Motion Capture System Based On Color and Edge Distribution

Sheetal S Jadhav, Navnath D Kale

Abstract—This paper describes a robust tracking algorithm for real-time, video based motion capture systems. Since conventional video based motion capture systems use many video cameras and take a long time to deal with many video images, they cannot generate motion data in real time. On the other hand, the prototype system proposed in this paper uses a few video cameras, up to two, it employs a very simple motion-tracking method based on object color and edge distributions, and it takes video images of the person, e.g., x, y position of the hand, feet and head, and then it generates motion data of such body parts in real time. Especially using two video cameras, it generates 3D motion data in real time. This paper mainly describes its aspects as a real-time motion capture system for the tip parts of the human body, i.e., the hands, feet and head, and validates its usefulness by showing its application examples.

Keywords— background subtraction; camera; frame; moving objects detection

I. INTRODUCTION

This paper treats a real-time, video based motion capture system. Many researches on the motion capture system have been done so far because there has been a great demand of motion data for CG animation productions and 3D game productions. Conventional video based motion capture systems use many video cameras to obtain accurate, desired motion data so they cannot generate motion data in real time since it takes a long time to deal with many video images. Consequently it is impossible to use them **as** a real-time input device of human motion. Moreover such systems take much cost and require a huge working space so that they are not suitable as input device for a standard PC. In this paper, we propose a real-time, video based motion capture system using two video cameras. We employ a very simple motion-tracking method based on object color and edge distribution. Using this method, our current system is capable of tracking the tip parts of the human body, i.e., the hands, feet and head, and generates their motion data in real time. The system takes video image of tip parts of the human body from two video cameras and extracts its **x,** y position data per frame. Using this data from two video cameras, more it estimates *z* position and generates 3D motion data. Moreover our system also has a network communication
facility and it works **as** an input device dedicated for an interactive 3D graphics application runs on another computer.

Related work

If there are two systems connected with each other through the network, they can work collaboratively. Many researches on the video based motion capture system have been done so far [l]. These systems need many video cameras and many markers. Recently motion capture systems without using any markers have been studied **[2].** Their standard method of tracking the human motion is based on construction of 3D shape **as** voxel data from several silhouette images [3][4]. However, this process needs huge computation time. Some particular techniques and other constraints are necessary in order to reduce this computation time, Weik and Liedtke[S] proposed a hierarchical method for 3D pose estimation. The remainder of this paper is organized as follows. Section 2 explains system overview. Section 3 explains tracking algorithms. Section 4 introduces one of application examples. Finally, Section ***5*** concludes this paper.

II. SYSTEM OVERVIEW

The system hardware consists of a standard PC, two video capturing boards, a head-mounted display, and two video cameras as shown in Figure **1.** This hardware generates motion data by extracting person's image from each frame of video camera images and by computing the difference between a person's two adjoining images. Moreover, using ***x,*** y position from two images obtained by camera, it estimates *z* position of the body parts. This motion data is used as input data for other 3D applications running on the same computer. If a 3D graphics application exists on another computer, the motion data is sent to the computer using the network communication facility through the network.

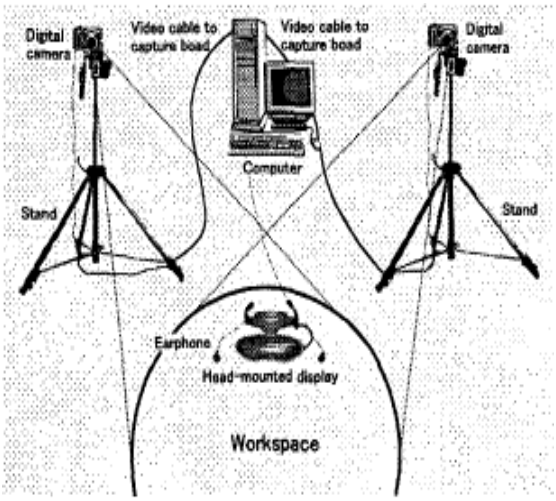

Figure 1: Hardware setting.

Manuscript received on April, 2013.

Sheetal S Jadhav, Computer Engineering Department, Pune University, Pune, India.

Navnath D Kale, Computer Engineering Department, Pune University, Pune, India.

