2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance

Real-Time 3D Human Pose Estimation from Monocular View with Applications to Event Detection and Video Gaming

Shian-Ru Ke, LiangJia Zhu,
Jenq-Neng Hwang
Department of Electrical Engineering
University of Washington, Box 352500
Seattle, WA 98195, USA
{srke, zhulj, hwang}@u.washington.edu

[1]Hung-I Pai, [1]Kung-Ming Lan, [2]Chih-Pin Liao
[1]Identification and Security Technology Center
[2]Human-Computer Interactive Technology Center
Industrial Technology Research Institute
Chutung, Hsinchu, Taiwan 31040, R.O.C.
{HIPai, blueriver,
liaochihpin.528822}@itri.org.tw

Abstract

We present an effective real-time approach for automatically estimating 3D human body poses from monocular video sequences. In this approach, human body is automatically detected from video sequence, then image features such as silhouette, edge and color are extracted and integrated to infer 3D human poses by iteratively minimizing the cost function defined between 2D features derived from the projected 3D model and those extracted from video sequence. In addition, 2D locations of head, hands, and feet are tracked to facilitate 3D tracking. When tracking failure happens, the approach can detect and recover from failures quickly. Finally, the efficiency and robustness of the proposed approach is shown in two real applications: human event detection and video gaming.

1. Introduction

Reconstructing 3D human poses from a monocular view has broad applications in human-computer interface, virtual reality and video surveillance. However, great challenges are still remained in automatically estimating 3D human poses from video sequences, due to the intrinsic depth ambiguity from monocular view, the large variations of human poses in motion, and the high degree of freedom (DOF) of articulation. Many approaches have been proposed to solve these problems [1], which can be categorized as either model-based or model-free ones. In model-based approaches, a 3D human body model is explicitly used, and the best fit is estimated between the model and image features derived from 2D image frames. A data-driven model-based approach is proposed in [2], where body components detection and human kinematics model are integrated to estimate 3D poses. Stochastic sampling, another model-based scheme, is applied in [3] to resolve the ambiguities and to escape from the local minima. In [4], dynamic models are incorporated into the human body tracking process to get a full 3D reconstruction of golf swing motion sequences. Conversely, in model-free approaches, the data mapping between 3D poses and 2D image features are directly learned. Different features such as contour [5] and edge [6] are commonly used as feature vectors that can robustly reflect human pose variations.

However, much less researches have been done in applications with real-time constraints for automatically tracking 3D human poses. Recently, a learning-based method is presented in [7], where 3D poses are estimated by hierarchically evaluating observation likelihood from image silhouettes. But learning-based methods may fail due to wide variations of poses and external parameters. Although a more flexible algorithm is given in [8] for monocular multi-cues tracking by using a loose-limbed body model and particle filters interacting through belief propagation on a factor graph, it cannot meet the real-time requirement.

In this paper, we extend our prior work [12] and present an approach for automatically estimating 3D human poses from monocular videos with real-time and robust performance. We follow an *analysis-via-synthesis* strategy and decompose human body in different parts [9]. Only body parts that have been detected as moving are tracked by using multiple cues such as silhouette, edge and color. In the tracking stage, results of 2D feature (the location of head, hands, and feet) tracking are integrated with 3D tracking to improve the robustness and efficiency of our approach. In addition, tracking failures are well-handled so that the approach is less sensitive to accumulated tracking errors over long video sequences.

The paper is organized as follows: Section 2 introduces the overview of the proposed system whose modules are discussed in the following sections. Section 3 discusses the human body detection module. The 2D feature extraction module is introduced in Section 4. Then, 2D feature tracking is mentioned in Section 5. In Section 6, the mechanism of 3D tracking is discussed. The result of validation is presented in Section 7. Then the experimental

978-0-7695-4264-5/10 $26.00 © 2010 IEEE
DOI 10.1109/AVSS.2010.80

results and two applications are shown in Section 8. Finally, a short conclusion and future work are discussed in Section 9.

