Space Gesture Recognition System Implementation

YANG-KEUN AHN, KWANG-SOON CHOI, YOUNG-CHOONG PARK
Korea Electronics Technology Institute
121-835, 8th Floor, #1599, Sangam-Dong, Mapo-Gu, Seoul
REPUBLIC OF KOREA
ykahn@keti.re.kr

Abstract: - This paper introduces a hand gesture recognition system that can detect both of a user's hands, using depth data generated through an infrared Time of Flight (TOF) method combined with a depth camera. Noise removal using expansion, erosion, and median filters, and hand recognition using integrated images, are applied to create a gesture recognition system that can detect both hands. Both hardware and software necessary for actual testing were gathered, and the recognition speed and accuracy were measured.

Key-Words: - Gesture Recognition, Depth Image, Time of Flight

1 Introduction

Developments in computer vision technology have increased interest in effective interfacing methods such as gesture recognition [1][2]. Hand-motion recognition [3][4] is a new interface method that is not restricted by device limitations, and remains natural to the end user.

Methods using RGB cameras and depth cameras have been attempting to achieve hand recognition. First, webcams were used as candidates, but though the sensor prices were cheap, they were far too sensitive to environmental changes. To overcome those downsides, infrared cameras were tried, but the binary nature of such images presented their own set of limitations.

The method presented here applies infrared depth cameras using a TOF method to generate the image, and using said image, find the user's hands within a predefined distance. Using a depth image minimizes the effect of changes in the environment, and allows for recognition to occur within a set range. This paper presents a method to perform hand recognition by removing noise from a depth image and then using the integrated image of the depth image. A system was set up to test the method.

2 Main Body

2.1 System Configuration

The system's hardware is as shown in Figure 1. The front of the TOF depth camera is aligned such that it is aimed towards the user. The user faces the camera, and performs gestures with arms extended. The user's hands are placed 80 to 90cm away from the camera.

The user extends two fists as seen in Figure 2 and gestures with them in angular directions clockwise and anti-clockwise.

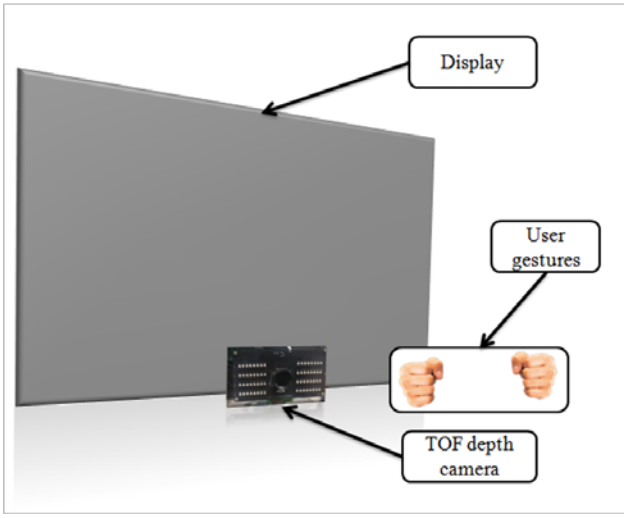

Fig. 1 System layout

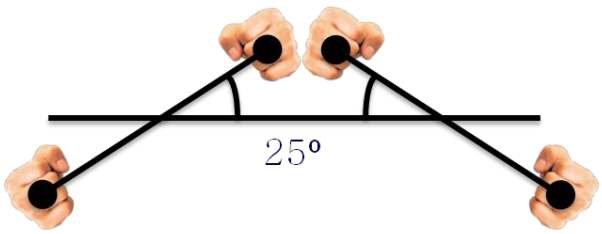

Fig. 2 Turning gestures using hands

2.2 Algorithm

The algorithm of the main system is divided as shown in Figure 3. During the 'Image generation' phase, the two infrared images taken from the TOF depth camera are generated. Noise removal is done by applying the median filter before the depth image is fully generated. This is because noise is present in both images, but they are independent of each other. To determine the depth value of the video taken when the infrared is on and off, pixel values are evaluated using Equation 1. Here Tx0 and Tx1 are pixel values for when the infrared is off and on, respectively.

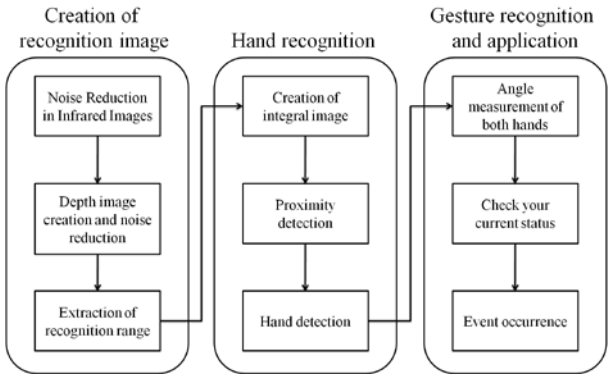

Fig. 3 Algorithm flowchart

Noise removal from infrared image – Depth image generation and noise removal – Recognition range extraction – Integrated image generation – Closest area detection – Hand detection – Angle measurement – Current state check – Event trigger

$$f = \frac{Tx0}{Tx0 + Tx1} \propto \frac{1}{Length} \quad (1)$$

Noise removal is performed again as the depth video is generated, where all areas below a predefined brightness are removed. This is because the reliability of the depth value decreases, the smaller the brightness value. Afterwards, expansion and erosion processes organize the depth video. Finally, the range at which recognition occurs is normalized, creating an 8-bit image as seen in Figure 4.

