國 立 交 通 大 學

電子工程學系 電子研究所碩士班

碩 士 論 文

多攝影機運動捕捉系統中基於人體模型之姿態估測研究

Model-Based Pose Estimation for Multi-Camera Motion Capture System

研 究 生：蕭晴駿

指導教授：王聖智 博士

中 華 民 國 九 十 七 年 七 月

多攝影機運動捕捉系統中基於人體模型之姿態估測研究

Model-Based Pose Estimation for Multi-Camera Motion Capture System

研　究　生：蕭晴駿　　　　　Student：Ching-Chun Hsiao

指導教授：王聖智博士　　　　Advisor：Dr. Sheng-Jyh Wang

國　立　交　通　大　學

電子工程學系　電子研究所碩士班

碩　士　論　文

A Thesis
Submitted to Department of Electronics Engineering & Institute of Electronics
College of Electrical and Computer Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of Master
in
Electronics Engineering
July 2008
Hsinchu, Taiwan, Republic of China

中華民國九十七年七月

多攝影機運動捕捉系統中基於人體模型之姿態估測研究

研究生：蕭晴駿　　　指導教授：王聖智 博士

國立交通大學

電子工程學系 電子研究所碩士班

摘要

不論在安全監控、人機互動、電腦動畫或甚至醫學應用上，人物動作的擷取與分析是個相當重要的議題。在本論文中，我們提出一個在多攝影機環境下，利用人體模型估測目標人物的姿勢與行為。我們使用流形嵌入技術中的拉普拉斯特徵映射，將三維人形的幾何形狀忠實地轉移到另一個容易切割分析的高維度空間，正確地切割出三維人形的各個部位並且找出三維人形的骨骼架構，以利後續行為分析的動作。當擷取出三維人形的骨骼架構後，我們利用粒子群體最佳化在高維度空間中有效地找出最佳姿態估測結果。我們的系統由影像的擷取至姿態的估測完全自動化，並且不需要在人體上貼附感應物，即可結合肢體的運動限制和時間軸上的動作流暢限制，估測多種動作。

Model-Based Pose Estimation for Multi-Camera Motion Capture System

Student : Ching-Chun Hsiao Advisor : Dr. Sheng-Jyh Wang

Department of Electronics Engineering, Institute of Electronics

National Chiao Tung University

Abstract

In this thesis, we propose a 3-D human body pose estimation method using multiple cameras. The reconstructed human body is transformed to a high dimensional space using our modified Laplacian Eigenmap. In this eigenspace, the body parts can be segmented more efficiently and easily. Then, the 3-D skeletons of the human body are extracted to obtain the kinematic information. Finally, pose estimation is performed by fitting a prior 3-D model to the extracted skeleton via particle swarm optimization (PSO). PSO is suitable for the optimization problem with nonlinear cost functions and doesn't need too much computational cost. Furthermore, with our proposed human model, the motion constraints can be easily combined with the optimization process. Temporal consistency of the pose estimation results is also achieved by adding temporal constraints over PSO. Our method can deal with various kinds of motion and has robust pose estimation results.

誌謝

很感謝王聖智老師從大三到碩二這段期間，給予我們耐心的指導，從專題到論文研究，不論是訂定題目、研究相關論文和問題的探討與解決，老師總是耐心開導與指引，從老師身上，學到了許多研究精神和待人接物的道理，也是我們心目中的模範。也感謝實驗室的學長姐、同學和學弟們，不論是什麼問題和困難，大家總是能一起分擔、互相幫忙：特別感謝敬群學長，在我研究起步與遇到瓶頸時，給予很多建議和幫忙，學長也是我們實驗室的活力來源;也感謝奕安、瑞男、庭瑋和文中，謝謝你們忙碌於實驗室的大小事務，特別是我有這麼快速的電腦可以拿來實驗模擬，真的很謝謝你們的幫忙！同時也謝謝我的家人，你們是我前進的動力與支持力，也是我永遠的依偎和寶貝。還要感謝我親愛的好姊妹們，匀珮、尊宇和婉清，謝謝你們的陪伴和關懷，讓我感受到無敵大溫暖。最後要感謝博凱，因為你一路走來的支持和幫助，和我一起品嚐研究生活的酸甜苦辣，讓我覺得既幸福又珍貴，更是我力量的泉源。很謝謝 Vision Lab 帶給我的一切，不論是研究精神的養成和友情的溫暖，畢業的學長姐們也很熱心的協助我找工作，能加入這個實驗室真的太幸福了！真的很感謝大家！

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Human motion capture is to estimate the configuration of human body parts using inputs from one or more video streams. It has gained in popularity among surveillance, human-machine interaction, computer animation and medical applications. Typically, a motion capture system can be classified into marker-based or markerless. Marker-based motion capture systems require the human body be equipped with markers. However, the use of markers is very cumbersome and may restrict the freedom in observation. Furthermore, this approach is expensive and it is hard to align kinematic motion to marker data. Recently, there have been two main approaches, monocular approach and multi-camera approach, for markerless motion capture system. The advantages of monocular approach are simple hardware setup and lower cost. However, as shown in *Figure 1-1*, there exist depth ambiguities when only one silhouette is available. Hence, multi-camera approaches, which reconstruct 3-D human bodies from a set of silhouettes, are preferred to offer more information for motion capture systems. With more view angles we can alleviate the occlusion problem and make the motion capture system more robust.

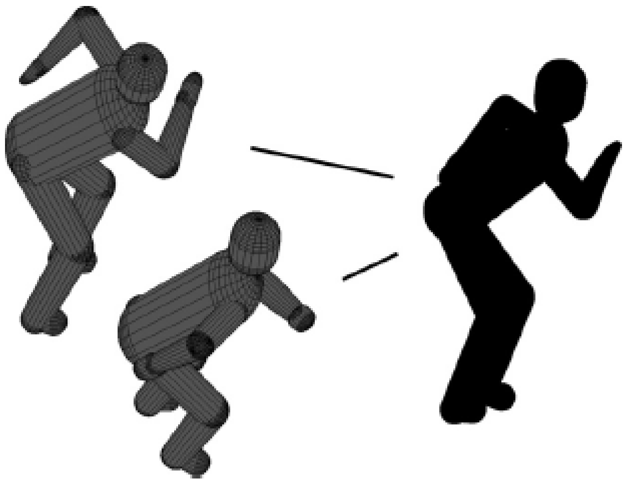


Figure 1-1 Depth ambiguity problems exist in monocular approaches[20]

In this thesis, we propose a markerless motion capture system equipped with multiple cameras. First, a 3-D human body represented by voxels is reconstructed from multiple video streams. A modified Laplacian Eigenmap algorithm is used to transform the 3-D voxel data into a high dimensional space. With this manifold embedding method, different body parts are mapped into discriminative branches and can be easily segmented. Unlike other approaches, this approach relieves the dependence on human model and the training database. After the segmentation of body parts, skeletons are extracted to describe the kinematic motion of the human body. Human shapes are usually deformed while skeletons can encode most of the motion information. As the skeletons are extracted from the 3-D human bodies, we use the particle swarm optimization (PSO) technique to deal with the pose estimation problem. The experimental results show that our system can handle various kinds of poses and can ensure temporal consistency and motion constraints.

This thesis is organized as follows. In Chapter2, we introduce the background of motion capture systems. In Chapter3, we discuss the properties of the Laplacian Eigenmap and the proposed pose estimation approach. Experimental results are shown in Chapter4. Finally, we will make a brief conclusion in Chapter5.

Chapter 2. Backgrounds

In this chapter, we will introduce a few motion capture approaches developed in recent years. Firstly, a brief introduction to motion capture and its functional taxonomy are discussed in Chapter 2.1. Since we'll focus on markerless motion capture systems equipped with multiple cameras, related algorithms are also mentioned in Section 2.2.

2.1 Motion Capture Systems

Some popular motion capture algorithms are to be reviewed in this section. Since systems with markers are costly and ineffective, we mainly discuss markerless motion capture systems, which have drawn much attention in recent years. According to the survey in [26], we can decompose markerless approaches into several submodules based on the following functional taxonomy:

- Initialization: the initialization of motion capture systems tends to acquire some prior knowledge for pose analysis. The prior knowledge may include kinematic structure, 3-D human shape, and so on.
- Tracking: this process typically includes foreground detection and continuous target tracking in the video.
- Pose estimation: given some prior knowledge obtained in the initialization step, this process tries to estimate the posture of a specific person.
- Recognition: this process identifies the activities or behaviors of the target person, such as running, walking, and so on.

In the proposed motion capture system, we mainly focus on the initialization module and the pose estimation module. The module of initialization aims to obtain reliable prior knowledge for pose estimation and recognition. Due to error propagation, incorrect prior knowledge may lead to wrong pose estimation. In the following discussion, we will focus on a few algorithms which are related to initialization and pose estimation.

In general, markerless motion capture systems can be classified into model based systems and model free systems, depending on whether a prior human model is used in the system. In the following sections, we will introduce several popular motion capture systems in each of these two categories.

2.1.1 Model-Based Systems

Model based systems use an explicit model to capture the motion of a specific person. Depending on whether the system is monocular or multi-view, the model can be either 2-D or 3-D. In the following subsections, we will briefly introduce model-based systems based on these two system types.

2.1.1.1 Monocular Motion Capture

Monocular motion capture systems only use one camera and deal with the problems in the 2-D image planes. For 2-D model-based motion capture systems, contour human models, stick models, and cardboard models are commonly used.

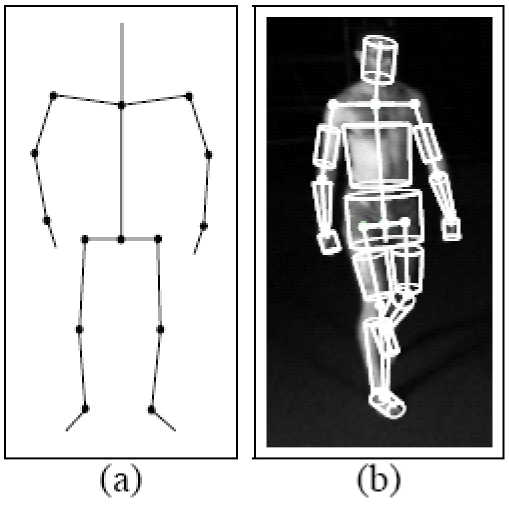


Figure 2-1 2-D human model examples (a) The 2-D skeleton human model consists of 17 kinematic chains (b) flesh the model in (a) by conical sections[15]

Deutscher [15] modified the particle filtering algorithm to estimate the posture in the 2-D image plane. Given foreground silhouettes, 2-D human models as shown in *Figure 2-1* are used to fit the detected human bodies. In this approach, human models are composed of 17 chains connected by joints. Each joint has its degree of freedom (DOF) and the whole model adds up to 29 DOF. Pose estimation is performed by fitting the human model to the foreground silhouettes. The fitting result, as illustrated in *Figure 2-1* (b), describes the posture of the person. However, the searching space of this approach is too large due to the high DOF. Even though the particle filter has been a useful tool in data fitting, it may get trapped in a local maximum and does require lots of computation in a search space of high dimensionality. In [15], the anneal particle filtering is adopted to effectively find the global maximum. However, depth ambiguities still exist and the processing speed is far too slow (about 1 hour for a 5-second footage).

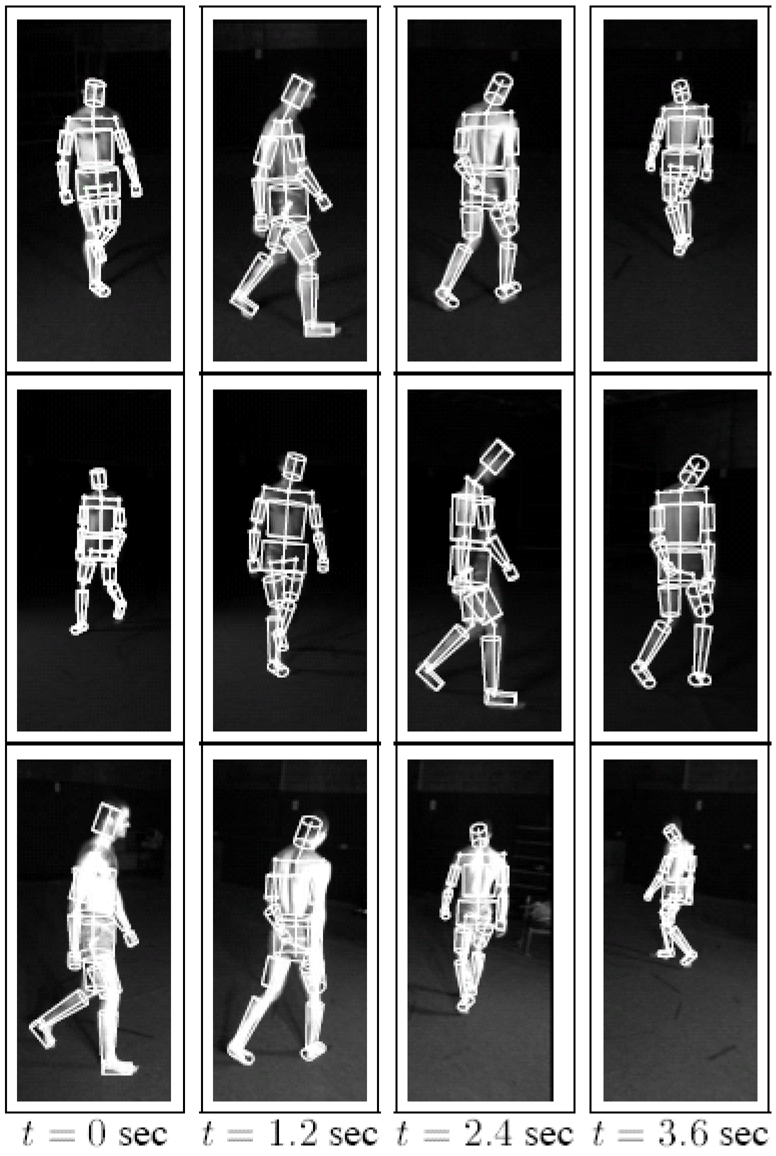


Figure 2-2 The pose estimation results in [15]

2.1.1.2 Multiple View Motion Capture

Estimating posture from 2-D images only is rather difficult. This is because the self occlusion problem and the depth ambiguity problem cannot be properly handled. In order to obtain more robust motion capture results, 3-D data are more preferable. In [5], the reconstruction of a "visual hull" based on images from multiple cameras is introduced. A visual hull is defined as the 3-D shape formed by the intersection of visual cones projected from the 2-D silhouettes, as illustrated in Figure 2-3 [5]. The visual hull of an object can be thought to be a close approximation of the object based on the observations from different viewpoints.