III. TRACKING ALGORITHM

This section explains how to track the person's motion. Before tracking, the system requests an initializing process. And then the system starts the tracking process.

A. *Initializing process*

Before tracking the body, it is necessary to measure the location and orientation of each of two video cameras, and its fovy. During the tracking, these parameters are used to estimate 3D position data of the tracking area of a performer. The details are explained in subsection C. After this initial measurement, the system tracks the person's motion by extracting a person's image from each frame of video camera images and by computing the difference between a person's two adjoining images. Actually for tracking of this type, the system needs to store a background image excluding a person **as** another initial process. After storing the background, the system starts to track the motion. For each video frame in the tracking process, the system extracts the silhouette of a person by subtracting the stored background image from the current video image, and extracts a person's image using this silhouette as shown in Figure 2.

As explained in next subsection, the motion tracking is based on the color information, the system needs to store an initial state of the color information of a person's image. The system requests the user to perform his/her initial pose in order to obtain the color information of a person's each tracking area as shown in Figure 2.

B. *Tracking the body*

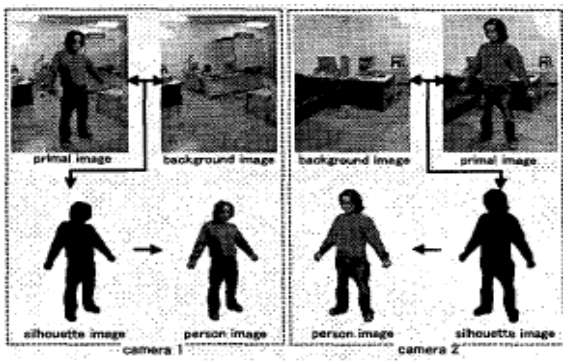

Figure 2: Person image extraction.

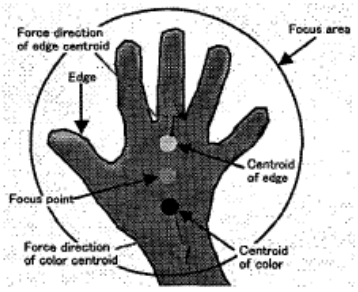

Figure 3: Tracking the hand.

The motion tracking is done based on the color information of each specific area of the body. Strictly speaking, the median point of the color information is used as the center of the corresponding focus area. It is calculated using the equation (1).

$$X_c = \frac{\sum_{i=1}^{m} X_c(i)}{m}, Y_c = \frac{\sum_{i=1}^{m} Y_c(i)}{m} \qquad (1)$$

where ***Xc,*** *Yc* are the centroid coordinates of the color distribution. ***Xc(i), Yc(i)*** are the X, Y coordinates of the i-th color point, and ***m*** is the number of color points. However, practically the color information is insufficient for tracking the motion robustly. For example, the color of the skin is uniformly distributed over the arm **as** shown in Figure 3. So if the user wants to track the hand, its color center is influenced by the arm color and it moves to the center of the arm area gradually. Consequently the system will loose the focus area. To compensate this weakness, we employ new measure concerning the edge distribution besides the color information. Similar to the color information, the median point of the edges, which are the contour pixels of a focus area, is used **as** the center of the area. It is calculated using the equation (2).

$$X_e = \frac{\sum_{i=1}^{n} X_e(i)}{n}, Y_e = \frac{\sum_{i=1}^{n} Y_e(i)}{n} \qquad (2)$$

Where ***Xe,*** *Y,* are the centroid coordinates of the edge distribution. ***Xe(i), Ye(i)*** are the X, Y coordinates of the i-th edge point and ***n*** is the number of edge points.

The edge centroid is always located on the upper part of the hand. So the system does not loose the focus area. However, the edge centroid is strongly influenced by the hand shape change. Therefore, we use weight values for both the color centroid and the edge centroid. As a result, the focus area becomes stable. The centroid of the focus area is calculated using the equation (3)

$$X_p = \frac{w_c X_c + w_e X_e}{w_c + w_e}, Y_p = \frac{w_c Y_c + w_e Y_e}{w_c + w_e} \qquad (3)$$

where *X,&* are the centroid coordinates of the focus area. ***we*** is the weight of the edge and ***wc*** is the weight of the color.

