Hand Gesture Recognition using Blob Detection for Immersive Projection Display System

Hasup Lee, Yoshisuke Tateyama, and Tetsuro Ogi

***Abstract*—**We developed a vision interface framework for an immersive projection system, CAVE in virtual reality research field using hand gesture recognition with computer vision techniques. A background image was subtracted from current image frame of webcam and we convert the color space of the image into HSV space. Then we mask skin regions using skin color range threshold and apply a noise reduction operation. We made blobs from the image and gestures were recognized using these blobs. Using our hand gesture recognition, we could implement an effective interface without bothering devices for CAVE.

***Keywords*—**CAVE, Computer Vision, Gesture Recognition, Virtual Reality.

I. Introduction

CAVE is one of famous device for virtual reality research. It has several projectors, screens, user input devices and etc. CAVE can give an immersion to users by surrounding them with VR contents. Photographs of real environment can be applied to this device to make more immersive contents. The idea of our technique came from real image-based rendering in computer graphics. A real image-based background modeling for CAVE system was made by us for user to feel more immersion in [1] [2] [3].

We add an affected interface to this system. To make such interface, bothering devices like game pad, glove and hat that are held or attached to body are excluded. A Standalone system like wireless network connected notebook, tablet pc is preferred for interface processing because of seamless feature of screens of CAVE. We used a game pad for navigation and manipulating for this but it has cord that is bothering for users. So we have developed gesture recognition interface using webcam and computer vision techniques. We developed gesture recognition interface using motion templates method [4]. It is one of computer vision algorithm robust to light change.

Our motion template based gesture recognition interface is natural and intuitive but cannot produce the exact meaning of gesture sometimes. So we developed gesture interface using hand and face blobs for exact results.

Hasup Lee is with Visual Simulation Laboratory, Graduate School of System Design and Management, Keio University, Yokohama, Japan (e-mail: hasups@sdm.keio.ac.jp).

Yoshisuke Tateyama is with Visual Simulation Laboratory, Graduate School of System Design and Management, Keio University, Yokohama, Japan (e-mail: tateyama@sdm.keio.ac.jp).

Tetsuro Ogi is with Visual Simulation Laboratory, Graduate School of System Design and Management, Keio University, Yokohama, Japan (e-mail: ogi@sdm.keio.ac.jp).

II. Hand Gesture Recognition

A. Framework

The background image was captured at the beginning. It was RGB format and converted into gray format for subtraction. After that, current frame was captured from a webcam. Current frame was resized into half for speed and flipped because mirror image is more intuitive. Using current frame and stored background image, background was subtracted from frame image.

Then we converted the color space of this image into HSV color space. The regions of face and hands were masked using skin color range. The result was binary image and opening operation was applied for noise reduction.

Blobs of face and hands were detected after that and we recognized hand gesture using them. A framework diagram for hand gesture recognition is shown in Fig. 1.

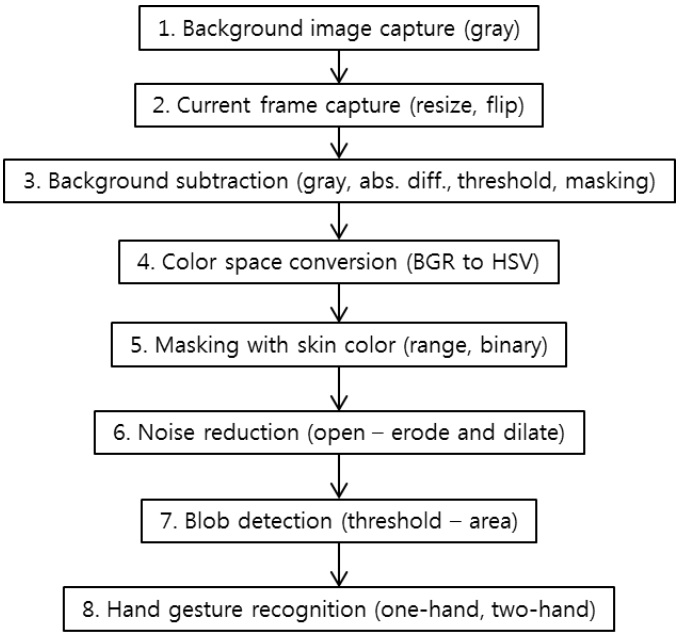

Fig. 1 Framework for hand gesture recognition

B. Background Subtraction

Background image was shot at the first stage and converted into gray scale. Then we subtracted it from gray converted current frame image. New image was taken from absolute value of this result. (|current frame – background|)

Then we made a mask using this subtracted result. The

binary mask was made by threshold and applied to current frame image. The background image is shown in Fig.2 and background subtracted image is in Fig. 3.