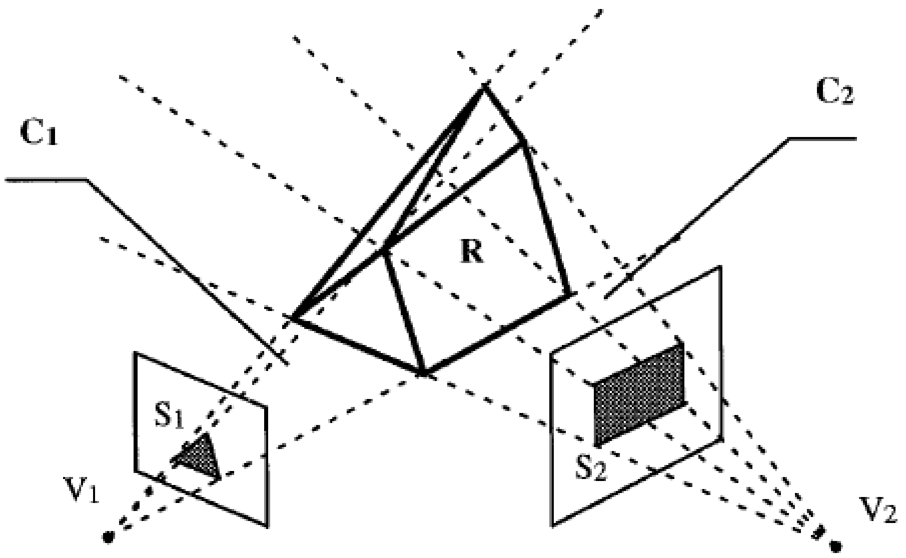


Figure 2-3 The visual hull is constructed by volume intersection [5]

The visual hull can be represented by voxels and several algorithms have been developed to construct the visual hull based on a set of silhouette. In practice, we don't actually compute the visual cones and their intersection. It's computationally costly and difficult to find their intersection parts. Instead, when implementing the visual hull reconstruction algorithm, voxels are back projected to the image planes to check whether their projections fall into the region of the foreground silhouettes. Then, the silhouettes on multiple cameras vote to decide whether a voxel belongs to the visual hull. Szeliski [24] proposed an octree representation of the voxel space where the resolution of voxels is variable according to their projections on the image planes. It can reconstruct a visual hull from coarse resolution to fine resolution. However, this approach is sensitive to the hypothesis of visibility. Kehl [22] introduced a fast visual hull construction method. A voxel look-up table is built to record the projection position in each image plane for all the voxels. To speed up the process while maintaining the high-resolution voxels, the projection is approximated by a cross patch. As shown in *Figure 2-4*, Voxels 7, 12 and 15 project to the same position in the image plane. Hence, they are recorded in the look-up table of that pixel. Once that pixel is labeled as "foreground", these voxels may belong to the visual hull.

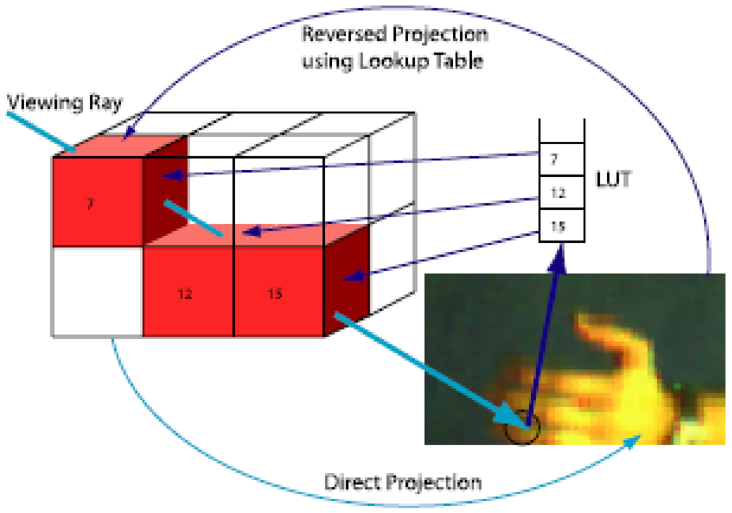


Figure 2-4 A look up table is kept to record the voxels which are projected into the same pixel [21]

To model the 3-D shape of the visual hull, Mikic [13] adopted a twist framework that has been used to model the kinematic chains for robots. Sixteen rotation axes and five kinematic chains of the body joints are formulated using twists and product of exponentials. Based on a torso-centered coordinate system, the rotation and shift of the body parts can be easily manipulated. Given voxel data reconstructed from 6 calibrated and synchronized cameras, pose estimation is performed by first doing template fitting and then using Bayesian network for refinement. However, the initialization of this approach is based on template fitting, which cannot deal with self occlusion. Moreover, this approach requires that the target person be dressed in tight clothes.

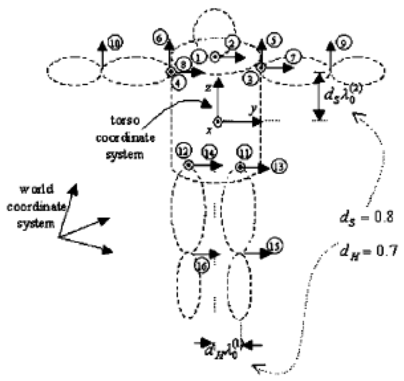


Figure 2-5 3-D articulated human body model used in [13]

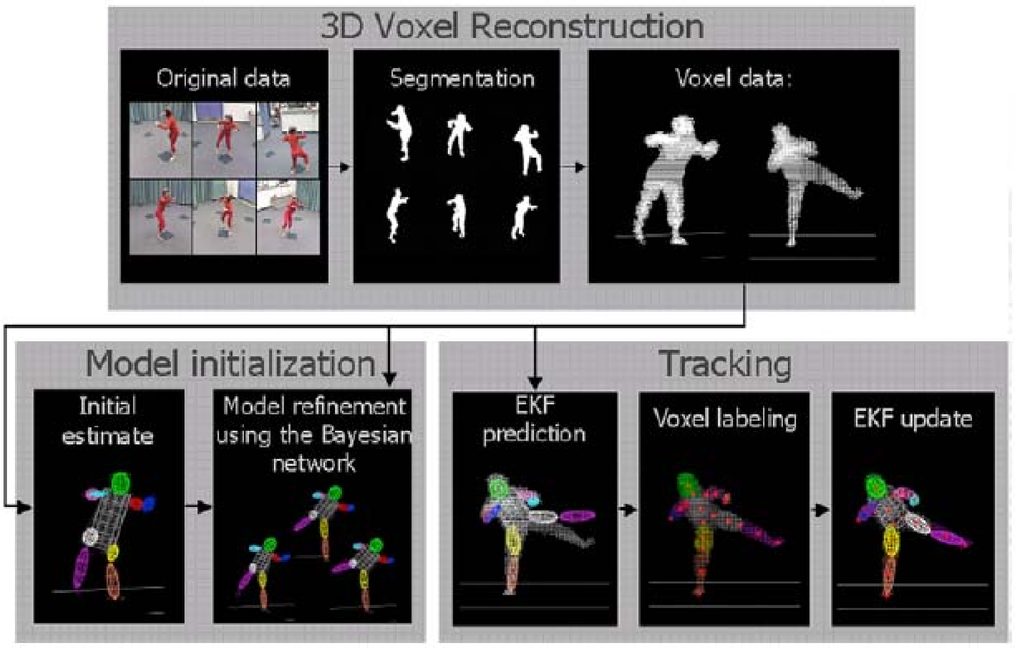


Figure 2-6 The flow chart of Mikic's approach [13]

Kehl [22] proposed a stochastic meta descent (SMD) optimization method to fit the body model to the visual hull. SMD is the generalization of gradient descent. It adapts its local step size to offer more rapid convergence and uses stochastic sampling to avoid being trapped in a local minimum.

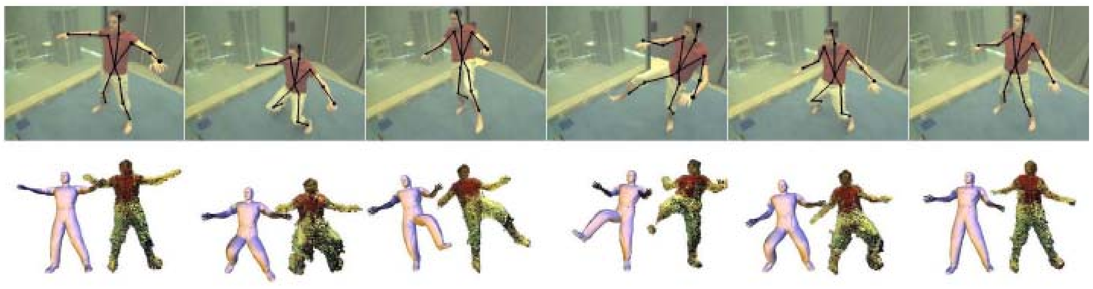


Figure 2-7 Pose estimation and visual hull texturing results in [22]

Instead of using shape models, Menier [7] adapted skeleton models to fit medial axis points extracted from visual hulls. This approach reduces the dependency on the dimension of human body. These 3-D medial axis points represent the observed skeleton data. A generic skeleton model is then fitted with the observed skeleton data based on a maximum a posteriori (MAP) estimation. The pose estimation of the first frame is based on the fitting process, while non parametric belief propagation is used to predict the pose in the following frames.

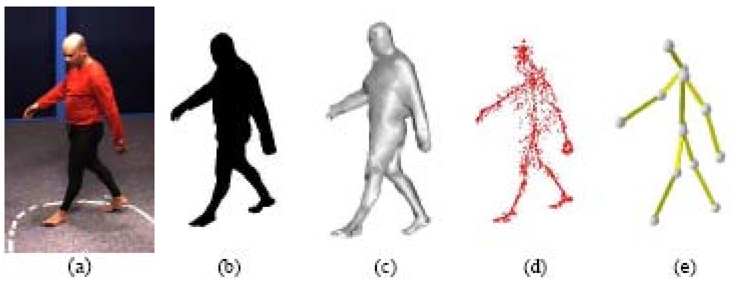
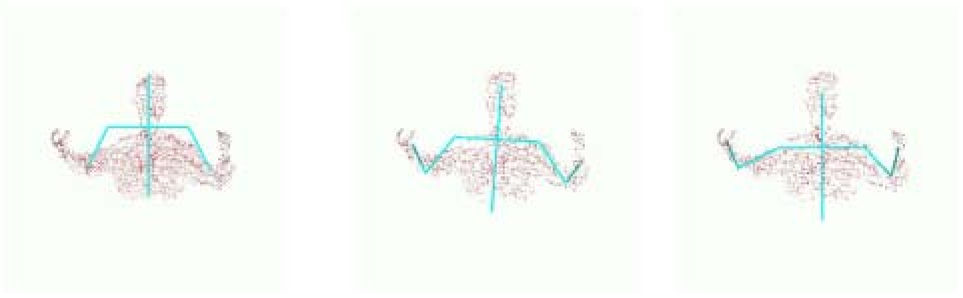


Figure 2-8 The pose estimation scheme of [7] : (a) Input images (b) detected foreground (c) reconstructed visual hull (d) extracted medial axis points (e) fitted skeleton model

Due to the high dimensionality of the search space and the complexity of the fitness evaluation function, some researchers have adopted particle swarm optimization (PSO) method to perform pose estimation. PSO [16] is an optimization technique that simulates the social behaviors of animals. Given a fitness function and a communication network, particles gradually move to the best position according to the self experience and the information obtained from their neighbors. Robertson [8] applied PSO to perform skeleton model fitting in a conference room environment, where the pose estimation is required only for the upper body. PSO is chosen for its ability to deal with nonlinear and non-convex optimization problems. Hierarchical and parallel PSO fitting is proved to be robust and computationally inexpensive.



(a) (b) (c)

Figure 2-9 Hierarchical fitting process: (a) root position and orientation (b) shoulders and trunk (c) all are fitted [8]

Model based motion capture systems use a prior model to reduce the search space and hence have less complexity. Moreover, they usually assume tight fitting clothes and the initialization of the human models has great influence on the accuracy of pose estimation. Since template fitting-based initialization cannot well handle self occlusion, embedding-based initialization has drawn much attention recently. In Section 2.2, we will introduce a few popular manifold embedding methods and explain how they are applied to initialize the motion capture systems. These embedding-based methods may produce robust human body labeling results and make the initialization process less sensitive to human poses and human models. In this thesis, we will also propose a pose estimation method that adopts embedding-based initialization.

2.1.2 Model-Free Systems

Model-free approaches don't have an explicit model. For most of this kind of motion capture systems, pose estimation is made by comparing the observed data with the database or by a pre-learnt mapping relation between the input images from the training data and their pose estimation results.

Agarwal [3] proposed a learning-based pose estimation for monocular motion capture systems. A shape descriptor is extracted from silhouettes to overcome the foreground segmentation errors. A database is used to train the relevance vector machine (RVM) and decisions are made after machine learning. Without the need of human models and initialization process, a 3-D pose can be estimated from a single silhouette.