2. Overview of Proposed System

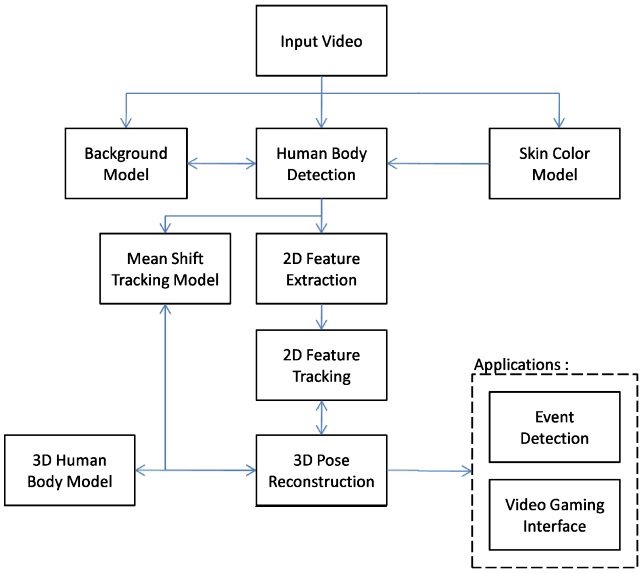


Figure 1: Overview of the proposed 3D system.

The overview of the proposed system is shown in Figure 1, including human body detection, 2D feature extraction, 2D feature tracking and 3D pose reconstruction modules. Initially, a Gaussian model is built for each location (x, y) in the image plane from a few background frames. Then, the human body detection module is triggered to detect whether a person enters into scenes by skin color model so that the person as the foreground is segmented. If a person is detected, the 2D feature extraction module will extract 2D features of the foreground including silhouette, edge and skin, and mean shift tracking model [13, 14] (more details in Section 6.3) will be appropriately initialized. Moreover, face, hands and feet are tracked in 2D feature tracking module. According to those 2D features, a 3D human body model is initialized and 3D poses of each decomposed body part in each frame are reconstructed by using downhill simplex search algorithm [11] to minimize the cost between the projected 2D features from 3D poses and the extracted 2D features from video frames. The tracking lost and occlusion events are also handled at the 3D pose reconstruction module. Finally, the results are sent to the two proposed applications: one is to detect and recognize various surveillance events such as lifting a bag. The other is for 3D video game driven by the 3D tracking results.

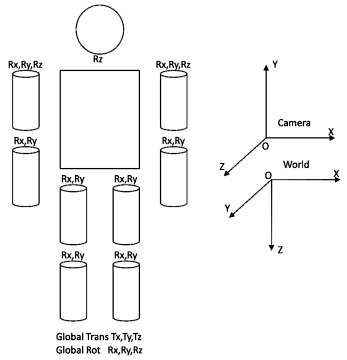


Figure 2: A simplified 3D human body model.

In Figure 2, the human body model is composed of torso, limbs and head. Four limbs are represented by cylinders, torso by 2D rectangle and head by circle, with 25 degree of freedoms (DOFs) for poses (global translations, rotations and joint angles) and 15 DOFs for shapes (the length and width of each body part). The orthographic camera model is used for 3D pose reconstruction and transformed into the world coordinates.

3. Human Body Detection

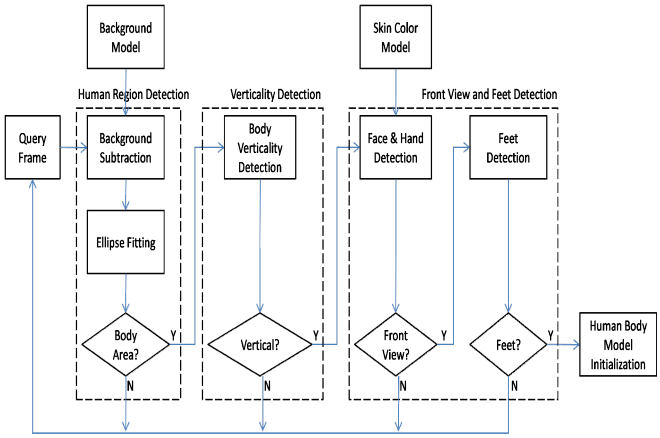


Figure 3: The flow chart of human body detection.

For better initialization of a 3D body model, the human body is initially assumed to enter the camera view in a predefined region of interest (ROI) with the full body, stand vertically with hands naturally hanging down, and face the camera. This assumption is currently being relaxed based on 2D body tracking, which will be described in details in other submission. The flow chart of human body detection is shown in Figure 3. Firstly, human region is extracted in each frame by using a background subtraction method. Once human region is extracted and the verticality is checked successfully, the skin blobs of the face and hands are classified through a skin color model. Finally, feet are detected by the gradient of the input frame.

