Vision-Based Bare-Hand Gesture Interface for Interactive Augmented Reality Applications

Jong-Hyun Yoon, Jong-Seung Park, and Mee Young Sung

Department of Computer Science & Engineering, University of Incheon,
177 Dohwa-dong, Nam-gu, Incheon, 402-749, Republic of Korea
{jhyoon, jong, mysung}@incheon.ac.kr

Abstract. This paper presents a barehanded interaction method for augmented reality games based on human hand gestures. Point features are tracked from input video frames and the motion of moving objects is computed. The moving patterns of the motion trajectories are used to determine whether the motion is an intended gesture. A smooth trajectory toward one of virtual objects or menus is classified as an intended gesture and the corresponding action is invoked. To prove the validity of the proposed method, we implemented two simple augmented reality applications: a gesture-based music player and a virtual basketball game. The experiments for three untrained users indicate that the accuracy of menu activation according to the intended gestures is 94% for normal speed gestures and 84% for fast and abrupt gestures.

1 Introduction

For the interactions at immersive 3D gaming environments, it is natural to use means of human-to-human interaction to the human-computer interaction (HCI). The hand gestures are a means of non-verbal interaction. In recent years, many vision-based gestural interfaces have been proposed and most of the previous work has been focused on the recognition of static hand postures. The fusion of the dynamic characteristics of gestures has only recently been taken much interest[1]. *FingerMouse*[2] allows users to perform pointing gestures to control the cursor. *Finger Track*[3] is a vision-based finger tracking system on top of the workspace. Hardenberg and Brard[4] also have developed a finger-finding and hand-posture recognition system. In the system, the finger, moving over a virtual touch screen, is used as a mouse. In a vision-based gesture interface, it is required to discriminate gestures given purposefully as instructions from unintended gestures. Lenman et al.[5] developed a gesture interface using marking menus. A kind of marking menu which is suitable for a pen device, called *FlowMenu*, was also proposed for use with a pen device on wall-mounted large displays[6].

In the gestural interface, hand poses and specific gestures are used as commands. Naturalness of the interface requires that any gesture should be interpretable. But the current vision-based gesture interfaces do not provide a satisfactory solution due to the complexity of the gesture analysis. It is hard to deal with uncontrolled light conditions, complex background of users, and movements of human hands based on articulation and deformation. These reasons increase difficulty both in the hand

R. Harper, M. Rauterberg, M. Combetto (Eds.): ICEC 2006, LNCS 4161, pp. 386 – 389, 2006.

tracking step and in the gesture recognition step. A compromise would be possible using some sensory devices, using simple shaped markers, using marked gloves, using simple backgrounds, or restricting the poses of hands to frontal directions. Those approaches might not be appropriate for gesture interface since they restrict natural human gestures or environments. Our research is concentrated on the naturalness of gesture interface not imposing strict restrictions on gestures.

Computer vision algorithms for interactive graphics applications need to be robust and fast. They should be reliable, work for different people, and work against unpredictable backgrounds. Fortunately, the problem becomes much easier due to the facts that the application context restricts the possible visual interpretations and the user exploits the immediate visual feedback to adjust their gestures. This paper describes a gesture-based natural interface which is fast and stable without any strict restrictions on human motion or environments.

2 Human-Computer Interactions Using Hand Gestures

Our idea of the natural user interface is from the way of a general touch screen system where the system utilizes information both from the user contact point and a touch to the touch sensor to determine the location of a touch to the touch sensor. Touch screens are capable of measuring the touch position for a single touched point. The location of a touch applied by a user is generally determined by measuring signals generated by a touch input and comparing the signals. A touch to the contact point may be an actual physical touch or a proximity touch when a finger is positioned sufficiently close to generate a signal.

For real-time barehanded human computer interaction, we developed a fast hand gesture classification method. We first filter skin color regions and determine the candidate image locations of hands. Then, we detect and track feature points in the skin color regions. The user intension of menu selection is inferred based on the trajectories of feature points near the menu items in a short time interval.

For the proximity touch to a menu item, we locate skin color regions and analyze gestures occurred in the regions. The HSV(hue-saturation-value) model is used to specify skin color properties intuitively. The important property of hue is the invariancy to various lighting environment. For the values r, g, and b in the RGB color space, each range from 0 to 1, hue h ranges from 0 to 360° and saturation s ranges from 0 to 1. For general digital color images, r, g, and b have discrete values range from 0 to 255 and the HSV color space also should be quantized to fit into the byte range. The value component v is given by $v=\max(r,g,b)$ and the saturation s is by $s=(v-\min(r,g,b))*255/v$ if $v \neq 0$. If $v=0$, then s is also zero. A fast computation of h is possible by the following rules:

- If $v=r$ then $h=(g-b)*30/s$.
- If $v=g$ then $h=90+(b-r)*30/s$.
- If $v=b$ then $h=120+(r-g)*30/s$.

If the value of h is negative, it is converted to positive by adding 180 to h. Since $0 \leq h \leq 255$, $0 \leq s \leq 255$, and $0 \leq v \leq 180$, they fit into a byte array.