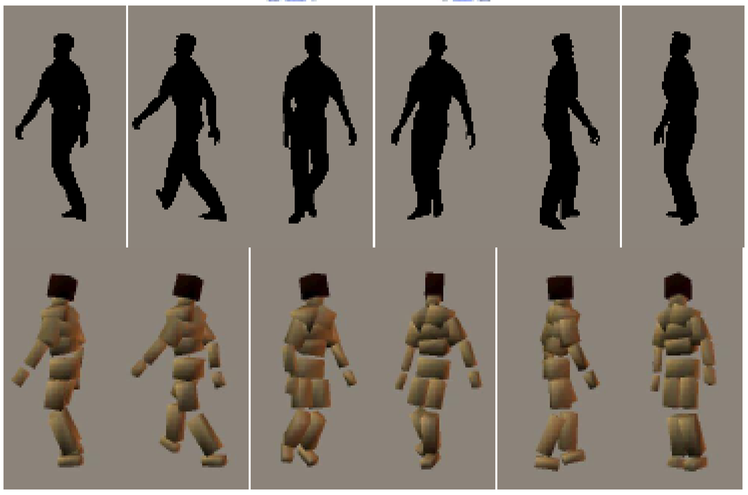


Figure 2-10 Pose estimation results from single silhouette [3]

Elgammal [4] applied the manifold learning methods to pose estimation. For the learning step, the mapping between silhouettes and 3-D poses are constructed using locally linear embedding (LLE), which is a nonlinear manifold embedding method. Then, pose estimation is done based on the manifold embedding results. That is, given detected silhouettes, its 3-D pose estimation is obtained from the mapping relation learned by LLE.

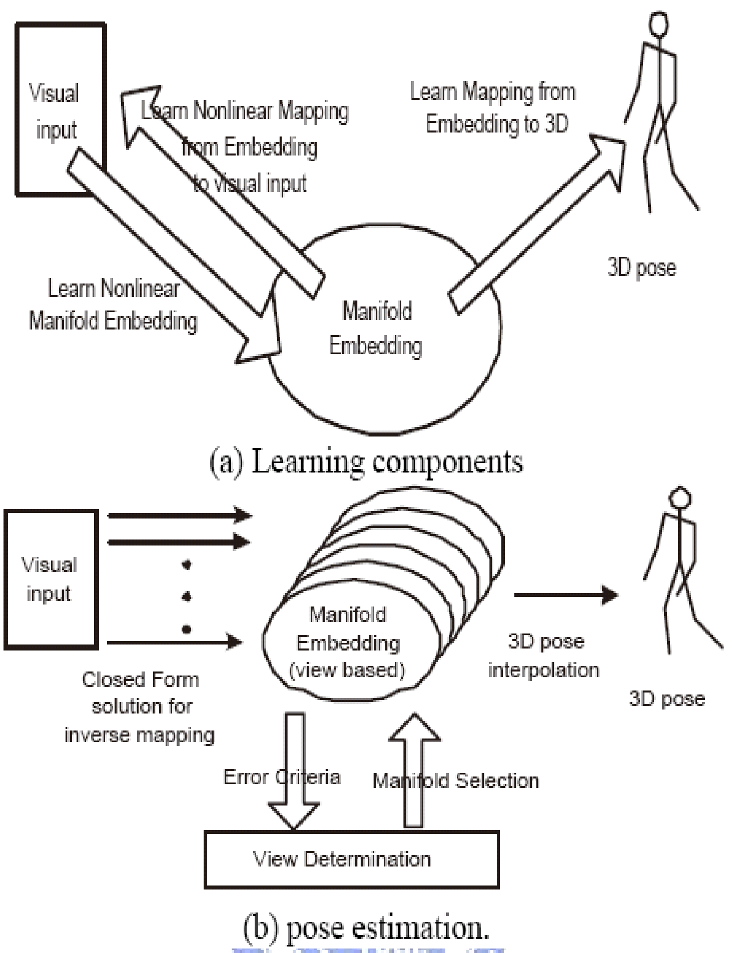

Visual input
Learn Nonlinear Mapping from Embedding to visual input
Learn Mapping from Embedding to 3D
3D pose
Learn Nonlinear Manifold Embedding
Manifold Embedding
(a) Learning components
Visual input
Closed Form solution for inverse mapping
Manifold Embedding (view based)
3D pose interpolation
3D pose
Error Criteria
Manifold Selection
View Determination
(b) pose estimation.

Figure 2-11 The flow chart of Elgammal's method. [4]

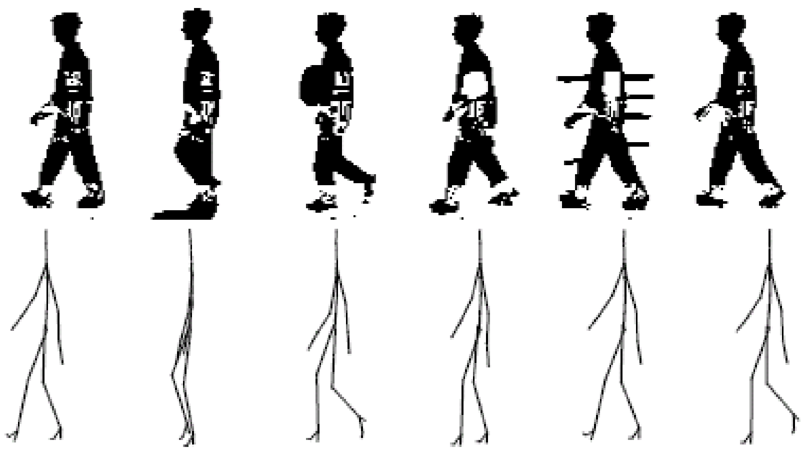
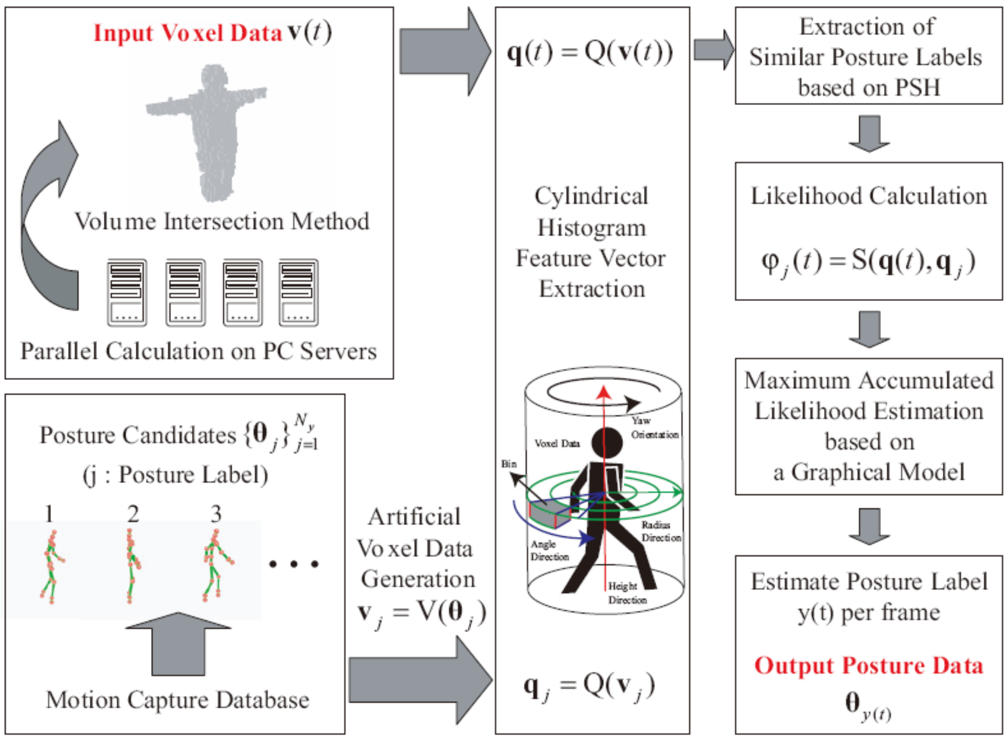


Figure 2-12 3-D pose estimation results of Elgammal's method. [4]

Sagawa [27] proposed an example-based approach to perform pose estimation. For each frame, the feature vector is first extracted from the reconstructed 3-D voxel data and then compared with the database. The evaluation process is further improved by a graphical model of motion that ensures the temporal consistency.



Input Voxel Data $\mathbf{v}(t)$
Volume Intersection Method
Parallel Calculation on PC Servers
$\mathbf{q}(t) = Q(\mathbf{v}(t))$
Cylindrical Histogram Feature Vector Extraction
Yaw Orientation
Voxel Data
Bin
Angle Direction
Radius Direction
Height Direction
$\mathbf{q}_j = Q(\mathbf{v}_j)$
Extraction of Similar Posture Labels based on PSH
Likelihood Calculation
$\varphi_j(t) = S(\mathbf{q}(t), \mathbf{q}_j)$
Maximum Accumulated Likelihood Estimation based on a Graphical Model
Estimate Posture Label y(t) per frame
Output Posture Data
$\boldsymbol{\theta}_{y(t)}$
Posture Candidates $\{\boldsymbol{\theta}_j\}_{j=1}^{N_y}$
(j : Posture Label)
1
2
3
Motion Capture Database
Artificial Voxel Data Generation
$\mathbf{v}_j = V(\boldsymbol{\theta}_j)$

Figure 2-13 The flow chart of [27]

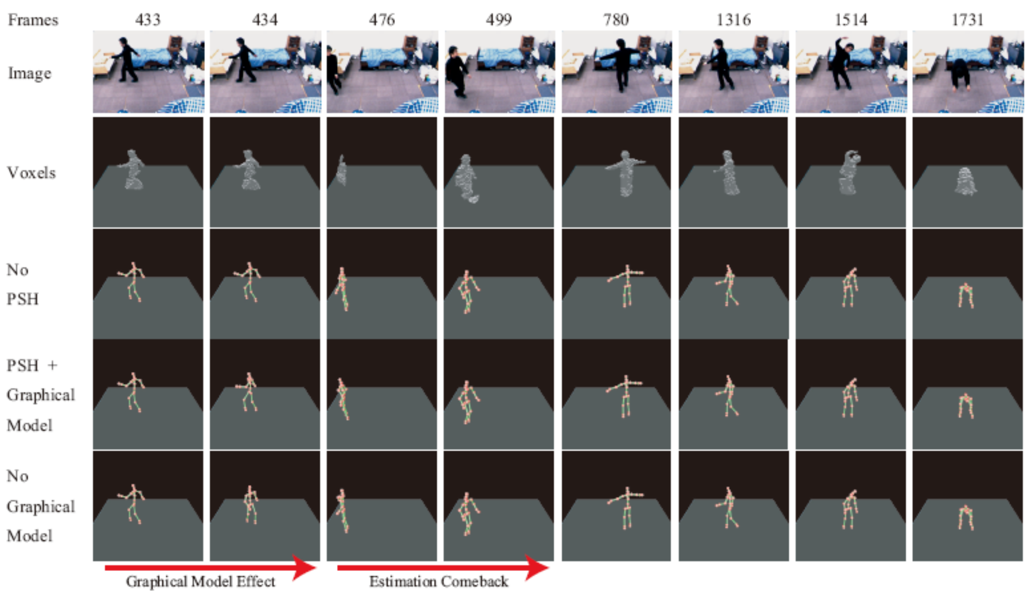


Figure 2-14 Pose estimation results of [27]

Model-free systems have the advantages of lower computational cost and can avoid large search space and nonlinear optimization. However, the performance of a model-free motion capture system greatly depends on the database. If the database is not diverse enough, the decision may be biased and may result in inaccurate pose estimation.

2.2 Manifold Learning

As mentioned earlier, model based motion capture systems have the advantages of complexity reduction and robustness. However, a good initialization is required to ensure that the system commences with a good body parts labeling and initial guess. Template fitting cannot deal with self occlusion while direct fitting human models to 3-D data results in high dimensionality of the search space. More reliable initialization approaches based on manifold learning have been proposed recently. In manifold learning, the intrinsic geometric properties are preserved after mapping the 3-D data to another space. Furthermore, this method makes the body parts labeling much easier and reliable. In the following sections, we will firstly introduce some manifold learning methods and then discuss how they are applied to motion capture.

2.2.1 Manifold Embedding Methods

Manifolds are defined as a topological space which is locally Euclidean [1]. Manifold embedding is a topic about how to find a transformed space for the manifold that preserves the connectivity and algebraic properties. Usually, manifolds are the data lying in the high dimensional space and the transformation helps reduce the dimensionality. Therefore, manifold embedding is a useful tool for dimension reduction. There have been many manifold embedding methods which can be classified into linear approaches and nonlinear ones. Linear manifold embedding methods such as principal component analysis (PCA) and multidimensional scaling (MDS) assume the data is a linear function of the features. This assumption isn't general and cannot deal with nonlinear cases. More generalized nonlinear approaches have been proposed since 2000. ISOMAP, locally linear embedding (LLE) and Laplacian Eigenmap (LE) are graph-based methods among the popular manifold embedding methods.

In these graph-based methods, graph models are used to approximate the structure of the manifold. A graph-based manifold embedding method typically have three basic steps:

1. Find the k nearest neighbors for each node to construct the graph.
2. Local properties in the neighborhood of each node are estimated.
3. Embed the graph globally to another space that preserves the local properties estimated in Step2.

ISOMAP [14] is abbreviated from isometric feature mapping. It tries to preserve the geodesic distances. Geodesic distance is a better measure than Euclidean distance for manifolds. However, the Euclidean distance is much easier to calculate. Hence, the geodesic distance between any two points is approximated by a chain of short paths, whose distance is calculated using Euclidean distance.

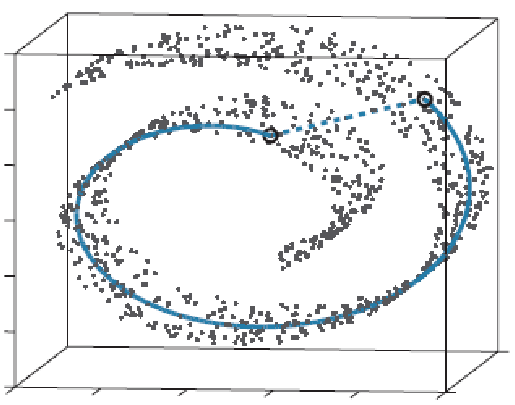


Figure 2-15 The geodesic distance between two points on the manifold "Swiss roll" is represented by the solid lines. Their Euclidean distance is the dotted line. It is obvious that the geodesic distance is a more reasonable measure to describe the relation between these two marked points [14].

Locally linear embedding (LLE) [25] assumes that manifolds are linear when viewed locally. For each node, the k nearest neighbors are selected. Then the LLE method reconstructs each node based on the linear combination of their k neighbors. The geometric properties are preserved by choosing the linear weights of these neighbors to minimize the reconstruction error.