3.1. Human Region Detection

At the step of human region detection, the foreground is extracted from the input frame by using background subtraction method with the learned background model. Threshold $(\mu_i(x,y)-\sigma_i(x,y)+M)$ is determined by the mean $\mu_i(x,y)$ and standard deviation $\sigma_i(x,y)$ of the difference between the frame and the background image with the empirical parameter M (=10 in our experiments). The pixels of any channel value $c_i(x,y)$ with $c_i(x,y)>\mu_i(x,y)-\sigma_i(x,y)+M$ are labeled as foreground pixels. Then, a Gaussian model is used to smooth the foreground pixels.

Moreover, foreground pixels are grouped as foreground regions and the largest foreground region is selected as the human body region at the ellipse fitting step. Then the parameters such as ellipse width, height, rotation angle and proportion of the human body region are used to verify possible human body occupied region.

3.2. Verticality Detection

The possible human body region is equally separated into 6 sub-regions shown in Figure 4. The standard deviation of the centroids of sub-regions along vertical direction is calculated. If the standard deviation is smaller than a predefined threshold, the person is regarded as standing vertically.

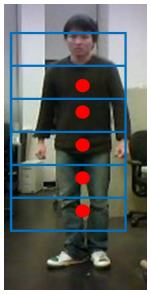


Figure 4: The detection of human standing vertically.

3.3. Front View & Feet Detection

The skin regions of the foreground are segmented based on skin color model (the skin pixel detection will be introduced in next section). Too large or too small skin regions are disregarded. The valid largest three ones are selected as candidates for face and hands. Then the distance between left/right hands and face, d_1 or d_2, is computed. The distance ratio, $r_d=\frac{\min(d_1,d_2)}{\max(d_1,d_2)}$, is used to check for possible geometrical configuration of the initial face-hand position. If r_d lies within the predefined range, one face blob and two hand blobs are validly found.

For feet detection, firstly, a reference point (cx,cy) is defined, where cx is the x component of the face blob and cy is the mean value of the y component of two hand blobs. Two tip points (x_0,y_0) and (x_1,y_1) are searched in the lower part of gradient of the current frame with the largest Euclidean distance from (cx,cy) and its x component being greater or less than cx. If $abs(y_0-y_1)$ is smaller than a predefined constant, then the detected two points, (x_0,y_0) and (x_1,y_1), are selected as feet positions as shown in Figure 5.

Figure 5: Identification of feet locations.

4. 2D Feature Extraction

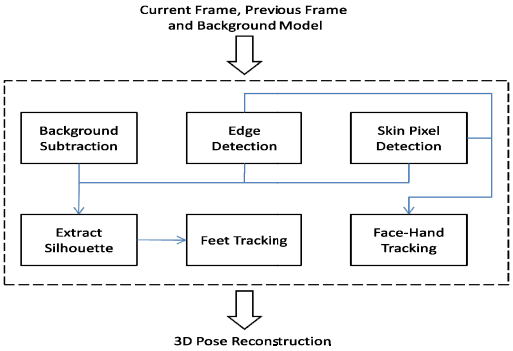


Figure 6: 2D features including edge, skin pixel and silhouette are extracted and used for face-hand and feet tracking.

After the human body is detected, 2D features including edge, skin pixel and silhouette are extracted and used for face-hand and feet tracking (see Figure 6).

4.1. Edge Detection

Firstly, the first order derivative images of the current frame in X and Y directions are calculated by using the Sobel operator. Then, the two resulting derive images are subtracted by two background derivative images, separately. Moreover, the gradient magnitude image is then computed from the subtracted derivative images. Finally, the resulting image is thresholded to obtain the edge image I_{Edge} of the human body.

4.2. Skin Pixel Detection

Skin color is classified in a specified region in R-G color space, that is, if the color satisfies $t_1 < r - g < t_2$, where r and g denotes the red and green color channels separately, and t_1 and t_2 are predefined thresholds, it will be classified as skin color [10].

The skin image I_{Skin} is defined as $I_{Skin0} \cap (I_{BSkin} \cup I_{Mot} \cup I_{Sub})$, where I_{Skin0} is the original classified foreground skin pixel image, I_{BSkin} is the background skin image, I_{Mot} is the motion image (difference image) between current frame and previous frame, and I_{Sub} is the background subtracted image.

4.3. Silhouette Extraction

The silhouette image I_{Sil} is defined as $I_{Sub} \cup I_{Edge} \cup I_{Skin}$. An example of silhouette extraction is shown in Figure 7.

Figure 7: The silhouette image I_{Sil} is composed as $I_{Sub} \cup I_{Edge} \cup I_{Skin}$.

