2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics

3D Human Postures Recognition Using Kinect

Zheng Xiao, Fu Mengyin, Yang Yi and Lv Ningyi
School of Automation
Beijing Institute of Technology
Beijing, China
2120101104@bit.edu.cn (Zheng Xiao)

***Abstract*—In many application cases, 2D human postures display haven't been able to meet people's requirements which is failure to show human motions comprehensive, image and vivid. However, 3D human Postures display could restore and show human motions well, which is convenient for people to observe and learn human motions. This paper presents a method to recognize 3D human postures by using Microsoft Kinect sensor. Kinect is used as a capturing device. Capturing 3D human features mainly uses depth images obtained from Kinect sensor. Each pixel of depth images contains three-dimensional coordinate information of camera's scenes. Finally, the captured 3D human postures can be displayed by employing a human skeletal joints model and using a LED cube.**

***Keywords*—*3D human postures;* human motions; Kinect; depth images; human skeletal joints model; LED cube**

I. Introduction

Human motions are colorful, different motions often represent different meanings. In many application cases, such as behavior monitoring and motion analysis, human motions hope to be presented comprehensively, but it is difficult in 2D space. However, if we can recognize human postures in real-time and display recognition results in 3D space, we can restore human postures more accurately and vividly which is convenient for people to observe and learn human motions. For this objective, it is necessary to find a method to recognize human postures in 3D space. In this particular paper, a novel method is proposed for recognizing 3D human postures by using Kinect sensor.

Many researchers have proposed different methods for human postures recognition and reconstruction. Posture probability density [1] has been used to reconstruct human postures. This approach is very important at the early stage of the posture modeling. However, it's impractical utilizing motion capture data for the daily-life posture modeling for the entire human. Multi-camera system [2] has been used to recognize 3D human postures. This method requires at least three CCD cameras to work at the same time and needs a real-time background subtraction method to extract human silhouettes from color images accurately. Increasing the number of camera is necessary to improve the accuracy of 3D reconstruction which would aggravate the amount of computation and increase the cost. The method [3], called modeling of human postures using stereo camera, obtains various human postures automatically by using a stereo camera system. This method has proposed a technique for acquiring a human posture by fitting a human skeleton model to a recovered 3D human surface data. A stereo camera system which can be mounted on a robot is employed for 3D recovery. However, that presented human skeleton model only models the upper part of a human body. Obviously, it isn't enough only to recognize human postures of the upper body in practical application. Hence, a more compact device and a more thoughtful method need to be considered to acquire entire human postures.

Therefore, in this paper, we provide a novel method of recognizing 3D human posture by using Kinect. Kinect [4, 5] is a new game controller technology introduced by Microsoft in November 2010. Since its launch date it manifest great potential that it can't be only used in computer gaming but also many other applications like robotics and virtual reality. Kinect (Fig. 1) includes a RGB camera, a depth sensor and a multi-array microphone. It provides full-body 3D motion capture, facial and gesture recognition. The depth sensor consists of an infrared laser projector and a monochrome CMOS sensor. Kinect is capable to capture color and depth image at the same time. The resulting point cloud can be loaded to a computer using open source libraries which enable Kinect to be operated with Windows, Linux or Mac. Data connection to a computer is through USB interface.

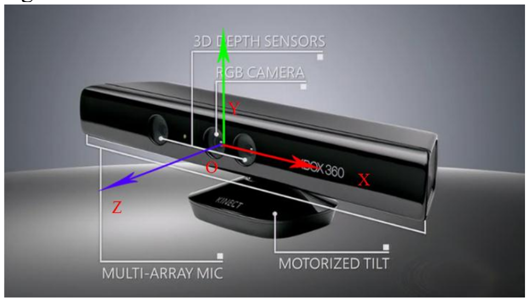

Figure 1. The Microsoft Kinect sensor. From left to right, the sensors shown are: the IR projector, the RGB camera, the monochrome camera used for depth computation.