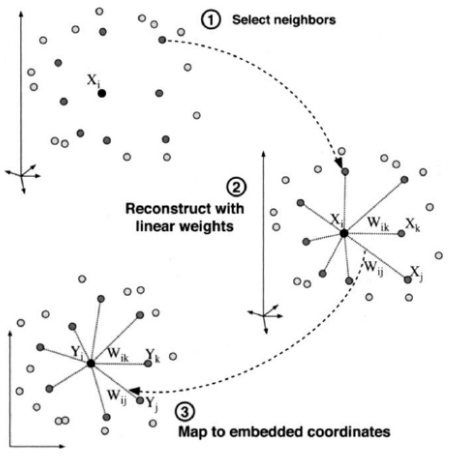
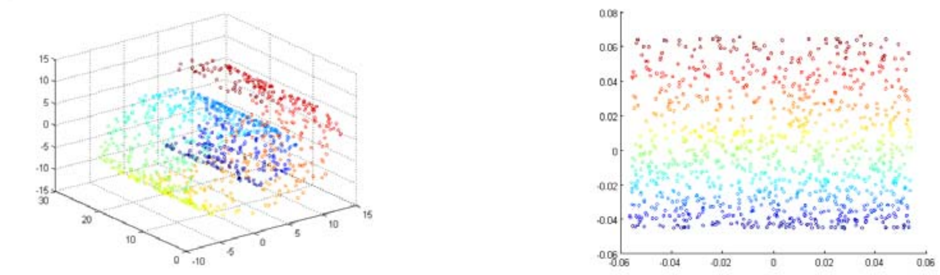
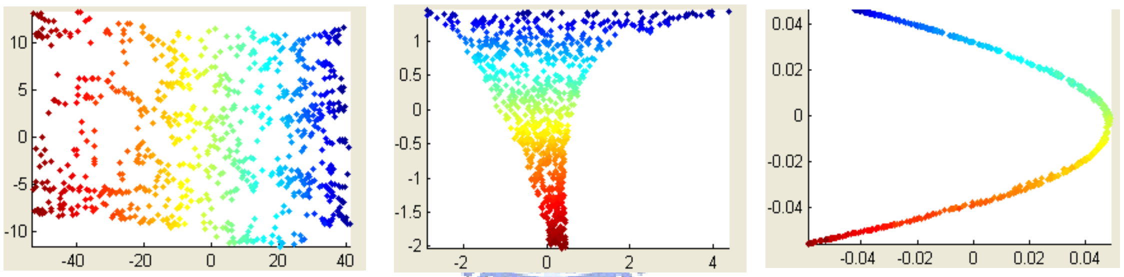


Figure 2-16 Illustration of the three steps of LLE [25].

On the other hand, Laplacian Eigenmap (LE) [18] uses graphs and Laplacian of the graphs to approximate the manifold structure and the Laplacian Beltrami operator, respectively. Laplacian Beltrami operator is the extension of the second-order differential operator, Laplacian, and is defined on manifolds as well as on surfaces. It can be shown that the eigenfunctions of the Laplacian Beltrami operator can well preserves the intrinsic geometric properties. Similarly, the eigenvectors of the Laplacian of the graph provide the appropriate embedding as well. Instead of preserving the geodesic distances, LE attempts to make the adjacent nodes in the normal space close to each other in the embedding space. By this approach, the intrinsic geometric properties can be easily preserved.



(a) Swiss roll (b) Intrinsic structure of the swiss roll



(c) ISOMAP (d) LLE (e) LE

Figure 2-17 Three manifold embedding results for Swiss roll [2]

2.2.2 Embedding Based Initialization

Since manifold embedding methods can preserve geometric properties, some embedding based systems have recently been proposed for motion capture. These manifold embedding based methods are independent of human models and are hence less restricted. Moreover, the embedding spaces possess some useful properties that make the initialization process simpler and more robust.

Chu [9] applied ISOMAP to motion capture systems. The fact that a prior human model is no longer needed makes the initialization process less restricted and more reliable. For each frame, skeleton points are extracted using ISOMAP. Finally, a kinematic model is fitted to each frame to perform pose estimation. ISOMAP can transform the 3-D volume data into a pose-independent space, which preserves the intrinsic geometric structure. Next, the nonlinear spherical shells (NSS) method is used to perform partitioning and clustering in the embedding space to obtain tree-structured principal curves. Finally, the curves are projected back to the normal space and the skeleton point features are extracted. Based on the sequence of the skeleton curves, a normalized kinematic model can be constructed. Pose estimation is done by applying this kinematic model to all frames.

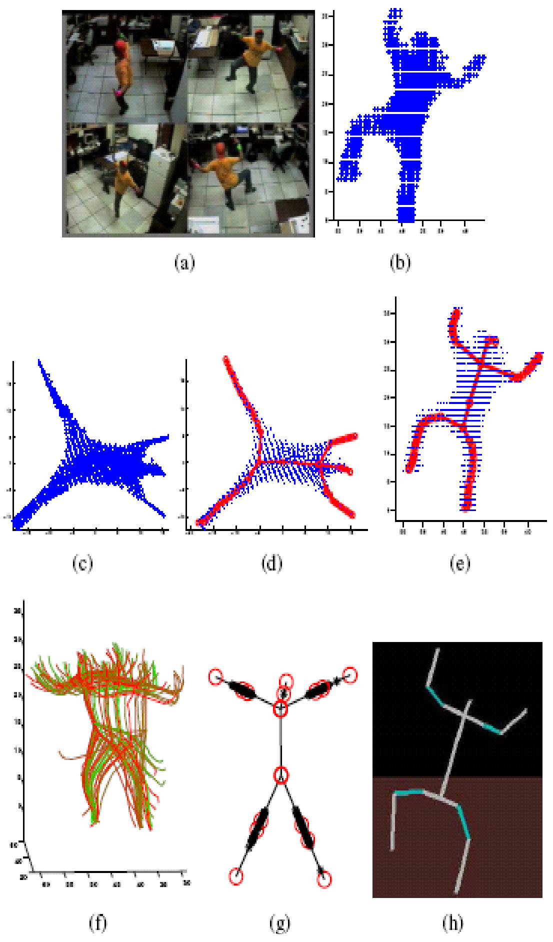


Figure 2-18 The processing flow of [9] : (a) the input sequence from multiple cameras (b) reconstructed volume data (c) transformed volume data in the 3-D embedding space (d) use NSS to obtain skeleton point features (e) project skeleton points back into the normal space to obtain the skeleton curves (f) a sequence of skeleton curves (g) a normalized kinematic model is estimated from (f) (h) use the normalized kinematic model to perform pose estimation

LLE is also applied to human motion capture. Cuzzolin [10] proposed a robust body parts labeling method along the temporal axis based on LLE. Temporal information is added to help segment the body parts in a consistent way. Unlike ISOMAP, LLE can enhance the separation of different body parts, which make the body parts segmentation much easier. Moreover, ISOMAP is computationally expensive. For these reasons, 3-D voxel data is transformed to a 3-D embedding space via LLE. After transformation, the segmentation is done by k-wise clustering. The clustering seeds are propagated through time. The clusters also merge or split according to the topology changes to ensure temporal consistency.

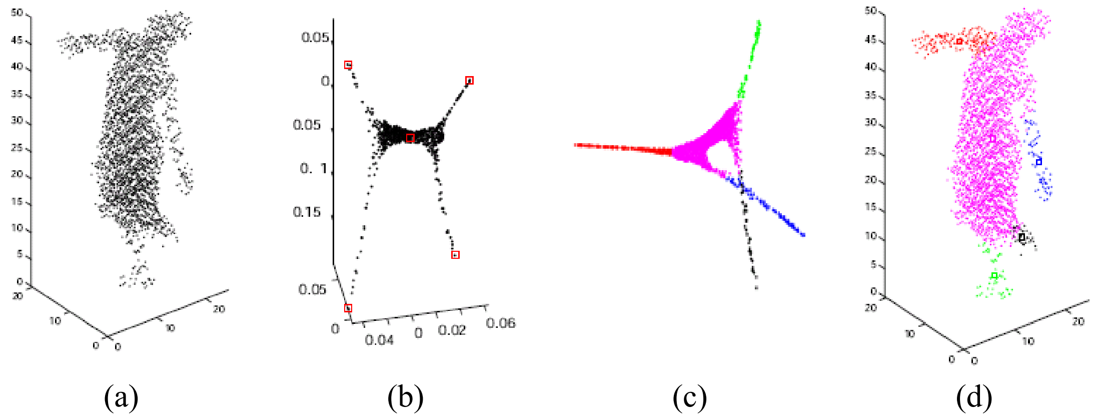


Figure 2-19 The segmentation process of [10]: (a) input 3-D voxel data (b) LLE transformation results and branch termination detection (c) segmentation using k-wise clustering in the embedded space (d) segmentation results in the normal space

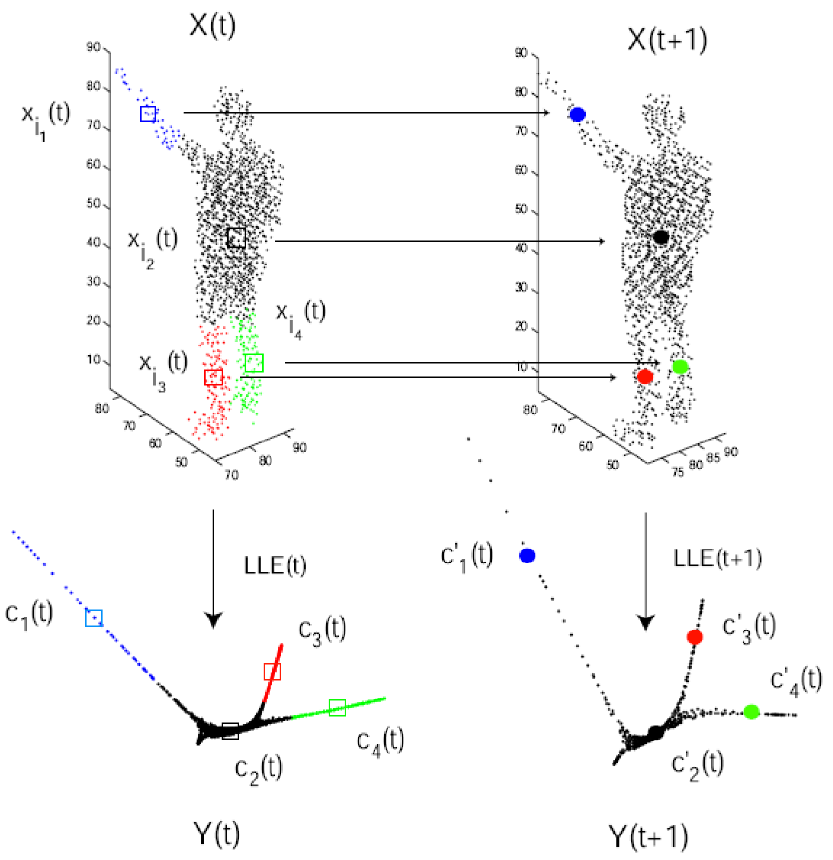
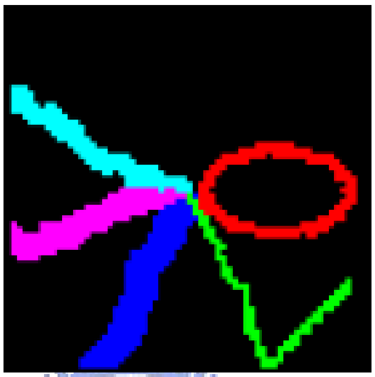
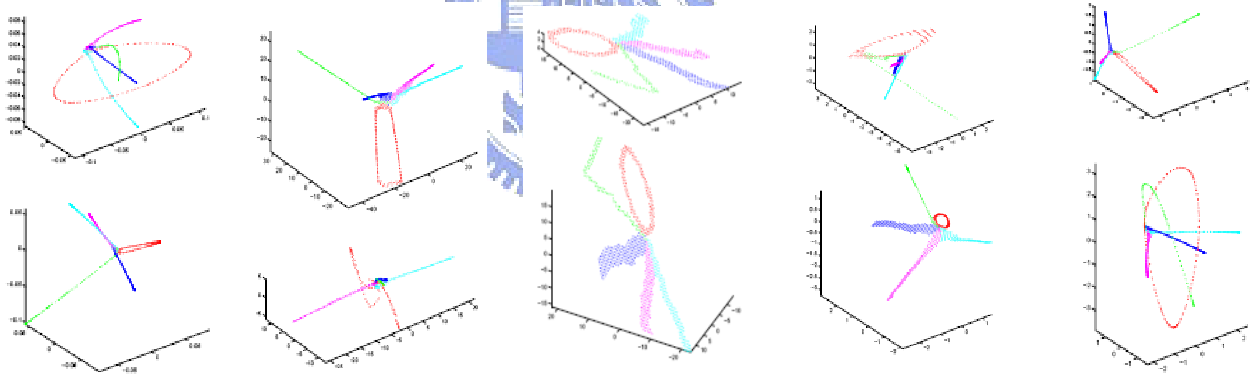


Figure 2-20 Clustering seeds are propagated along time to ensure temporal consistency [10]

In [6], Sundaresan proposed a segmentation approach for pose estimation based on Laplacian Eigenmap. Unlike ISOMAP, LE tries to preserve the geometric structure instead of geodesic distances. In this approach, different branches in the normal space, such as separated body parts, are transformed into distinguishable smooth curves in the embedding space. Compared with other popular manifold learning methods, only LE and Diffusion map have this special property, as shown in *Figure 2-21*. This property makes the segmentation of 3-D human body a lot easier. Moreover, since Diffusion map is only a variation of LE, LE is chosen in this thesis for its low computational cost.



(a) Test image



(b) LE (c) ISOMAP (d) MDS (e) LLE (f) Diffusion map

Figure 2-21 Embedding results using different manifold learning techniques [6]

LE also has two extra important properties:

- Different branches in the normal space are mapped into different curves in the eigenspace. The higher the dimensionality of the eigenspace is, the better the discriminative capability is.
- Braches with the length-over-width ratio greater than 2 can be mapped to smooth curves. Moreover, due to the preservation of geometric relationship, one can infer the position of the node in the branch by its position in the curve.