5. 2D Feature Tracking

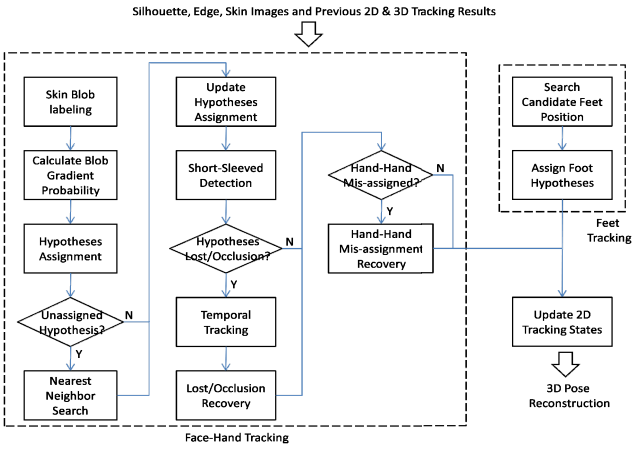


Figure 8: steps of 2D feature tracking.

5.1. Skin Blob Labeling and Gradient Probability

First of all, as shown in Figure 8, the connected areas in I_{Skin} are labeled as skin blobs. For each skin blob B_k, the blob gradient probability is calculated based on (1), where the normalization term $N(B_k)$ is the number of pixels in B_k. The blobs with low p_{sg} are eliminated.

$$p_{sg}(B_k) = \frac{\sum_{(x_i,y_i) \in B_k} I_{Edge}(x_i, y_i)}{N(B_k)} \tag{1}$$

5.2. Image Matching Cost Computation

Image matching cost $C_{img}(i,j)$ is computed based on (2). As an example in Figure 9, the current maximum upper and lower arm lengths are computed first. Then, the image cost is computed in the valid search interval sampled with uniform step for all possible configurations. Finally, the maximal one is selected as $C_{img}(i,j)$:

$$C_{img}(i,j) = \max_l \left\{ \frac{\sum_{(x_p,y_p) \in CF(i,j,l)} I_{Edge}(x_p, y_p)}{len(CF(i,j,l))} \right\} \tag{2}$$

where $CF(i,j,l)$ is one possible 2D body part configuration, and $len(CF(i,j,l))$ is its length.

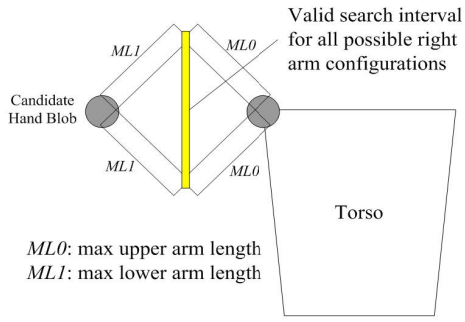


Figure 9: Image matching cost computation.

5.3. Nearest Neighbor Search

The nearest neighbor is searched along the path from $(0,0)$ to (M_H, M_B), which has the maximal cost $C(i,j)$ as (3), where M_H, M_B are maximum number of lost hypotheses and unassigned blobs. The overall cost $C(i,j)$ of assigning a blob B_j to a hypothesis H_i is defined as (4), where the image matching cost $C_{img}(i,j)$ and the blob gradient probability $p_{sg}(j)$ are as defined in (1) and (2). Besides, the maximum moving distance D_i at frame k is defined in (5).

$$\max_{i,j} C(i,j) \tag{3}$$

$$C(i,j) = \exp\left(-\frac{d(H_i, B_j)}{D_i}\right) * C_{img}(i,j) * p_{sg}(j) \tag{4}$$

$$D_i(k) = \begin{cases} D_i(k-1)*\alpha & ,if \quad H_i(k-1) \quad is \quad not \quad lost \\ D_i(k-1)*L_i(k-1)*\alpha & ,if \quad H_i(k-1) \quad is \quad lost \end{cases} \tag{5}$$

where $L_i(k-1)$ is the number of consecutively lost tracking frames as measured at frame $k-1$, and α is a predefined constant.

5.4. Short-Sleeved Detection

For the special case of human wearing short-sleeves, the short-sleeved event is detected if the area of the skin blobs of hands is greater than a predefined threshold. Once short-sleeved scenario is triggered, the areas of hands skin blobs of hands are weighted by a factor *f*, which is empirically set as the area of head/10 and the centers of the skin blobs of hands are moved downward according to the proportion of the radius of the hand skin blob to the length of the arm. In this way, the centroids of hand blobs can represent the fist locations for long-sleeved and short-sleeved cases.