II. Calibration of the Kinect Sensors

In order to use Kinect for depth measurement, one must calibrate it first. The most relevant data to this study are taken from Kinect's depth sensors. The depth camera returns an 11-bit number which needs further processing in order to exact the true depth from the sensor. According to the calibration procedure developed in [6, 7], the raw value d_{raw} of a point p in 3D can be defined as

$$d = K\tan(Hd_{raw} + L) - O$$

where d is the depth of that point in cm, $H = 3.5\times10^{-4} rad$, $K = 12.36cm, L = 1.18rad, O = 3.7cm$ .The tan approximation has a sum squared difference of $0.33cm^2$. According to distance of objects from Kinect depth camera, we set each pixel of false colored depth image different color (Fig. 2).

Once the depth has been obtained using the measurement above, we can recover the complete coordinate vector for point p in the Kinect camera frame. Let (i, j) be the coordinates (pixels) of the projection of point p onto the Kinect camera frame (Fig. 1). Let (x, y, z) be the coordinates of the 3D point p in the camera frame expressed in cm. In [8], the authors proposed the following equations to compute vector (x, y, z) from projection (i, j) and depth d for point p

$$x = (i - c_x)d / f_x$$
$$y = (j - c_y)d / f_y$$
$$z = d$$

where $f_x = 594.2143, f_y = 591.0405, c_x = 339.3078, c_y = 242.7391$. Finally, let us report some consideration on accuracy. In [8] the authors found that the accuracy for point p reconstruction from the Kinect depth camera is lower than 1 cm. For more details on the accuracy of measurements that one can get with this sensor we refer the reader to the Kinect node of the ROS project at MIT [9].

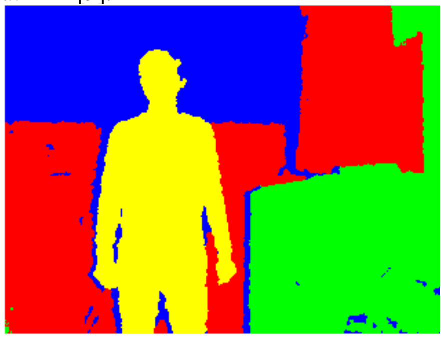

Figure 2. False colored depth image from Kinect depth camera.

III. HUMAN POSTURES RECOGNITION IN 3D

Kinect has been designed for Microsoft game console Xbox 360, it can be used to replace traditional game controller in many motion sensing games. In the applications, Kinect can capture human postures and use captured motions to control role of the games. In [10], the authors have presented working principles of real-time human pose recognition in parts from single depth images in detail. The method has been used in Kinect for Xbox to recognize human postures in motion sensing games. As we all know, the effect is extremely good. In [11], a new method of human detection using depth information by Kinect has been put forward. The authors have proposed a model based approach, which detects humans using a 2D head contour model and a 3D head surface model. In our research, we developed our method based on Kinect for Windows Software Development Kit (SDK) and proposed new ideas to optimize our program. The following parts contain principles and process of work in our research.

A. Body Part Labeling

The human body is capable of an enormous range of poses which are difficult to simulate. Instead, SDK capture a large database of motion capture of human actions. Our aim is to span the wide variety of poses people would make in any scenario. The database consists of approximately 500k frames in a few hundred sequences of driving, dancing, kicking, running, navigating menus, etc.

We have defined several localized body part labels that densely cover the body. Some of these parts are defined to directly localize particular skeletal joints of interest, while others fill the gaps or could be used in combination to predict other joints. Our intermediate representation transforms the problem into one that can readily be solved by efficient classification algorithms. The parts are specified in a texture map that is retargeted to skin the various characters during rendering. The pairs of depth and body part images are used as fully labeled data for learning the classifier. In this paper, we use 31 body parts LU/RU/LW/RW head, neck, L/R shoulder, LU/RU/LW/RW arm, L/R elbow, L/R wrist, L/R hand, LU/R-U/LW/RW torso, LU/RU/LW/RW leg, L/R knee, L/R ankle, L/R foot (Left, Right, Upper, Lower). Distinct parts for left and right allow the classifier to disambiguate the left and right side of the body.

Of course, the precise definition of these parts could be changed to suit particular application. For example, in an upper body tracking scenario, all the lower body parts could be merged. Parts should be sufficiently small to accurately localize body joints, but not too numerous as to waste capacity of the classifier.