To sum up, n chains whose lengths are twice longer than their widths can be mapped into n discriminative smooth curves in the eigenspace whose dimension is between n-1 and 2n.

After transformation, nodes in the curves can be segmented using spline fitting. In the eigenspace, each node has its “site value” which represents its position along the chain. The site values are used to extract the skeletons from the visual hull. A generic human model is then fitted to the skeleton data in order to estimate the posture.

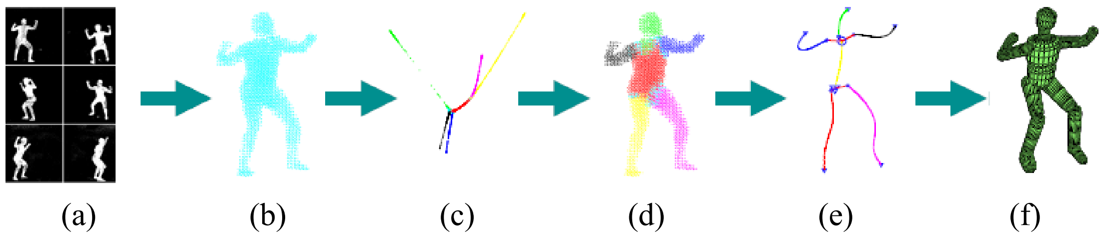


Figure 2-22 Pose estimation process of [6]: (a) input images from multiple cameras (b) 3-D voxel data acquired by space carving (c) transform the 3-D data using LE and segment it using spline fitting (d) project the segmented chains back into the normal space (e) skeleton extraction (f) top-down pose estimation

Chapter 3. Proposed Method

In this thesis, we proposed an efficient initialization process and a robust markerless pose estimation system. The goal of pose estimation is to capture the motion of a specific person. The motion of the articulated body parts is described using some parameters of a generic human model. The motion capture technique has been widely applied to different areas, such as surveillance, computer animation and biomechanical engineering. As mentioned earlier, markerless systems are more flexible and are applicable to different scenarios. However, the major problems of markerless pose estimation systems are the high dimensionality of the search space and the difficulties in estimation and labeling caused by deformable muscles. Using images from a set of synchronized and calibrated cameras, we can reconstruct the visual hull based on volume intersection. Then the 3-D voxel data are transformed into an embedding space using our modified Laplacian Eigenmap technique. Body parts segmentation is done in the eigenspace and the skeletons of the body parts are extracted individually. Finally, we fit the human model into the skeleton data using the PSO algorithm. Pose estimation is then iteratively performed for optimization. In , we show the flow chart of the proposed system.

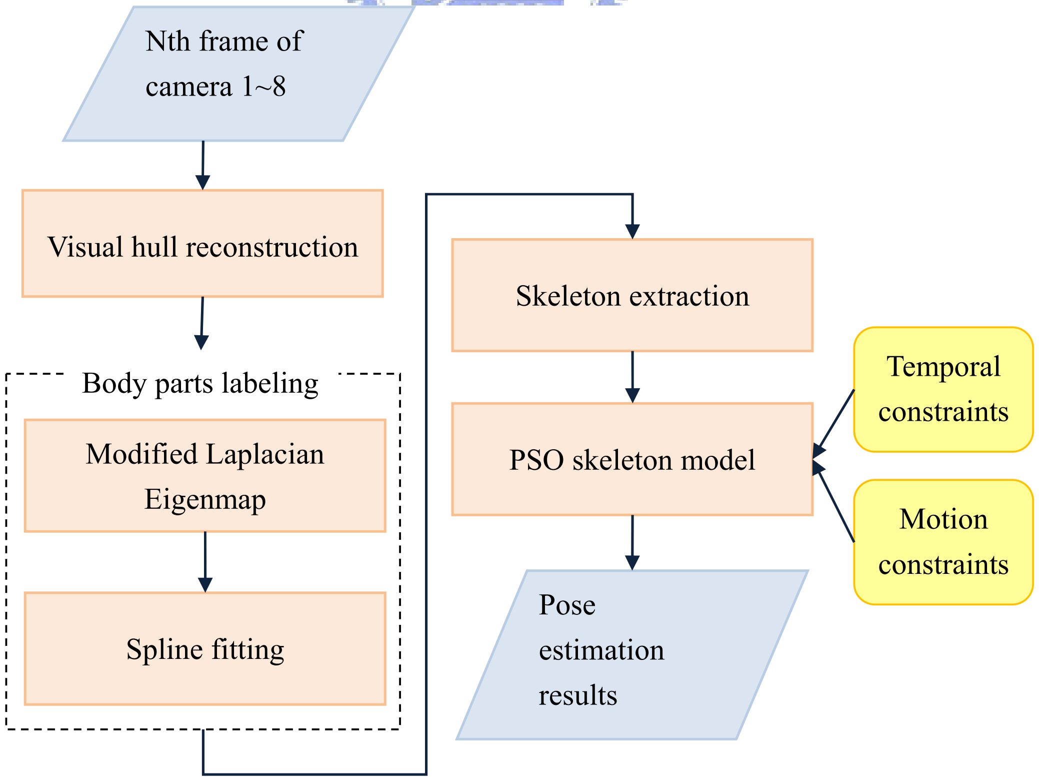


Figure 3-1 The block diagram of the proposed system

We have conducted our experiment in both the synthesized world and the real world environment. For video synthesis, the ObjectVideo Virtual Video (OVVV) [11] is used to simulate different kinds of the camera setup. For the real world environment, four cameras mounted on the ceiling of our laboratory are used. We will discuss these two different environments and explain why we choose OVVV as our simulation tool.

3.1 Initialization of Motion Capture

The initialization of our motion capture systems includes image acquisition, camera calibration, visual hull reconstruction, human body parts segmentation, and skeleton extraction. We will discuss the initialization of our pose estimation system in the following sections.

3.1.1 Images Acquisition

Images are acquired in both the virtual world and the real world. As mentioned earlier, OVVV is our simulation tool for different camera setups. OVVV is developed by ObjectVideo and is based on the game engine offered by "Half Life2." The architecture of OVVV is shown in *Figure 3-2*. The virtual cameras can be set up independently with different locations, PTZ parameters, fields of view, frame dimensions and frame rates. After setting up the cameras, virtual video can be rendered and simulated for various kinds of scenarios. With the help of OVVV, the experimental scenarios can be easily repeated and rearranged. This saves a lot of time and cost. On the other hand, for the real world environment, the sequences are acquired by four synchronized cameras mounted in the ceiling of our laboratory. The configuration of these cameras will be discussed later.

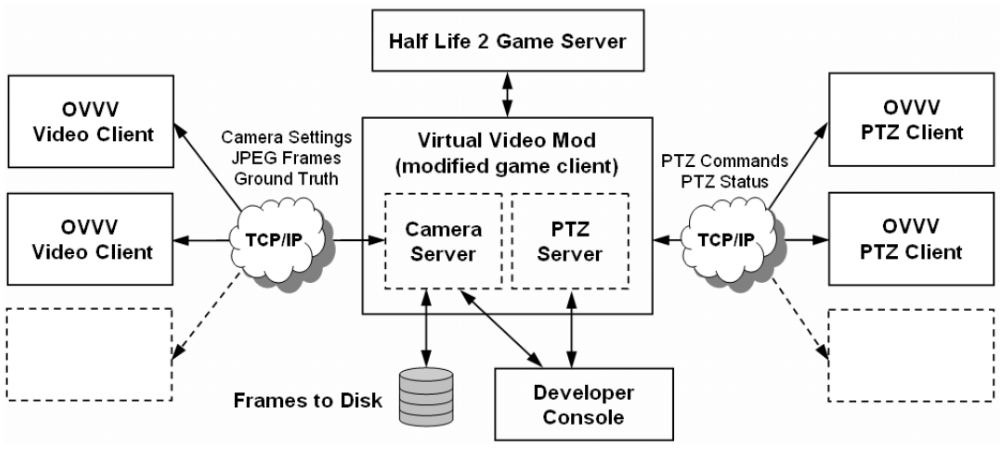


Figure 3-2 The architecture of OVVV [11]

3.1.2 Camera Calibration

Camera calibration provides the internal and external parameters of cameras. For the OVVV system, we can obtain these parameters directly from the ground truth. For the real environment, on the other hand, we choose [12] as our calibration tool. Chen has developed an efficient and robust technique for multiple camera calibration. With only two sheets of A4 paper, we can easily obtain the external parameters of four cameras. Once we have acquired the calibration data of multiple cameras in an offline manner, the visual hull can be reconstructed using the volume intersection method.

3.1.3 Visual Hull Reconstruction

Through background subtraction, we can obtain the foreground silhouette of the object in each camera. Visual hull reconstruction is then performed by the volume intersection technique. For the silhouette of each camera, the lines connected the camera center and a silhouette point forms a cone-like area. The intersection of these areas is the 3-D approximation of the target person.

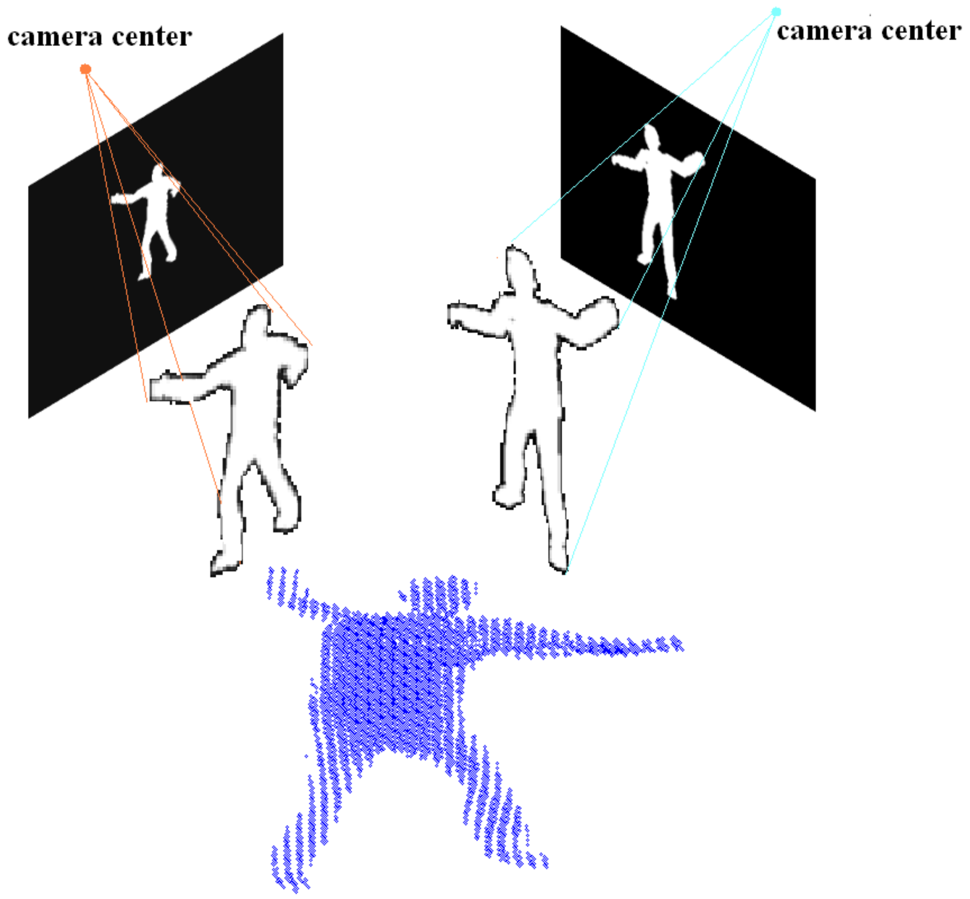


Figure 3-3 Visual hull reconstruction using volume intersection

In practice, we project each voxel to the image planes. If the projection falls inside one of the silhouettes, it gets one vote. A voxel is classified as “foreground” if the number of votes exceeds some predefined threshold. The higher the threshold is, the smaller the false alarm rate is. However, the probability of detection may also get decreased when we raise the threshold. Basically, this approach is more effective and is less computationally expensive than the direct implementation of the volume intersection method.

For the OVVV system, back-projections of voxels are evaluated using the calibration ground truth for each camera. Given the rotation angles (θ_x, θ_y, θ_z), the position (X, Y, Z) and the FOV (field of view) of each camera, the projection of voxels can be calculated as follows:

Rotation matrix:

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-\theta_x) & -\sin(-\theta_x) & 0 \\ 0 & \sin(-\theta_x) & \cos(-\theta_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textit{Eq. 3-1}$$

$$R_y = \begin{bmatrix} \cos(-\theta_y) & 0 & \sin(-\theta_y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(-\theta_y) & 0 & \cos(-\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textit{Eq. 3-2}$$

$$R_z = \begin{bmatrix} \cos(-\theta_z) & -\sin(-\theta_z) & 0 & 0 \\ \sin(-\theta_z) & \cos(-\theta_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textit{Eq. 3-3}$$

Translation matrix:

$$T = \begin{bmatrix} 1 & 0 & 0 & -X \\ 0 & 1 & 0 & -Y \\ 0 & 0 & 1 & -Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{Eq. 3-4}$$

Camera calibration matrix:

$$C = \begin{bmatrix} 0 & fov & 0 & 0 \\ 0 & 0 & fov & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \textit{Eq. 3-5}$$

Projection matrix:

$$Pj = C * R_x * R_y * R_z * T \qquad \textit{Eq. 3-6}$$

The back projection of a voxel ***p****: (****bp*** *vector)*

$$\begin{bmatrix} imP_1 & imP_2 & imP_3 \end{bmatrix}^T = Pj * p$$

$$\begin{bmatrix} bp_1 \\ bp_2 \end{bmatrix} = \begin{bmatrix} imP_1/imP_3 - img\_width/2 \\ imP_2/imP_3 - img\_length/2 \end{bmatrix} \qquad \textit{Eq. 3-7}$$

For the OVVV system, the calibration of multiple cameras refers to the same world coordinate. On the other hand, the four real cameras mounted in the ceiling of our laboratory are calibrated using Chen's method, which aligns all cameras' coordinates to one of the cameras' [12]. After calibration, the visual hull can be reconstructed in a similar way. The accuracy of the visual hull depends on several factors as listed below:

- The resolution of the voxels
- The accuracy of the calibration result
- The number and the configuration of cameras
- The synchronization among multiple cameras
- Foreground detection results

For these two experimental environments, we use 30mm×30mm×30mm voxels to compromise between computational cost and resolution. The calibration and synchronization is fairly accurate for the OVVV environment, since the data actually comes from the ground truth. In comparison, in the real-world environment, the back projections of voxels may deviate around four pixels, which are pretty acceptable. Besides, the background subtraction is used to detect foreground. A more precise background modeling technique based on the mean-shift algorithm is adapted to achieve robust results [19]. The major differences between the OVVV synthetical environment and our laboratory environment are the number of cameras and the configurations of cameras. In the real-world environment, four PTZ cameras are fixed in the ceiling, as shown in *Figure 3-4*. As for the OVVV environment, we can modify the number of cameras and setup of cameras to enhance the accuracy of the visual hull. Mündermann [17] suggested that the number of cameras should be more than 7 to reduce the artifacts in the visual hull. Four cameras aren't enough to provide reliable visual hull. Artifacts, such as ghost legs, will impair the results of body parts labeling and pose estimation. To understand how the number of cameras influences the reconstruction of visual hulls, we use OVVV to simulate the environment with eight cameras. As shown in *Figure 3-5*, the configuration of virtual cameras is circular, with equal height and the cameras are separated by 45 degrees. There are still many different setups for eight cameras. According to [17], the configuration with one camera above the target person and the others equally separated and surrounding the human body circularly will obtain the most accurate visual hull. To observe the artifacts of the visual hull, we first use the camera configuration shown in *Figure 3-5*. As for our experimental environment, most of our sequences are captured under the configuration suggested by [17] to alleviate the problem of artifacts.