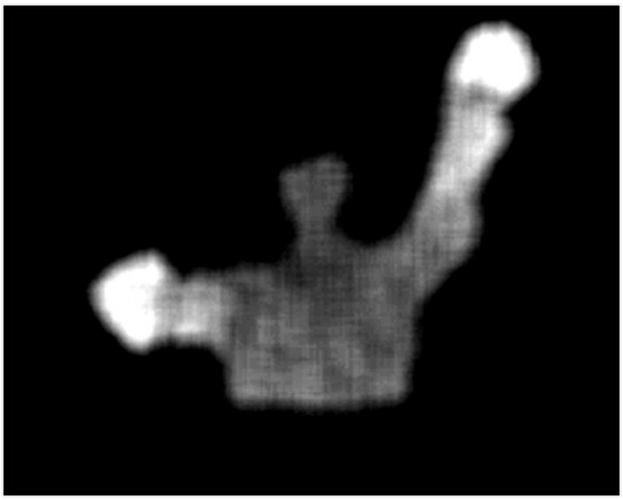

Fig. 4 8-bit depth image

Once the hand recognition phase begins, regions closest to the camera are sought in order to find the hands. This is based on the assumption that when a user extends his or her hands to perform gestures, the hands are likely to be closest to the camera. Using a single pixel value to find the closest region is unreliable, hence the mean value of a rectangular area is used. The size of the area is assigned a value inversely proportional to the depth value. Since calculating the sum of the rectangular area for a large number of pixels increases the computational load, to reduce the load, an integrated image is used. The creation and application method is as shown in Equation 2.

$$I(x,y) = Pixel(x,y) \text{ in } Original \text{ } Image$$
$$I'(x,y) = Pixel(x,y) \text{ in } Integral \text{ } Image$$
$$I'(X,Y) = \sum_{x<X,\, y<Y} I(x,y)$$
$$\sum_{x_1 \le x < x_2,\, y_1 \le y < y_2} I(x,y) =$$
$$I'(x_2,y_2) - I'(x_1,y_2) - I'(x_2,y_1) + I'(x_1,y_1) \quad (2)$$

Once the closest area is found, it remains to be determined whether the region corresponds to the user's hands. This paper uses the size of the regions to determine whether it is a hand or not. The two closest detected areas are binarized as shown in Figure 5, and the two largest of those areas are selected.

Fig. 5 Binarized image

The gesture recognition and application phase confirm that two hand inputs are present and calculates the angle they form. To calculate this angle, the center of each area is determined, and using Equation 3, the angle of the line between these two points is calculated.

$$r = atan\left(\frac{\triangle y}{\triangle x}\right) \quad (3)$$

When gestures are recognized, interactions can be performed by triggering keyboard events depending on the type of content.

2.3 Experimental Results

A hardware and software system was created to measure the speed and accuracy of the gesture recognition system. A PC fitted with an Intel i5 CPU, 2G of RAM, and 32 bit Windows 7 was used. First off, to measure recognition speed, the user was instructed to turn the hands clockwise and counter-clockwise 50 times each. Figure 6 is the measurement result, with a mean of 96.27ms. To test for accuracy, the user gestured by rotating his hands clockwise and counter-clockwise 100 times each. Table 1 details the measurement results, which were 97.5% accurate overall.

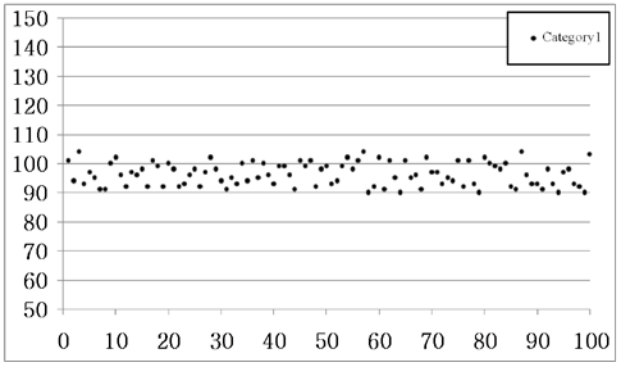

Fig. 6 Recognition speed graph

Table 1 Recognition accuracy measurement

Gesture	Count of input	Normal recognition count
LEFT	100	98
RIGHT	100	97
Total	200	195

3 Conclusion

This chapter describes the method of extracting fingertips from extracted hand-area information, and controlling the screen by means of fingertip trace information.

3.1 Extracting Fingertip Candidates

This paper presents an interfacing method to recognize gestures from two separate hands using a TOF depth camera. The generation of depth images and how they are used is explained, and using said depth images, an interactive system was created which can detect hands within a given distance range. A simple algorithm allowed for the creation of a system that could recognize a user's gestures. However, depth images are limited in their resolution and become less reliable at large distances, leaving much room for improvement.

References:

[1] I. F. Ince, M. S. Garzon, and T. C. Yang, "Hand Mouse: Real time hand motion detection system based on analysis of finger blobs", International Journal of Digital Technology and its Applications, Vol. 4, No. 2, pp. 40-56, 2010.
[2] S. Koepnick, R. V. Hoang, M. R. Sgambati, D. S. Coming, E. A. Suma, and W. R. Sherman, "RIST: Radiological Immersive Survey

Training for Two Simultaneous Users", Computers & Graphics Special Issue on Graphics for Serious Games, Vol.34, No. 6, pp. 665-676, 2010.

[3] Intel Corporation. Open Source Computer Vision Library reference manual. December 2000.

[4] Y. Hirobe, T.Niikura, Y. Watanabe, T. Komuro,M. Ishikawa, “Vision-based Input Interface for Mobile Devices with High-speed Fingertip Tracking,“ Adj. Proc. ACM UIST 2009, pp. 7-8.

[5] OpenCV 2.4.6.0 documentation. http://docs.opencv.org/