5.5. Temporal Tracking

The temporal tracking checks for the current 2D tracking states, which are defined as ACTIVE, LOST or OCCLUSION. Then, a temporal tracking list for lost/occluded hypotheses is created. For each candidate Tr_i in the tracking list, the unassigned blob B_j with shortest Euclidean distance to Tr_i is assigned to Tr_i, and the unassigned blob B_j is deleted.

5.6. Lost/Occlusion Recovery

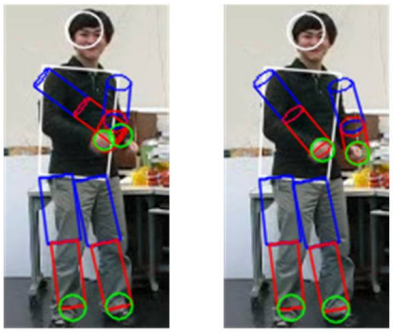
A valid candidate is an object in temporal tracking list with the number of consecutive tracking frames greater than $MINTRACKFRM * (1 - p_{sg})$, where $MINTRACKFRM$ is a constant that controls the minimum number of tracking frames required for selecting a valid candidate. The cost of assigning a lost/occluded hypothesis H_i to a candidate Tr_j is defined as (6). Finally, a pair is selected from $(0,0)$ to (M_H, M_{TR}) that maximizes the cost $C(i,j)$, where M_H, M_{TR} are maximum number of lost hypotheses and candidates in the temporal tracking list.

$$C(i,j) = \exp\left(-\frac{d(H_i, Tr_j)}{D_i}\right) * C_{img}(i,j) \tag{6}$$

5.7. Hand-Hand Mis-Assignment Recovery

The hand-hand mis-assignment happens when both 2D hand tracking states are ACTIVE, but either of 3D tracking is lost consecutively for a certain number of frames. If hand-hand mis-assignment is detected, the cost of hand-hand re-assignment is computed as (7), where C_0 is the original configuration cost, and C_1 is the swapped hands configuration cost (see Figure 10). If $C_1 > C_0$, hand-hand assignment is swapped.

$$C_0 = \frac{C_{img}(H_{LH}, B_{LH}) + C_{img}(H_{RH}, B_{RH})}{d(B_{LH}, P_{LShoulder}) + d(B_{RH}, P_{RShoulder})}$$
$$C_1 = \frac{C_{img}(H_{LH}, B_{RH}) + C_{img}(H_{RH}, B_{LH})}{d(B_{LH}, P_{RShoulder}) + d(B_{RH}, P_{LShoulder})} \tag{7}$$



C0 = 0.063 C1=0.068

Figure 10: The computation for hand-hand assignment.

5.8. Foot Hypotheses Assignment

Foot occlusion is detected when the Euclidean distance between two feet position candidates, p_0 and p_1, is smaller than a threshold. The foot hypotheses assignment cost is computed as (8) for non-occlusion case and (9) for occlusion case. If $C_0 > C_1$, p_0 is assigned to right foot, p_1 to left foot. On the contrary, if $C_1 > C_0$, p_0 is assigned to left foot, p_1 to right foot.

$$C_0 = abs(p_0 - H_{RF}) + abs(p_1 - H_{LF})$$
$$C_1 = abs(p_0 - H_{LF}) + abs(p_1 - H_{RF}) \tag{8}$$

$$C_0 = \frac{C_{img}(p_0, H_{RF}) + C_{img}(p_1, H_{LF})}{d(p_0, P_{RPelvis}) + d(p_1, P_{LPelvis})}$$
$$C_1 = \frac{C_{img}(p_0, H_{LF}) + C_{img}(p_1, H_{RF})}{d(p_0, P_{LPelvis}) + d(p_1, P_{RPelvis})} \tag{9}$$

6. 3D Tracking

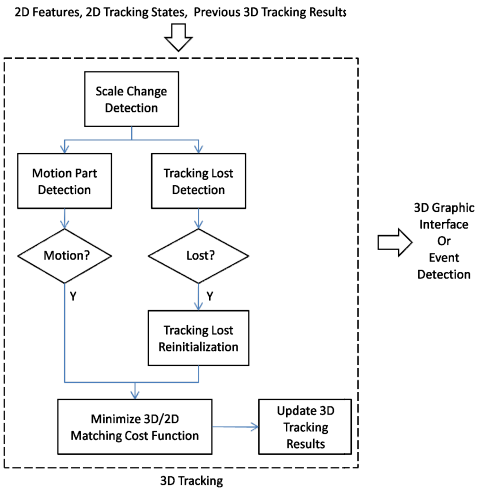


Figure 11: The flow chart of 3D tracking.