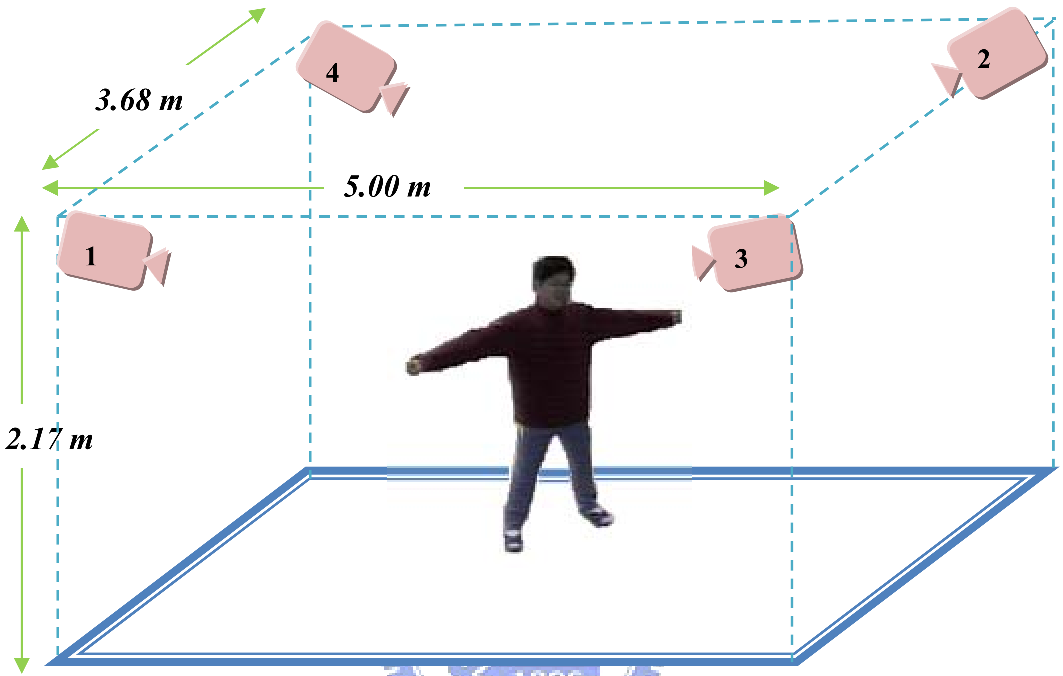


Figure 3-4 The setup of four cameras in our laboratory

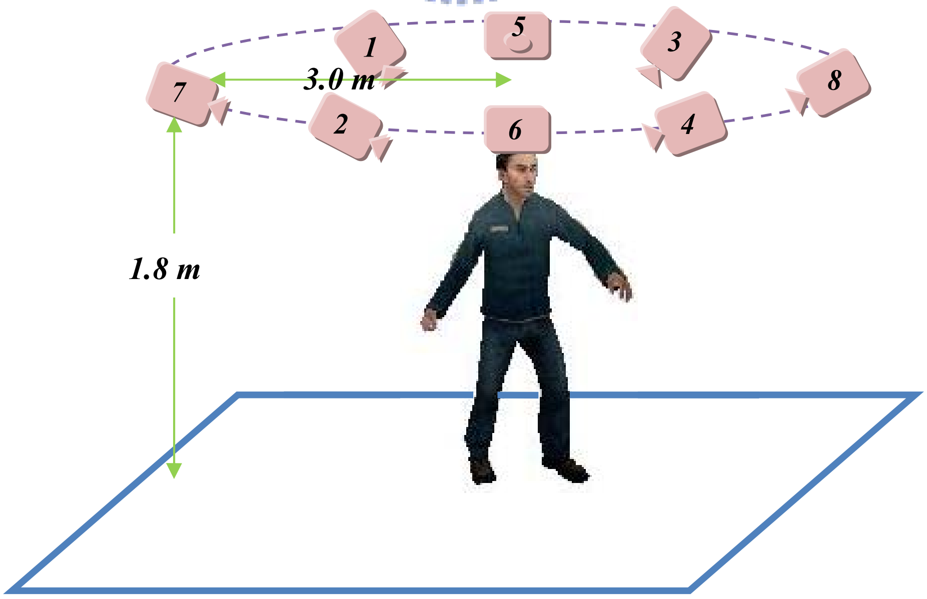
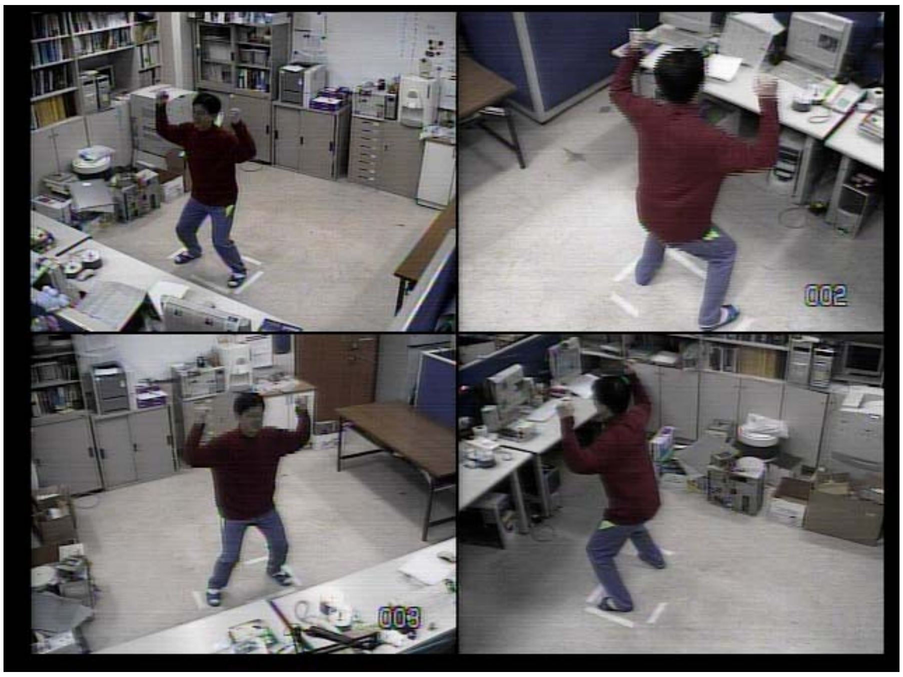
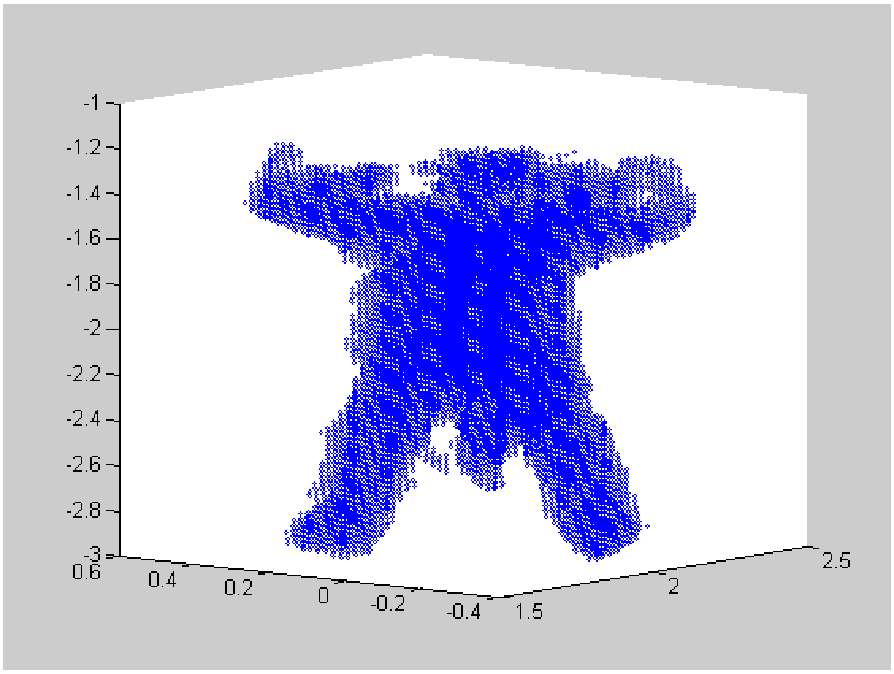


Figure 3-5 The circular configuration of the OVVV virtual cameras: eight cameras are separated by the angle of 45°

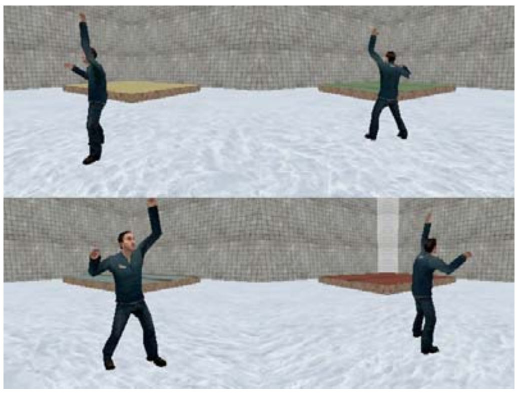


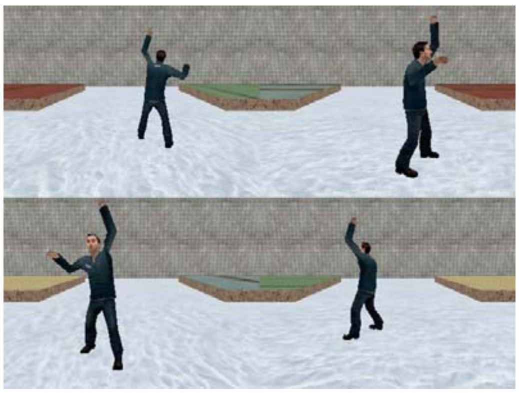
(a)

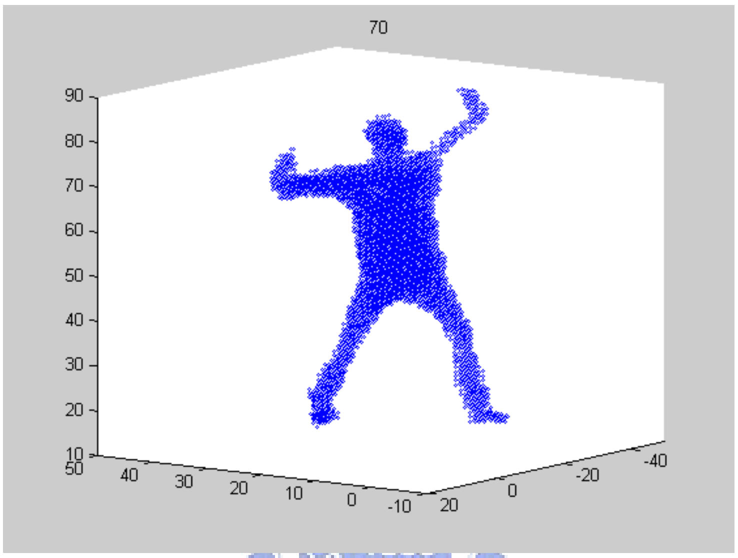


(b)

Figure 3-6 (a) four captured images in our laboratory (b) reconstructed visual hull


(a)

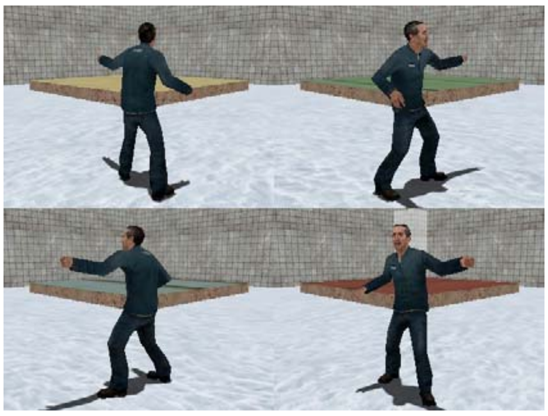

(b)