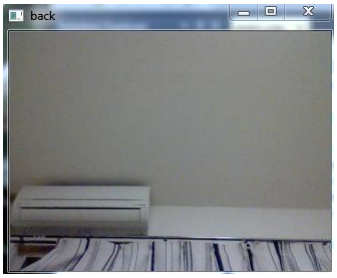

Fig. 2 Background frame image

Fig. 3 Background subtracted frame image

C. Color Space Conversion

HSV color space is relatively robust to light variation, so it is effective to detect skin. Skin color modeling and detection methods were well surveyed in [5]. A comparison of skin color modeling was studied in [6]. HSV(HS) color model produced best results in [6]. So we converted the color space of background subtracted frame image into HSV from RGB(BGR).

D. Masking with Skin Color

The region of hands and face can be extracted using skin color modeling. The range of skin color is defined in [7]. The maximum of each component is assumed as 1 and minimum is 0. The range of skin model is as followed.

$0 \le H \le 50,\ 0.20 \le S \le 0.68,\ 0.35 \le V \le 1.0$
$(0 \le h \le 35,\ 52 \le s \le 173,\ 90 \le v \le 255$ if max. is 255)

The result mask image is shown in Fig. 4. It is binary image and will be used as input for blob detection after noise reduction.

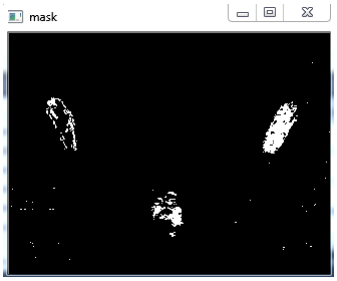

Fig. 4 Mask image with skin color range thresholds

E. Noise Reduction

Opening operation is used for noise reduction and consists of two steps. First, erode operation and dilate operation is applied. Opening operation does not change the size of the area of the object but blur does. The result image of opening operation is shown in Fig. 5.

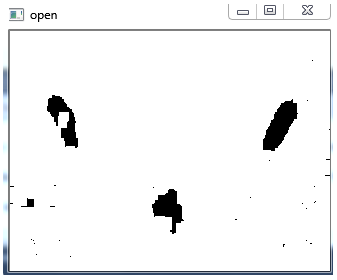

Fig. 5 Noise reduced frame image

F. Blob Detection

Blobs of face and hands were detected from skin masked and noise reduced frame image of Fig. 5. We used cvBlobsLib code library for blob detection. (cvBlobsLib - OpenCV Wiki, http://opencv.willowgarage.com/wiki/cvBlobsLib) Algorithm of this library is based on and improved from the component labeling with contour tracing of [8]. The result of blob detection is shown in Fig. 6(a).

Blobs of area of bigger than threshold were detected. Fig. 6(b) is the coordinates of image. The blobs are sorted by y-values and labeled b1, b2, b3 in ascending order.

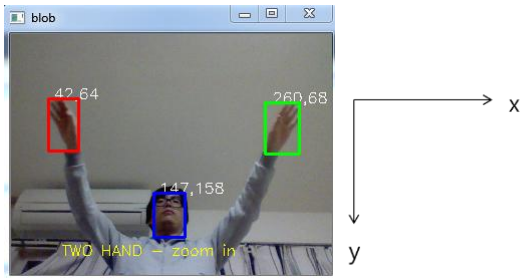

Fig. 6 (a) detected blobs and (b) coordinates

G. Gesture Recognition

To recognize gestures, we categorized gestures into two – one hand and two hand gesture. 'One hand gesture' is gesture by one hand and contains 'left' and 'right commands'. 'Two hand gesture' is gesture by two hands and contains 'zoom in' and 'zoom out commands'.

If y_{b1} is y-value of center of b1, y_{b2} is y-value of center of b2, y_{b3} is y-value of center of b3 and dy is some constant then algorithm of gesture specification is as follows (1).

//Gesture specification
If number of blob = 1 then
  No gesture
If number of blob = 2 then
  If $y_{b1} - y_{b2} > dy$ then 'one hand gesture' (1)

else no gesture
If number of blob = 3 then
If $y_{b1} - y_{b2} > dy$ then 'one hand gesture'
If $y_{b2} - y_{b3} > dy$ then 'two hand gesture'
else no gesture

'One hand gesture' only matters distance between b1 and b2 and 'two hand gesture' does b2 and b3. For example, all configurations in Fig. 7 are categorized into 'one hand gesture'.

Fig. 7 One hand gestures – (a) left hand and face blob (b) left hand and right hand + face blob (c) left, right hand and face blob

For 'one hand gesture', if $x_{b1}(n)$ is x-value of center of b1 in current frame, $x_{b1}(n-1)$ is x-value of center of b1 in previous frame and dx is some constant then we determined 'right' and 'left command' by following algorithm (2).