6.1. Scale Change Detection

The scale change, see Figure 11, is detected when $abs(A_{3D} - A_{Sil})$ is greater than a threshold, where A_{3D} is the area of the projection of 3D human body model with the previous pose parameters on image plane and A_{Sil} is the area of the silhouette image I_{Sil}. Once the scale change is detected, the scale of 3D body model is reset to fit the detected scale. When the scale is large enough so that the projected feet positions are out of camera range, only upper body tracking is used (see Figure 12).

Figure 12: When the scale is so large that projected feet positions are out of camera range, only upper body tracking is used.

6.2. Motion Part Detection

Motion parts that are grouped as TORSO, RLEG (Right Leg), LLEG (Left Leg), RLIMB (Right Limb) and LLIMB (Left Limb) are detected. First of all, the motion skin image $I_{MotSkin}$ and the motion edge image $I_{MotEdge}$ are computed as (10) and (11). Then, each pixel with a positive value in both $I_{MotSkin}$ and $I_{MotEdge}$ is assigned to the nearest body part. Take the body part B_{Torso} as an example, if the number of positive pixels of B_{Torso} is greater than half of the perimeter of B_{Torso}, the motion flag of the body part TORSO is marked.

$$I_{MotSkin} = I_{Motion} \cap I_{Skin} \tag{10}$$

$$I_{MotEdge} = I_{Motion} \cap I_{Edge} \cap \bar{I}_{Skin} \tag{11}$$

where $I_{Motion} = abs(currentFrame, previousFrame)$

6.3. Tracking Lost Recovery

Once a human body is detected, the mean shift model will start four trackers with individual targets, i.e., RightArm, LeftArm, RightLeg and LeftLeg, for mean shift tracking [13, 14]. One example of four mean shift targets is shown in Figure 13.

Figure 13: An example of four targets for mean shift tracking, including right arm, left arm, right leg and left leg.

For each feature point, the tracking is detected to be lost if the weighted distance error between image feature points and projected model positions is greater than a threshold. More specifically, the weighted distance error is computed as $d(p^i_{img}, p^i_{\text{model}}) * pr_i$ at the current frame or an accumulated value in several consecutive frames, where pr_i serves as a reliability factor, which is updated based on the code chip in Figure 14.

```
if ( blob_i.flag == active )
    blob_i.pr == 1.0;
else if ( blob_i.flag == lost )
    blob_i.pr == 0.5/(1 + blob_i.lostNum);
else if ( blob_i.flag == occlusion)
    blob_i.pr == 0.5;
```

Figure 14: The code chip to update the probability of blobs.

Once one of the four parts, RightArm, LeftArm, RightLeg and LeftLeg is detected as tracking lost, the corresponding mean shift tracker is triggered. Based on the color histogram in the target, the tracker will perform the mean shift tracking algorithm [13, 14] to identify the best matched area as ROI (region of interest). Then 2D feature images will be constrained in the ROI and used in 3D/2D matching cost function (see Section 6.4 for details). A tracking lost recovery example is shown in Figure 15.

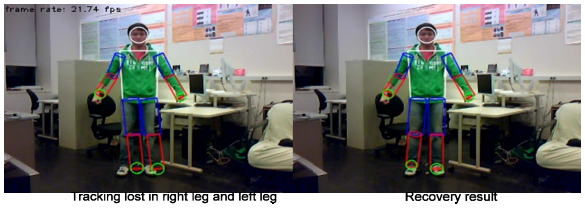


Figure 15: A tracking lost recovery example.

6.4. Minimize Cost Function

The 3D pose reconstruction is estimated for the best match configuration of limbs, torso and head by using downhill simplex algorithm [11] to minimize the cost function. The cost function $F(S_{image}, S_{model})$ between 2D feature points and 3D model positions in the state space is composed of silhouette score, edge score, motion score and feature points score, which is defined as follows:

$$F(S_{image}, S_{model}) = w_0 C_{sil} + w_1 C_{edge} + w_2 C_{motion} + w_3 C_f$$

$$\text{Silhouette Score: } C_{sil} = N_{XOR}(S^{sil}_{image}, S^{sil}_{model})$$

$$\text{Edge Score: } C_{edge} = -N_{And}(E_{image}, E_{model})$$

$$\text{Motion Score: } C_{motion} = -N_{And}(M_{image}, E_{model})$$

$$\text{Feature Points Score: } C_f = \sum_{i=1}^{5} d(f^i_{image}, f^i_{model}) * pr_i$$

where S^{sil}_{image} is the silhouette image and S^{sil}_{model} is the model projection image, E_{image} is the edge image and E_{model} is the model outline image, M_{image} is the edge motion image, f^i_{image} are human body model parameters and f^i_{model} is human body model measurement. (The weights w_0, w_1, w_2, w_3 are set to be 1, 5, 10 and 10 in our experiments.)