B. Depth Image Features

We employ simple depth comparison features. At a given pixel x, the features compute

$$f_\theta(I, x) = d_I\left(x + \frac{u}{d_I(x)}\right) - d_I\left(x + \frac{v}{d_I(x)}\right), (1)$$

where $d_I(x)$ is the depth at pixel x in image I, and parameters $\theta = (u, v)$ describe offsets u and v. The normalization of the offsets by $1/d_I(x)$ ensures the features are depth invariant: at a given point on the body, a fixed world space offset will result whether the pixel is close or far from the camera. The features are thus 3D translation invariant (modulo perspective effects). If an offset pixel lies on the background or outside the bounds of the image, the depth probe $d_I(x')$ is given a large positive constant value.

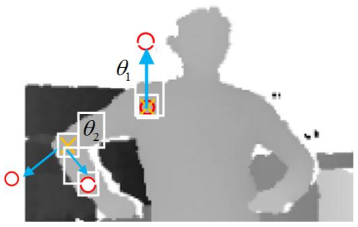

Figure 3. Depth image features. The yellow crosses indicate the pixel x being classified. The red circles indicate the offset pixels as defined in Eq. 1.

Fig. 3 illustrates two features at different pixel locations x. Features f_{θ_1} looks upwards: Eq. 1 will give a large positive response for pixels x near the top of the body, but a value close to zero for pixels x lower down the body. Features f_{θ_2} may instead help find thin vertical structures such as the arm.

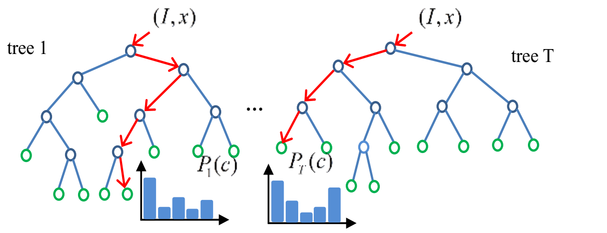

Figure 4. Randomized decision forests. .A forest is an ensemble of trees. Each tree consists of split nodes (blue) and leaf nodes (green). The red arrows indicate the different paths that might be taken by different trees for a particular input.

C. *Randomized Decision Forests*

Randomized decision trees and forests have proven fast and effective multi-class classifiers for many tasks. As illustrated in Fig. 4, a forest is an ensemble of T decision trees, each consisting of split and leaf nodes. Each split node consists of a feature f_θ and a threshold τ . To classify pixel x in image I, one starts at the root and repeatedly evaluates Eq. 1, branching left or right according to the comparison to the threshold τ . At the leaf node reached in tree t, a learned distribution $P_t(c/I,x)$ over body part labels c is stored. The distribution are averaged together for all trees in the forest to give the final classification

$$P(c/I,x)=\frac{1}{T}\sum_{t=1}^{T}P_t(c/I,x),(2)$$

Training. Each tree is trained on a different set of randomly synthesized images. A random subset of 2000 example pixels from each image is chosen to ensure a roughly even distribution across body parts. Each tree is trained using the following algorithm:

1. Randomly propose a set of splitting candidates $\phi=(\theta,\tau)$ (features parameters θ and thresholds τ).
2. Partition the set of examples $Q=\{(I,x)\}$ into left and right subsets by each ϕ:
$$Q_l(\phi)=\{(I,x)/f_\theta(I,x)<\tau\},(3)$$
$$Q_r(\phi)=Q\setminus Q_l(\phi),(4)$$
3. Compute the ϕ giving the largest gain in information:
$$\phi^*=\underset{\phi}{\operatorname{argmax}}\,\mathrm{G}(\phi),(5)$$
$$\mathrm{G}(\phi)=H(Q)-\sum_{S\in\{1,r\}}\frac{|Q_s(\phi)|}{|Q|}H(Q_S(\phi)),(6)$$

where Shannon entropy $H(Q)$ is computed on the normalized histogram of body part labels $l_I(x)$ for all $(I,x)\in Q$.

4. If the largest gain $G(\phi*)$ is sufficient, and the depth in the tree is below a maximum, then recurse for left and right subsets $Q_l(\phi^*)$ and $Q_r(\phi^*)$.

D. *Joint Position Proposals*

Body part recognition as described above infers per-pixel information. This information must now be pooled across pixels to generate reliable proposals for the positions of 3D skeletal joints. These proposals are the final output of our algorithm, and could be used by a tracking algorithm to self-initialize and recover from failure.