Based on the statistics, we filter skin color regions. We only use h and s to eliminate the lighting effects. However, under the very dark illumination, the skin

color filtering is unstable. Thus, we exclude the case when intensity is too small. The skin color subspace is bounded by the constraint: ($0 \le h \le 22$ or $175 \le h \le 180$) and $58 \le s \le 173$ and $v \ge 76$. This skin color boundary also works for different human races and different lighting conditions.

Finding fingertip shape requires a fairly clean segmented region of interest. The requirement is hard to be satisfied in a general environment. Instead of finding fingertips, we determine the proximity of hand parts to menu positions and infer the intended user click action using the moving direction of hand parts. We detect enough number of feature points and track them. When a feature point is failed in matching, a new one is detected and added to the feature point set.

The trajectories of all feature points are kept during a fixed interval of time. When one of the skin color regions overlaps one of menu item regions, the trajectories of feature points inside the skin color regions are inspected. If the trajectory shape agrees the user intended menu selection, the corresponding menu item is activated. Unintended gestures include inconsistently moving trajectories containing abrupt change of moving direction, small motion trajectories where the starting positions are near the menu item, and confusing trajectories crossing multiple menu items.

3 Experimental Results

We implemented two practical augmented reality applications: a music player and a virtual basketball game. In the music player, several menu icons are displayed on the top of the screen and a user can activate a menu by hand gestures. The music player has four menu items to play, stop, forward, and backward music. Each menu item is shown as an icon located on the top side of the camera view. In the virtual basketball game, a virtual ball is bouncing in a virtual cube space and the real video stream is shown in the background. A user can hit the virtual ball with his hand gestures. The virtual basketball game is shown in Fig. 1. The game player manipulates virtual ball in 3D space by means of hands in the video camera field of view to direct it into the basket.

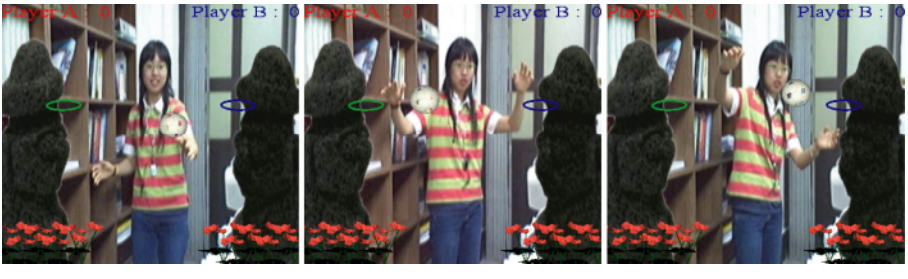

Fig. 1. The virtual basketball game using the bare-hand gesture-based interface

The experiments were performed on a 2.6GHz Pentium 4-based system. The frame size is fixed to 320x240. Color space conversion to HSV space and skin color region filtering requires 15 milliseconds on average per frame. Feature tracking for 500

feature points requires 42 milliseconds on average per frame. The overall processing speed for the interface is about 15 frames per second.

The menu activation accuracy was measured for three untrained users. Each user tried menu selection gestures a hundred times in front of the camera watching the monitor. One of them is requested to do them with fast hand motion. For normal speed gestures of two users, 93 trials and 95 trials were succeeded. For fast gestures of the last user, 84 trials were succeeded.

4 Conclusion

We described a barehanded interaction method using human hand gestures applicable to augmented reality applications to help in achieving the ease and naturalness desired for HCI. For the discrimination of intended gestures from accidental hand movements, we utilize a smooth trajectory toward one of virtual objects or menus. We have implemented and tested two simple augmented reality applications: a gesture-based music player and a virtual basketball game. The accuracy of menu activation is about 94% for normal speed gestures. We have still lots of room for improvement in accuracy. Obvious directions for future work for the improvement include applying the unified framework of various user contexts such as velocity of motion, patterns of hand gestures, current human postures, and human body silhouette. Typical applications of our hand gesture interface include remote control of electronic appliances, interaction for a virtual environment, input devices for wearable computers, and user control and response in 3D games.

Acknowledgments. This work was supported in part by grant No. RTI05-03-01 from the Regional Technology Innovation Program of the Ministry of Commerce, Industry and Energy(MOCIE) and in part by the Brain Korea 21 Project in 2006.

References

1. Pavlovic, V., Sharma, R., Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. IEEE Transactions on PAMI 19 (1997) 677–695
2. Quek, F., Mysliwiec, T., Zhao, M.: FingerMouse: A freehand computer pointing interface. In: Proc. of Int'l Conf. on Automatic Face and Gesture Recognition. (1995) 372–377.
3. O'Hagan, R., Zelinsky, A.: Finger track - a robust and real-time gesture interface. In: Australian Joint Conference on Artificial Intelligence. (1997) 475–484
4. von Hardenberg, C., Brard, F.: Bare-hand human-computer interaction. In: Proceedings of Perceptual User Interfaces. (2001) 113–120
5. Lenman, S., Bretzner, L., Thuresson, B.: Using marking menus to develop command sets for computer vision based hand gesture interfaces. In: NordiCHI '02: Proc. of the second Nordic conference on Human-computer interaction, ACM Press (2002) 239–242
6. Guimbretièere, F., Winograd, T.: Flowmenu: combining command, text, and data entry. In: Proc. of the ACM symposium on User interface software and technology. (2000) 213–216