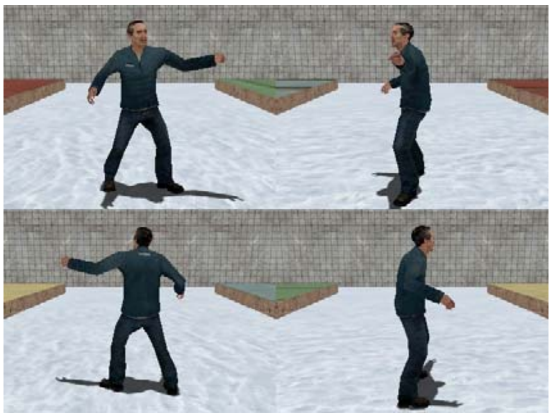


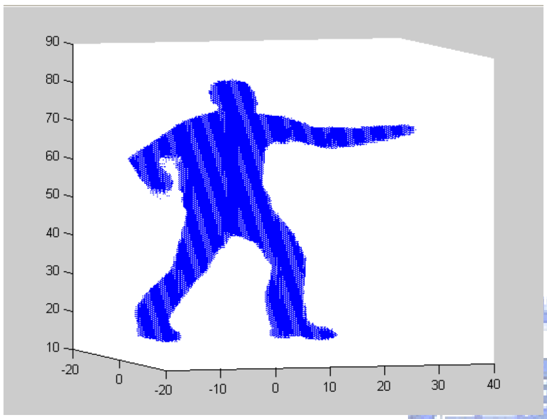
(c)

Figure 3-7 (a)(b) eight captured images in the OVVV simulation environment (c) reconstructed visual hull

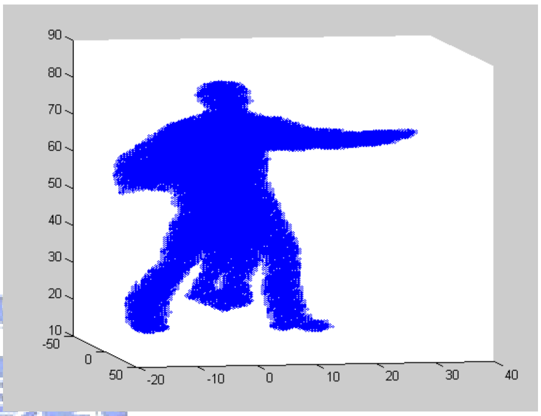
In *Figure 3-6*, we can see that the visual hull reconstructed in our laboratory has "ghost leg" and lots of artifacts around the arms. The serious artifact problem may result in inaccurate body parts labeling and poor pose estimation. For example, we don't know whether a leg is real or just is an artifact. On the other hand, the visual hull reconstructed in the OVVV is more accurate. To demonstrate the artifacts in visual hulls, we use OVVV to simulate two different environments with eight and four cameras separately. The 8-camera environment is just like the figure shown in *Figure 3-5*, while the 4-camera environment is an environment with four of the eight cameras in *Figure 3-5*.


(a)

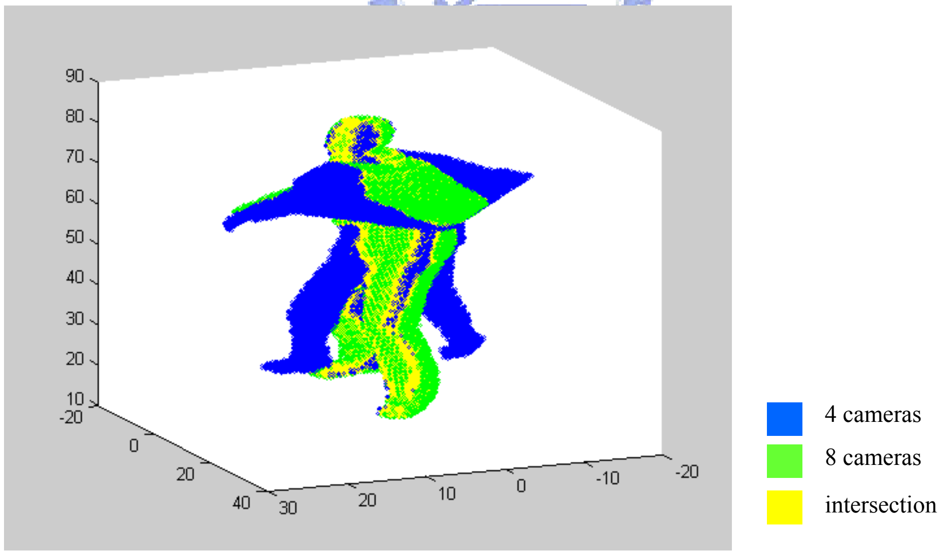

(b)



(c)



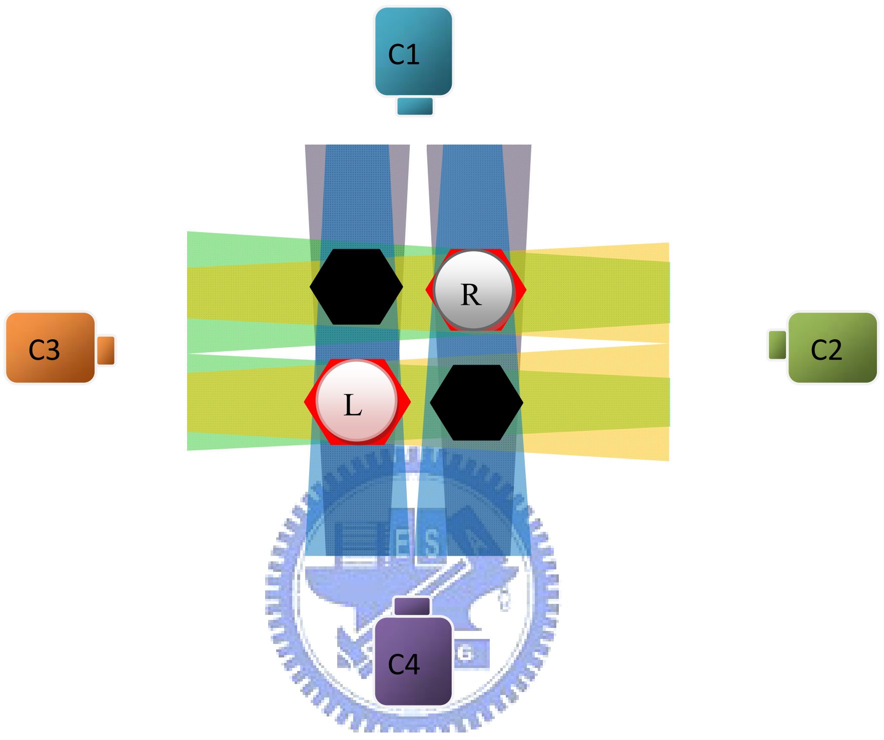
(d)



(e)

Figure 3-8 (a)(b) eight capture images from Camera1~Camera8. (c) reconstructed visual hull from eight cameras. (d) reconstructed visual hull from Camera1~Camera4. (e) the superposition of (c) over (d), where the green parts are the visual hull of (c) while the blue parts are that of (d). the yellow parts represent the overlap of (c) and (d)

For the case of four cameras, four apparent legs appear in the visual hull. Compared with that reconstructed from eight cameras, there are also some artifacts around the chest and the back. The formation of ghost legs can be easily illustrated in *Figure 3-9*. Due to insufficient number of cameras, ghost legs appear inside the intersection of the cone-like volumes. If we back-project the voxels of the four legs into the image planes, they all fall inside the silhouettes. Hence, it's difficult to identify which two legs are actually artifacts.



Top view

Figure 3-9 The formation of ghost legs: for clearness, the cone-like volume started from camera centers are simplified using trapezoid solid. The intersection is represented as red and black hexagons. Red hexagons are real visual hull while black ones are artifacts. Two circles marked "L" and "R" represented left leg and right leg of a specific person.

Serious artifacts, such as ghost legs, will result in inaccurate body parts segmentation. Especially when the calibration isn't so accurate and the foreground detection results have false positive or false negative, it is hard to remove these artifacts. In order to avoid serious artifacts in visual hull reconstruction, we choose OVVV as our simulation tool to acquire reliable visual hull from eight virtual cameras.

3.1.4 Human Body Segmentation

If we can segment each body parts separately, it will help pose estimation a lot. The search space is reduced and we can infer the posture based on the segmented body parts. Among recent research works, the embedding-based methods have the advantage of no model dependency and are robustness for various kinds of stature. These methods apply manifold embedding techniques to transform the original 3-D voxel data to another embedded space. Having preserved the intrinsic properties of the visual hull, the use of transformation makes it much easier to perform human body labeling.

3.1.4.1 Sunaresan's Method

In [6], Sundaresan has compared Laplacian Eigenmap with other popular manifold embedding methods. The Laplacian Eigenmap method possesses the capability of transforming a 3-D long branch into a smooth curve. After applying Laplacian Eigenmap, the segmentation of human body parts can be easily performed by applying spline fitting over the smooth curves. In Sundaresan's algorithm, the process of body part segmentation can be summarized as follows.

A. Laplacian Eigenmap Transformation

The first step of the Sundaresan's algorithm is to transform the visual hull into data in a six dimensional eigenspace. A graph G is constructed to record the relationship among voxels of the visual hull. For each voxel $\mathbf{v}_i$ in the visual hull, this approach checks the 6-adjacent neighbors of $\mathbf{v}_i$. If any of the neighbors, say $\mathbf{v}_j$, also belongs to the foreground voxels, then $\mathbf{v}_i$ and $\mathbf{v}_j$ are connected by an edge. The eigenvectors of the Laplacian of the graph corresponding to the six smallest nonzero eigenvalues are selected as the basis of the transformation map. In this step, each voxel $\mathbf{v}_i$ in the visual hull is transformed to a 6-D vector $\mathbf{u}_i$.

B. Segmentation in the eigenspace

In the LE transformation, different body parts are transformed into separated smooth branches. The segmentation of body parts is then performed using a spline fitting in the eigenspace. The use of spline fitting segments these branches into individual curves, with each curve labeling a body part in the normal space. The following paragraphs explain three major steps in spline fitting.

B-1、 Spline initialization

Points with the largest dimension are chosen as the starting points of branches.

Since the Laplacian Eigenmap encodes the geometric relation for the original voxel data, these starting points in the eigenspace correspond to the tip of each body parts in the normal space. For each branch, the nearest P nodes are selected around the starting point. The principal direction of the (P+1) points are evaluated using the PCA (Principal Component Analysis) method. Then these (P+1) points are projected into the principal direction to get their "site values" s_i. We can think of the principal direction as a ruler and the site values as the graduation where the projection of the point falls.

B-2、 Spline fitting

A 6-D cubic spline ***f*** is used to fit the site values s_i's. That is, ***f*** is chosen to minimize the fitting error:

$$\sum_i \left\| \boldsymbol{u}_i - \boldsymbol{f}(s_i) \right\|^2 \qquad \textit{Eq. 3-8}$$

B-3、 Spline propagation

The fitting process is propagated using the nearest N points at the end of (P+1) points. A new principal direction is recalculated if the angle between it and the previous one is greater than some predefined threshold.

B-4、 Spline termination

The spline propagation continues until the number of outliers exceeds a pre-defined threshold OT_1. A point is viewed as an outlier if its fitting error is greater than a predefined threshold OT_2.

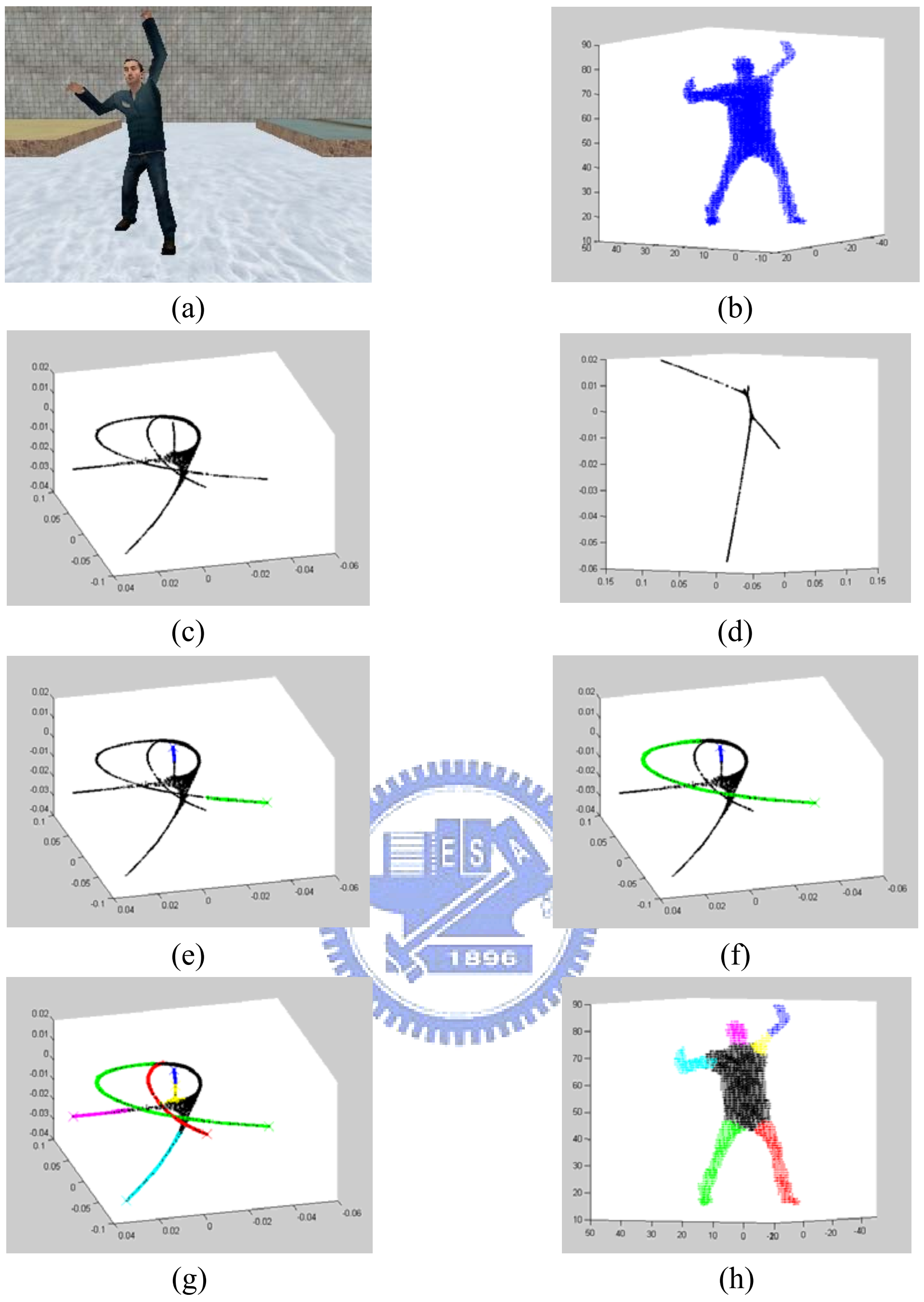


Figure 3-10 The body parts segmentation process of Sundaresan's method (a) one of the eight captured images (b) reconstructed visual hull (c)(d) the Laplacian Eigenmap transformation result: dimension1~dimension6 (e) spline initialization for the green branch in dimension1~3 (f) spline propagation and then termination for the green branch in dimension1~3 (g) six segmented braches: black points represent the unfitted ones (h) transform the segmentation results back to the normal space. Most of the nodes in the torso part are unfitted.