A simple option is to accumulate the global 3D centers of probability mass for each part, using the known calibrated depth. However, outlying pixels severely degrade the quality of such a global estimate. Instead we employ a local mode-finding approach based on mean shift with a weighted Gaussian kernel.

We define a density estimator per body part as

$$f_c(\hat{x})\propto\sum_{i=1}^{N}w_{ic}\exp(-\left\|\frac{\hat{x}-\hat{x}_i}{b_c}\right\|^2),(7)$$

where $\hat{x}$ is a coordinate in 3D world space, N is the number of image pixels, w_{ic} is a pixel weighting. $\hat{x}_i$ is the reprojection of image pixel x_i into world space given depth $d_I(x_i)$, and b_c is a learned per-part bandwidth. The pixel weighting w_{ic} considers both the inferred body part probability at the pixel and the world surface area of the pixel:

$$w_{ic}=P(c/I,x_i)\cdot d_I(x_i)^2,(8)$$

This ensures density estimates are depth invariant and gave a small but significant improvement in joint projection accuracy. Depending on the definition of the body parts, the posterior $P(c/I,x)$ can be pre-accumulated over a small set of parts. A skeletal joints model of entire human body employed in this research is shown in Fig. 5. The skeletal joints model has 20 skeletal joints and 19 bones [12].

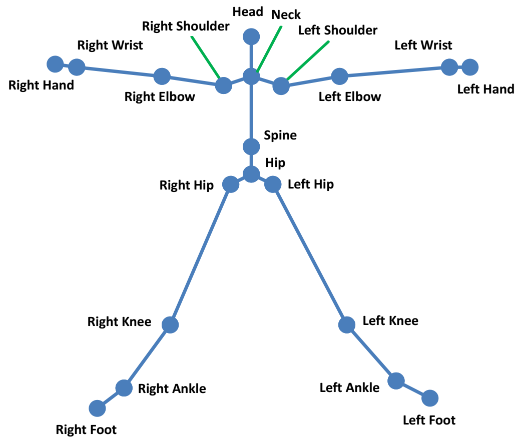

Figure 5. Skeletal joints model of entire human body.

IV. Experiments

A. LED Cube

In order to display 3D human postures recognized by Kinect, we have developed a LED cube(Fig. 6). The LED cube is able to display any patterns we wish in 3D by lighting or extinguishing correlative LEDs.

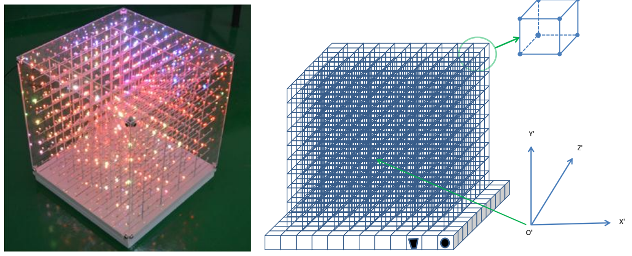

Figure 6. LED cube. The LED cube consists of 1000 LEDs. Their distribution is $10\times10\times10$. Each LED can not only emit blue, red and green color light, but also any mix color lights of them.

B. Transformation Of Coordinates

According to the skeletal joints model of entire human body (Fig .5), we can obtain 20 skeletal joints of a captured human. The joints contain physical coordinates in space coordinates $O-XYZ$ (Fig .1). To display human postures using the LED cube, we need to transform the joints' coordinates into space coordinates $O'-X'Y'Z'$ (Fig. 6). Because of distortion of human postures, the following formula should be satisfied

$$\frac{\Delta x}{\Delta x'}=\frac{\Delta y}{\Delta y'}=\frac{\Delta z}{\Delta z'},(9)$$

To the nearest whole number is inevitable during coordinates transforming for the arrangement of the LEDs is discontinuous. The accuracy of the human postures displayed using LED cube depends on the arrangement density of LEDs. Appropriate modifies have been done to make the human postures more vivid and image. The LEDs around head joint and foot joints are lighted to highlight human features.

C. Experimenal Results

The experimental results are shown in Fig. 7.

V. CONCLUSIONS

In this paper, we propose a method of recognizing 3D human postures by using Kinect. This method can measure human body position in physical coordinates and recognize 3D human postures accurately. Moreover, human features can be extracted according to depth image and human skeletal joints model. This method can be used for robot control, behavior monitoring and driving behavior learning.