Based on the cost function, the downhill simplex algorithm is applied to fit the 2D feature for the 3D poses by four operations, i.e., reflection, expansion, one-dimensional contraction and multiple contractions. In order to increase the computational efficiency, the best 3D pose is searched in a hierarchical way, i.e., taking Head and Torso as a base followed by four limbs, as shown in Figure 16.

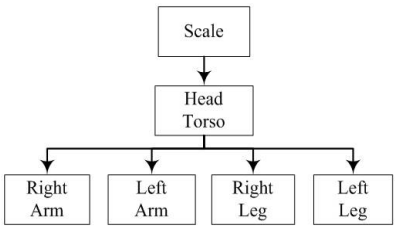


Figure 16: The 3D pose is searched hierarchically.

7. Validation

Videos with ground truth values are obtained from [15]. The disparity is calculated by a two-camera system, and it can be used to calculate the depth information. We compare 13 joints of the result of our proposed system with the ground truth values. All the 13 joints of a human body are shown in Figure 17.

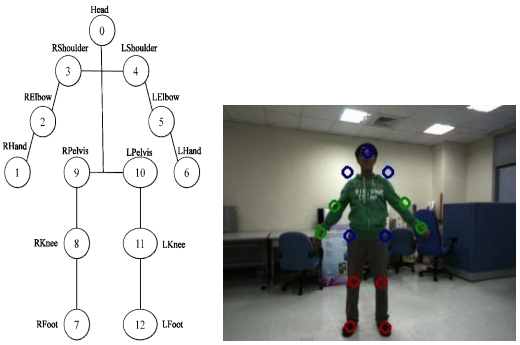


Figure 17: All 13 joints of a human body.

In Figure 18, the mean square error (MSE) of 2D coordinates of joints is presented. For example, for Head joint, the average distance between a head point in our method and one in ground truth is 1.42 pixels. Moreover, the curve shows depth values along the frame number. We can see in Figure 18 that the trend of the depth curve of our method and the depth curve in ground truth is quite consistent. The snapshots of the test video are shown in Figure 19.

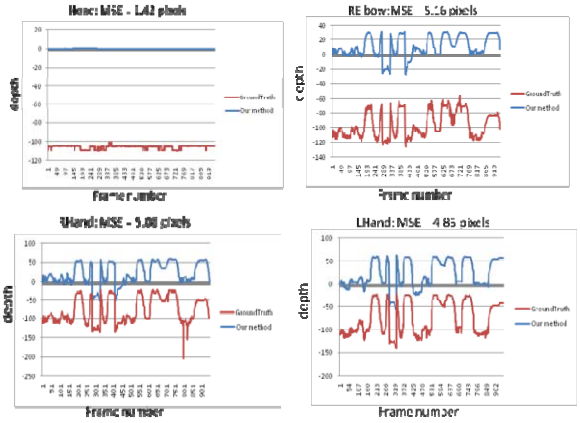


Figure 18: MSE and depth for Head, REblow, RHand, LHand.

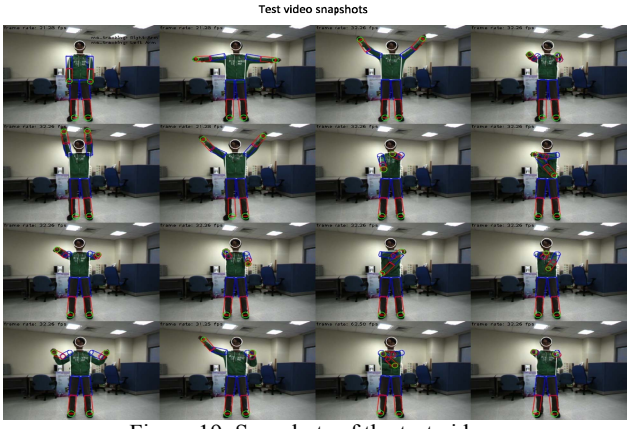
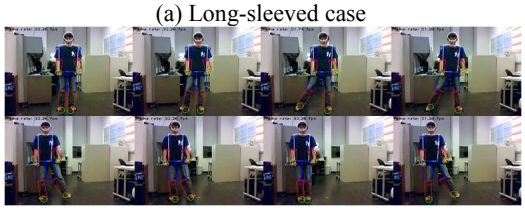


Figure 19: Snapshots of the test video.