//One hand gesture
If $x_{b1}(n) - x_{b1}(n-1) > dx$ then 'right command' (2)
If $x_{b1}(n-1) - x_{b1}(n) > dx$ then 'left command'
Else no command

For 'two hand gesture', if $x_{b2}(n)$ is x-value of center of b2 in current frame, $x_{b2}(n-1)$ is x-value of center of b2 in previous frame and dw is some constant then 'zoom in' and 'zoom out command' are determined by following algorithm (3).

//Two hand gesture
$d(n) = | x_{b1}(n) - x_{b2}(n) |$
$d(n-1) = | x_{b1}(n-1) - x_{b2}(n-1) |$
If $d(n) - d(n-1) > dw$ then 'zoom in command' (3)
If $d(n-1) - d(n) > dw$ then 'zoom out command'
Else no command

Fig. 8 shows configurations of 'two hand gesture'. If user widen the distance between two hands, it is considered as 'zoom in commands' and if narrows, 'zoom out'.

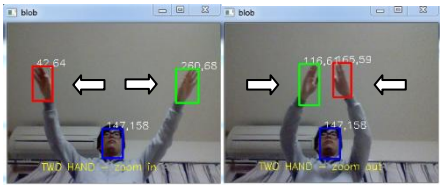

Fig. 8 Two hand gestures – (a) zoom in (b) zoom out command

III. Implementation

The implementation of hand gesture interface for CAVE is shown in Fig. 9. It was developed on our K-CAVE system [9] which has 4 screens, 8 stereo projectors and magnetic positioning sensor. It contains 5 Linux machines – 1 master and 4 renderer machines for each 2 stereo projectors. Sony Vaio™ Z-series notebook PC with built-in webcam was used for input and processing vision interface module. Frame images were captured from built-in USB webcam.

Fig. 9 Gesture recognition interface for CAVE

Vision module was developed using OpenCV library ver. 2.3.1 (OpenCV Wiki, http://opencv.willowgarage.com/wiki/). Vision module sent predefined commands to master machine via TCP protocols.

The constant parameters used in this paper are as follows. The resolution of camera input is 640x480 but frame image is resized to 320x240 for speed. The gray threshold value of background subtraction is 10. The area threshold value of blob detection is 200 pixel and dy, dx and dw of gesture recognition are 30, 10 and 10.

IV. Conclusion

In this paper, an interface for CAVE using hand gesture recognition was developed. We presented the framework for hand gesture recognition and it can be applied for other environment, too. It used built-in webcam of notebook PC so does not need high cost devices.

In future research, more complex scenario could be developed for interface. And multi modal devices like voice recognition could be included.

Acknowledgment

This work was supported by G-COE (Center of Education and Research of Symbiotic, Safe and Secure System Design) program at Keio University.

REFERENCES

[1] Hasup Lee, Yoshisuke Tateyama and Tetsuro Ogi, "Realistic Visual Environment for Immersive Projection Display System", The 16th International Conference on Virtual Systems and Multimedia, pp.128-132, October 2010.

[2] Hasup Lee, Yoshisuke Tateyama and Tetsuro Ogi, "Panoramic Stereo Representation for Immersive Projection Display System", The 9th International Conference on VRCAI (VR Continuum and Its Applications in Industry), pp.379-382, December 2010.

[3] Hasup Lee, Yoshisuke Tateyama and Tetsuro Ogi, "Image-based Stereo Background Modeling for CAVE System", International Symposium on VR innovation (ISVRI) 2011, March 2011.

[4] Hasup Lee, Yoshisuke Tateyama and Tetsuro Ogi, "Unaffected User Interface for CAVE using Motion Templates Method", International Journal of Control and Automation (ISSN: 2005-4297), To be published.

[5] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A survey of skin-color modeling and detection methods", Pattern Recogn. 40, 3 (March 2007), pp.1106-1122.

[6] Benjamin D. Zarit, Boaz J. Super, Francis K. H. Quek, "Comparison of Five Color Models in Skin Pixel Classification", International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-time Systems, pp. 58-63, September 1999.

[7] Y. Wang and B. Yuan, "A novel approach for human face detection from color images under complex background", Pattern Recognition 34(10) (2001) pp. 1983–1992.

[8] Fu Chang, Chun-Jen Chen, and Chi-Jen Lu, "A linear-time component-labeling algorithm using contour tracing technique", Comput. Vis. Image Underst. 93, 2 (February 2004), pp. 206-220.

[9] Y. Tateyama, S. Oonuki, S. Sato, and T. Ogi, "K-Cave demonstration: Seismic information visualization system using the OpenCABIN library," Proceedings of the ICAT 2008, pp. 363-364, 2008.