Acknowledgment

This research was being carried out at Integrated Navigation & Intelligent Navigation Lab, Beijing Institute of Technology, Beijing, China. This research was a part of Beijing Natural Science Foundation, no. 4101001 "Research on Key Techniques of Urban Traffic Complex Network Based on Driving Behavior Cognition".

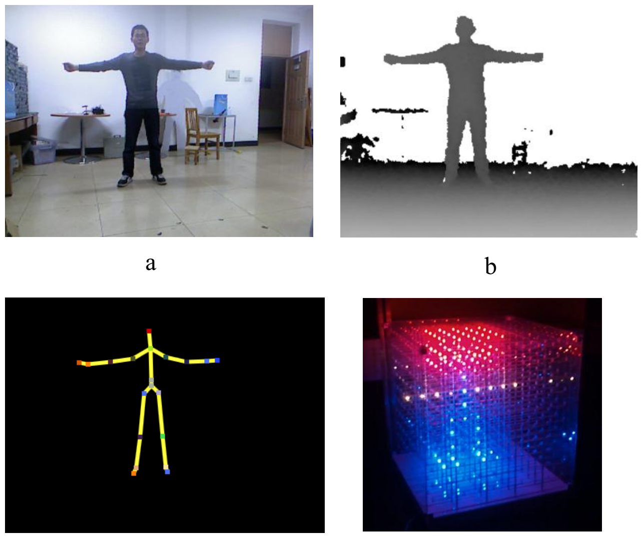

Figure 7. Experimental results. (a) RGB image, (b) Depth image, (c) Human skeleton image, (d) Human postures displayed using LED cube.

References

[1] Harada, T. ; Sato, T. ; Mori, T. ; "Human Posture Reconstruction Based On Posture Probability Density", Graduate Sch. of Inf. Sci. & Technol., Tokyo Univ., Japan, pp. 4063-4070, December, 2005.

[2] Takahashi, K. ; Nagasawa, Y. ; Hashimoto, M. ; "Remarks on 3D Human Posture Estimation System Using Simple Multi-Camera System", Fac. of Eng., Doshisha Univ., Kyotanabe, pp. 1962-1967, July, 2007.

[3] Yoshida, S. ; Joo Kooi Tan ; Hyoungseop Kim ; Ishikawa, S. ; "Modeling of Human Postures Using Stereo Camera", Grad. Sch. of Eng., Kyushu Inst. of Technol., Fukuoka, Japan, pp. 1432-1435, October, 2010.

[4] Stowers, J.; Hayes, M.; Bainbridge-Smith, A.; "Altitude Control of a Quadrotor Helicopter Using Depth Map from Microsoft Kinect Sensor", Electr. & Comput. Eng., Univ. of Canterbury, Christchurch, New Zealand, pp. 358-362, April, 2011.

[5] Frati,V. ; Prattichizzo,D. ; "Using Kinect for hand tracking and rendering in wearable haptics", Dipt. di Ing. dell'Inf., Univ. di Siena, Siena, Italy, pp. 317-321, June, 2011.

[6] ROS (MIT), Ros kinect calibration. http://www.ros.org/wiki/ kinect_calibration/technical, 2010.

[7] Open kinect imaging information. http://openkinect.org/wiki/Imaging _ Information, 2011.

[8] Matthew Fisher, Kinect study. http://graphics.stanford. edu/~mdfisher/Kinect.html, 2012

[9] ROS (MIT). Ros implementation of hand tracking. http://www.ros.org/ wiki/kinect_tools, 2011.

[10] Shotton,J. ; Fitzgibbon,A. ; Cook, M. ; Sharp, T. ; Finocchio, M. ; Moore, R. ; Kipman, A. ; Blake, A. ;"Real-time human pose recognition in parts from single depth images", Microsoft Research Cambridge & Xbox Incubation, pp. 1297-1304, August, 2011.

[11] Lu Xia ; Chia-Chih Chen ; Aggarwal, J.K. ; "Human detection using depth information by Kinect", Dept. of Electr. & Comput. Eng., Univ. of Texas at Austin, Austin, TX, US, pp. 15-22, October, 2011.

[12] Kinect for Windows SDK beta. Programming Guide. "Getting Started with the Kinect for Windows SDK Beta from Microsoft Research", pp. 19-20, July, 2011.