C. *3D motion data*

The motion data calculated by the above equations is in x, y position form. However, most 3D graphics applications need 3D position data, for instance, to manipulate a 3D object. As previously mentioned, to estimate 3D position data, it is necessary to measure some parameters concerning two video cameras. They are the location and orientation, i.e., ***X,,Y, ,Z ,,angle,*** for one camera and ***X,,*** *Y,* , ***Z,*** , ***angle,*** for the other camera. Moreover, *fovy*, and *fovyz* are necessary. During the tracking, two sets of x, y position data of a tracking area are obtained from two images get from video cameras.

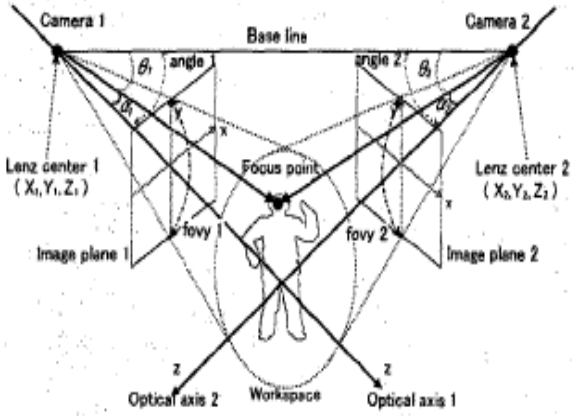

Figure 4: Estimating z position of focus point.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Then, using these values with *fovy*, and *fovy*, , ***a,*** and ***a,*** are calculated. Furthermore, **8,** is calculated from ***angle,*** and ***a,*** . Similarly, 8, is calculated from ***angle,*** and ***a2.*** Finally 3D position data is obtained from e,, e,, In this way, our system generates 3D motion data of some specific tracking areas of the human body.

IV. APPLICATION EXAMPLE

This section introduces one typical example of interactive virtual Reality applications. This application is developed using ***IntelligentBox*** **[7,8],** which is a constructive visual 3D software development system. This application provides a 3D virtual space in which the avatar controlled by a user can walk through using our prototype system **as** an input device, and communicates with other avatars controlled by other users. A camera is attached with the forehead of the avatar. This camera is not a real camera but also a software component provided by ***IntelligentBox.***

Figure 5: Head mounted display.

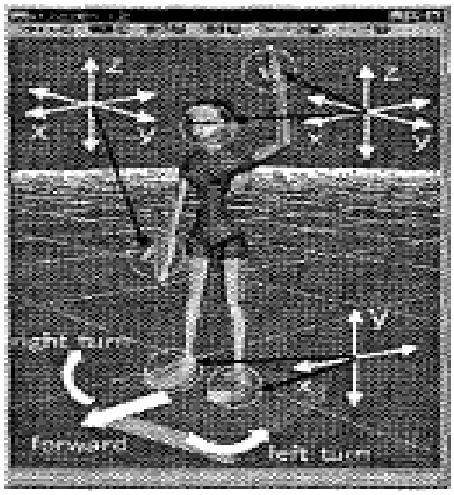

Figure 6: Avatar.

This camera is used to display the avatar's view image on a head-mounted display screen as show in Figure 5. Then by looking at this view image, the user can feel **as** if he is in the 3D virtual space and can operate the avatar efficiently. As shown in Figure 6, this avatar consists of 17 joints and it has many degrees of freedom. To make the avatar walk through and communicate with other avatars, we have to enter 13 values, i.e., three sets of **x,** y, z position data for feet. So the user uses his/her hands and head to control his/her avatar's hands and head **as** shown in Figure 7.

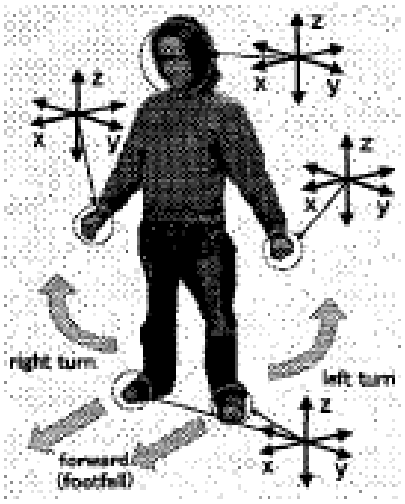

Figure 7: User control.

