wu, Mubarak Shah and N. da Vitoria Lobo, "A Virtual 3D Blackboard: 3D Finger Tracking using a Single Camera" *Proceedings of the Fourth International Conference on Automatic Face and Gesture Recognition*, pp. 536-543, 2000.

[6] Gary R. Bradski, "Computer Vision Face Tracking for Use in a Perceptual User Interface,” *Intel Technology Journal* Q2, 1998.

[7] Milan Sonka, Vaclav Hlavac, and Roger Boyle, "Image Processing, Analysis, and Machine Vision" Second Edition, PWS Publishing

Ji-Young Park received the B.S and M.S. degrees in Electrical and Computer Engineering from Sungkyunkwan University, Korea, in 2000 and 2002, respectively.

She is currently a Ph. D candidate at Sungkyunkwan University. Her primary research interests include gesture recognition for vision-based computer interaction and object shape recovery using structured light.

June-Ho Yi received the B.S. degree from the Seoul National University, Seoul, Korea, in 1985, the M. S. degree from the Pennsylvania State University, University Park, PA, in 1987, and the Ph.D. degree from Purdue University in 1994, all in Electrical and Computer Engineering.

In 1989, he was a research scientist at the Samsung Advanced Institute of Technology. From 1994 to 1995, he was a research scientist at the University of California, Riverside. From 1995 to 1996, he was a senior research scientist at the Korea Institute of Science and Technology, Seoul, Korea. In 1997, he joined Sungkyunkwan University, Korea, and he is currently an associate professor of School of Information and Communication Engineering. His research interests are broadly in the areas of computer vision and vision-based computer interaction.