8. Experimental Results

The system is implemented in C++ and run on a laptop (CPU Intel Core 2 Duo T8100 2.1 GHz, RAM 3 GB, Windows 7). The image resolution is 320x240 pixels and the real-time frame processing rate is 26~32 fps. Figure 20 shows the long-sleeved and short-sleeved cases. Besides, two applications are shown as follows.

(a) Long-sleeved case



(b) Short-sleeved case

Figure 20: 3D tracking for long-sleeved and short-sleeved cases.

8.1. Event Detection

The 3D tracking result is sent to event detection module for object tracking and detecting various behaviors. As shown in Figure 21, when the person lifts a purple bag, the event detection module is triggered and tracks the purple bag.

Figure 21: An event of lifting a purple bag.

8.2. Video Gaming

A simple video game is implemented in OpenGL as shown in Figure 22. The avatar, which is real-time generated by the derived 3D poses result of the proposed system, is controlled to hit balls so as to get scores or to avoid attacks from flying balls.

Figure 22: A 3D video game console.

9. Conclusion and Future Work

We have proposed a real-time system for tracking 3D human poses from videos by decomposing the human body model into different parts and integrating different kind of cues. The difficult cases such as tracking lost and occlusion are effectively dealt with in the proposed system. In the future, the system can be further improved to recognize more challenging behaviors from different camera viewing perspectives and to solve the problem of occlusion between the objects and the human body.

References

[1] T.B. Moeslund, A. Hilton, and V. Kruger, "A survey of advances in vision-based human motion capture and analysis", Computer Vision and Image Understanding, pp. 90-126, 104, 2006.

[2] M.W. Lee and I. Cohen, "A model-based approach for estimating human 3D poses in static images", PAMI, vol. 28, pp. 905-916, 2006.

[3] C.Sminchisescu and B.Triggs, "Kinematic jump processes for monocular 3D human tracking", CVPR'03, pp. 69-76.

[4] R. Urtasun, D.J. Fleet, and P. Fua, "Monocular 3D tracking of the golf swing", CVPR, vol.2, pp. 932-938, 2005.

[5] A. Agarwal and B. Triggs, "Recovering 3d human pose from monocular images", PAMI, vol.28, pp.44–58, 2006.

[6] G.Shakhnarovich, P.Viola, and T.Darrell, "Fast pose estimation with parameter-sensitive hashing", ICCV, vol. 2, pp.750–757, 2003.

[7] R. Okada, B. Stenger, "A single camera motion capture system for human-computer interaction", Trans. IEICE, vol. E91-D, No.7, pages 1855-1862, July 2008.

[8] P. Noriega, O Bernier, "Multicues 3D monocular upper body tracking using constrained belief propagation", BMVC, UK, Warwick, 2007.

[9] R. Holt, A. Netravali, T. Huang and R. Qian, "Determining articulated motion from perspective views: a decomposition approach", in Proc. IEEE Workshop on Motion of Non--Rigid and Articulated Objects, pp. 126-137, Texas, 1994.

[10] Saleh A. Al-Shehri, "A simple and novel method for skin detection and face locating and tracking", Lecture Notes in Computer Science, vol. 3101, pp. 1-8, Berlin, 2004.

[11] W. Press, S. Teykolsky, W. Vetterling, and B. Flannery. "Numerical recipes in C++: the art of scientific computing", Pearson Education, 1992.

[12] LiangJia Zhu, Jenq-Neng Hwang, Chih-Chang Chen, Ming-Hui Lin, Chen-Lan Yen, "Real-time 3D pose reconstruction of human body from monocular video sequences", Taipei, ISCAS, pp. 717-721, 2009.

[13] Dorin Comaniciu, Visvanathan Ramesh, Peter Meer, "Real-time tracking of non-rigid objects using mean shift", CVPR, vol. 2, pp.2142, 2000.

[14] Dorin Comaniciu, Peter Meer, "Mean Shift: a robust approach toward feature space analysis", PAMI, 2002.

[15] J.F. Wang, P.L. Chen, C.P. Liao, Y.Y. Tsai, J. Huang, K.S. Wang, "Real time depth sensing with automatic determined depth range for human computer interactive applications", IPCV, Las Vegas, NV, 12-15 July, 2010.