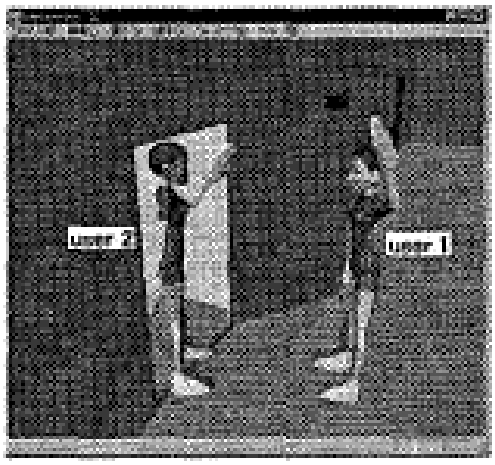

Figure 8: Communication with other user.

This correspondence between the user body motion and the avatar motion makes the user feel as if he/she was the avatar. This means good immersion. However as for the control of the body movement, it is too difficult to feel immersion. As you can understand easily, when the avatar walks, its location should change in a virtual space. But, in the real world, the user always should be located in the tracking space illustrated in Figure 1. So the user cannot change his/her location even if he/she acted like walk motion. In this application, we employ the virtual input device metaphor based on collision detection and motion capture proposed by Okada, et a1 **[9].** This concept enables the system to recognize any types of gestures, and then the user can control his/her avatar's body movement by two sets of **x,** y location data of his/her feet. Strictly speaking, when the user moves his/her foot forward, the avatar moves forward, and when the user moves his/her foot to left hand side, the avatar turns left/right. In this way, the avatar moves freely in the 3D virtual space by the user body motion.

Moreover, as previously mentioned, our system provides a network communication facility. Using this facility, multiple users can work collaboratively. For example, as shown in Figure 8, when some users use this application on different computers, each of them can control his/her own avatar simultaneously and communicate each other.

Finally **as** for the performance of our system, the sampling time, its resolution is 320x240 pixels, is at most 250 m sec on the standard PC (Pentium **IV** 2.0 ***GHZ,*** **1.5GB)** using two video cameras.

V. CONCLUSION

This paper proposed the real-time, video based motion capture system using only two video cameras. Since conventional video based motion capture systems use many video cameras and take a long time to deal with many video images, they cannot generate motion data in real time. On the other hand, our proposed system uses only **two** video cameras and generates 3D motion data in real time since our system employs a very simple tracking mechanism based on color and edge distributions of tracking focus areas. In this paper, we clarify usefulness of the proposed algorithm with the help of an application example. As future work, we will develop more application examples and evaluate their performance to improve our algorithm.

REFERENCES

1. D. M. Gravrila, "The visual analysis of human movement: A survey", CWR, vol. 73, pp. 82-98,1999
2. C. Wren, A. Azarbayejani, T. Darrel, and A. Pentland, "Pfinder: Real-time tracking of the human body", IEEE Trans. Pattern Anal. and Machine Intel., vol. 9, no. 7, pp.
3. D. Snow, P. Viola, and R. Zabih, "Exact voxel occupancy with graph cuts", in Proc. EEE CVPR, 2000
4. K. M. Cheung, T. Kanade, J. Y. Bouguet, and M. Holler, "A real-time system for robust 3D voxel reconstruction of human motions", in Proc. IEEE CVPR, 2000
5. **S.** We&, and C.-E. Liedtke, "Hierarchical 3D pose estimation for articulated human body models from a sequence of volume data", Robot Vision 2001, LNCS 1998,
6. Jason Luck, Dan Small, and Charles Q.Little, "Real-time tracking of articulated human models using a 3D shapefrom- silhouette method", Robot Vision 2001, LNCS 1998,
7. Okada, Y. and **Tanaka,** Y.: IntelligentBox: A Constructive Visual Sohare Development System for Interactive 3D Graphic Applications, Proc. of Computer Animation '95, IEEE Computer Society Press, pp.l14-125,1995.
8. Okada, Y. and Tanaka, Y.: Collaborative Environments of IntelligentBox for Distributed 3D Graphics Applications,The Visual Computer, Vol. 14, No. **4,** pp. 140-152, 1998.
9. Okada, Y. and Tanaka, Y.: Virtual Input Devices based on Motion Capture and Collision Detection, Computer Animation 1999, pp. 201-209, 1999.