In *Figure 3-10*, the spline fitting method handles the segmentation of head and four limbs properly. However, the labeling of the trunk failed since its Laplacian Eigenmap transformation doesn't have a thin structure for spline fitting. Even though the Laplacian Eigenmap can transform a long branch into a smooth curve, the ratio of the length with respect to the width must be greater than 2 to ensure stable spline fitting. If the branch is not long enough, its transformation becomes a thick branch. When fitted by a spline, the nodes on the boundary of the thick branch may have larger errors and the number of outliers increases rapidly. As a result, thick branches usually have a premature termination. This causes the difficulty in segmenting thick body parts, such as the trunk of the human body. Furthermore, the site values for thick branches cannot be easily calculated. If the starting point is not in the middle of the thick branch, the principal direction may deviate from the medial line of that branch. Hence, the estimated site values may not be accurate enough. In the following, we illustrate this problem by a simple 2-D test image and its transformation into the 3-D eigenspace.

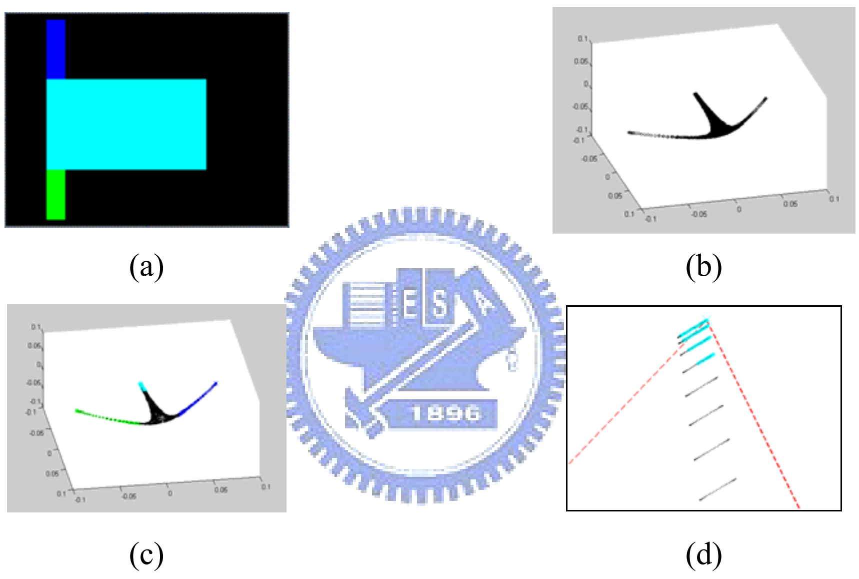


Figure 3-11 (a) test image (b) the result of Laplacian Eigenmap transformation. (c) the segmentation result in the eigenspace (d) illustration of the failed spline fitting for the thick branch. Here, the cyan branch is zoomed in for clearer illustration. The cross and the two dashed lines represent the starting point of the cyan branch and the first two principal components, respectively.

Figure 3-11 shows that it is difficult to fit a thick branch. From the starting point, the number of outliers increases rapidly due to the widely spreading nodes along the branch. This makes it difficult to clearly segment the thick branch from the others. The deviation of the starting point is also a problem. No matter which principal component is chosen for spline fitting, its direction doesn’t follow the actual trend of the branch. This may lead to inaccurate site values. Note that the site values play an important role in skeleton extraction since they trace the position of each node along the branch that encodes the intrinsic structure of the corresponding body part in the normal space. In this thesis, we'll propose a more efficient way to segment the body parts based on a modified Laplacian matrix. Furthermore, a new skeleton extraction method is also developed to overcome the problem in the segmentation of thick branches.

3.1.4.2 Modified Body Parts Segmentation Method

Inspired by Sundaresan's algorithm, we develop our initialization method based on a modification of the Laplacian Eigenmap.. Given n points $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ in the p-dimension, Laplacian Eigenmap aims to find its transformation $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n$ in the r-dimension to minimize the object function:

$$\sum_{i,j} \left\| u_i - u_j \right\|^2 E_{ij} \qquad \textit{Eq. 3-9}$$

where E is the adjacency matrix of the graph constructed from $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. That is, if $\boldsymbol{v}_j$ is in the neighborhood of $\boldsymbol{v}_i$, then E_{ij} is equal to 1. Otherwise, E_{ij} is set to zero. The value of E_{ij} can be defined by a value that is related to the distance between nodes $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$. Here, we define a heat kernel to compute the value of E_{ij}:

$$e^{-\frac{\left\| \boldsymbol{u}_i - \boldsymbol{u}_j \right\|^2}{k}} \quad \textit{where k is a predefined parameter.} \qquad \textit{Eq. 3-10}$$

Besides *Eq. 3-9*, an extra constraint is added for the minimization of the object function. The constraint says

$$U^T D U = I$$

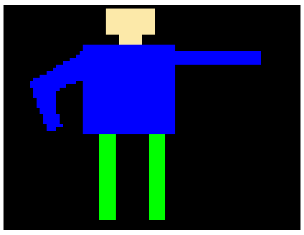
$$\text{where } U = \begin{bmatrix} \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_n \end{bmatrix}^T \qquad \textit{Eq. 3-11}$$

D is a diagonal matrix whose element D_{ii} represents the degree of Node i. This constraint is to normalizing the scaling factor when manifold embedding is performed. We can unroll *Eq. 3-11* to obtain the following constraints:

$$\begin{aligned} & u_{11}D_{11} + u_{21}D_{22} + \ldots + u_{n1}D_{nn} = 1 \\ & u_{12}D_{11} + u_{22}D_{22} + \ldots + u_{n2}D_{nn} = 1 \\ & \vdots \\ & u_{1r}D_{11} + u_{2r}D_{22} + \ldots + u_{nr}D_{nn} = 1 \end{aligned} \qquad \textit{Eq. 3-12}$$

In *Eq. 3-12*, we observe that nodes with more neighbors tend to converge to positions around the origin after the transformation. This explains the phenomenon that the tip of each body part is transformed to the tip of the branch.

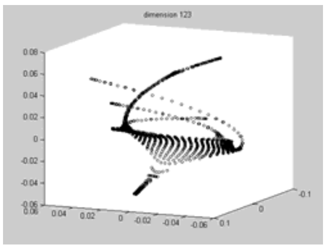
As aforementioned, the segmentation of trunk is a major difficulty in Sundaresan's method. Since the nodes in the trunk tend to have bigger values of D_{ii}, this fact makes the transformed values of the trunk voxels spread around the origin of the eigenspace. Having exploited this property of Laplacian Eigenmap, we manage to assign trunk voxels with bigger values of D_{ii} so that their transformed data will shrink even closer to the origin. Once these nodes are shrunk to the origin of the 6-D eigenspace, the segmentation of the limb parts will become much easier. Since most of the nodes in the trunk have their 6-connnection neighbors connected, their transformations tend to be drawn to the positions near the origin. Further away from the center of the torso, nodes have fewer connectivities and hence are far away from the origin in the eigenspace. Once these nodes with larger values of D_{ii} can be shrunk even closer to the origin of the 6-D eigenspace, the segmentation of the torso part can be performed easily using a simple threshold. Hence, in the modified version, the shrinkage of these nodes is accomplished by further increasing the value of D_{ii} to those nodes that have larger D_{ii} values in the graph. As a result, to satisfy the constraints in *Eq. 3-12*, the transformation results of these nodes as well as their neighbors will be shrunk even closer to the origin in the eigenspace. We illustrate this technique by a simple 2-D image firstly. Then we apply this modification over the Laplacian Eigenmap to segment visual hulls.



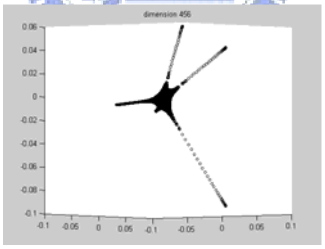
(a)

🟦 1st body part
🟩 2nd body part
🟦 3rd body part
🟥 4th body part
🟪 5th body part
🟨 6th body part
⬜ unfitted

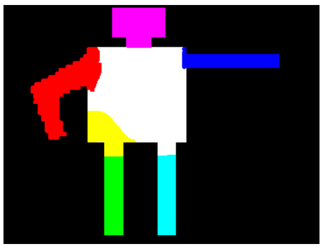
(b)



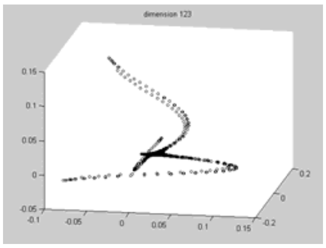
(c)



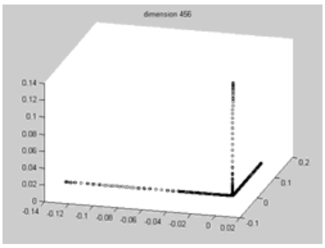
(d)



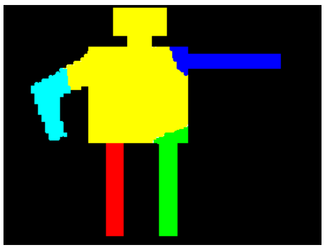
(e)



(f)



(g)



(h)

Figure 3-12 (a) test image (b) the color representation for segmentation results (c) LE transformation result in the first three dimensions (d) LE transformation result in the last three dimensions (e) segmentation result using the original LE. The white part is unfitted (f) modified LE transformation result in the first three dimensions (g) modified LE transformation result in the last three dimensions (h) segmentation result using modified LE

As shown in *Figure 3-12*, the 24-connectivity neighbors of each foreground pixel are checked. If there is a connection, the value of the corresponding E_{ij} in the adjacency matrix is assigned to one. Hence, the largest value of the degree of a foreground pixel is 24. The original LE transformation results are shown in (c) and (d). As stated earlier, the transformation of the torso pixels spread around the origin in the 6-D eigenspace. Unlike the other body parts, the transformed points are widely spreading and cannot be easily fitted using the same threshold. For (f) and (g), we increase the value of E_{ij} by 2 for those nodes with 24 connections. Since both torso and head have strong connectivity between their neighbors, their transformations are shrunk to the origin. The segmentation of each branch in the eigenspace is initialized and propagated in a same way as Sundaresan's method. However, in the modified version, the termination of the segmentation process can be more easily detected. Here, the growth of the branch stops as the branch approaches the origin of the coordinates. The segmentation result is shown in (h). Except for the head, the performance of (h) is better than that of (e). Note that these head nodes also converge to the origin. This is because the head part has strong connectivity within itself but is not long enough to generate a distinct branch in the eigenspace. To avoid the shrinkage of the head part, color information is added to the calculation of E_{ij}. In our approach, the color difference between two pixels i and j is defined as:

$$d \triangleq \sqrt{(r_i - r_j)^2 + (g_i - g_j)^2 + (b_i - b_j)^2} \qquad \textit{Eq. 3-13}$$

The larger d is, the less similar these two pixels are. Our strategy for using color information to assist the calculation of E_{ij} is stated below:

$$E_{ij} = E_{ij} + \begin{cases} kd + l & d < thd \\ h & otherwisie \end{cases} \qquad \textit{Eq. 3-14}$$

For some threshold *thd*, the smaller value of d will get higher positive weight on E_{ij}. On the other hand, we assign negative weights for those have less color similarity for penalty. We can visualize the curve of "color weight" in *Figure 3-13*.

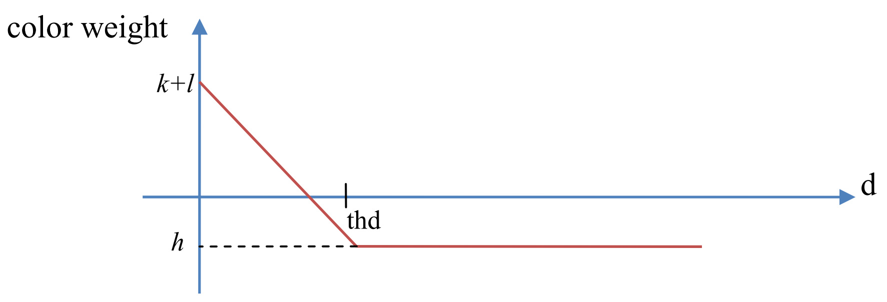
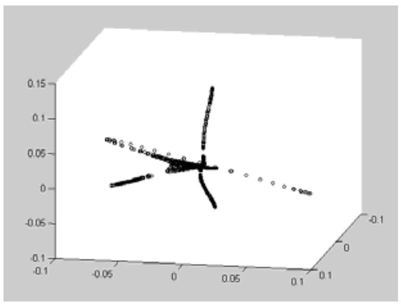
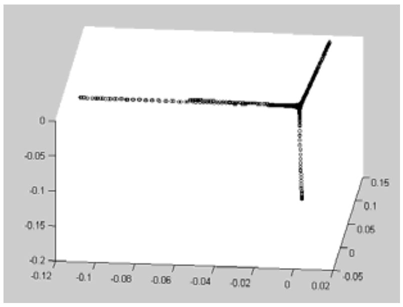


Figure 3-13 The color weight curve

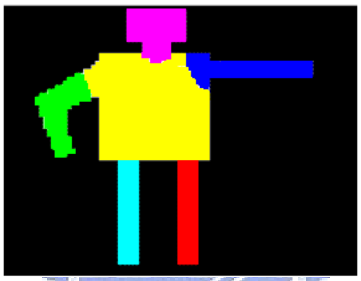
The linear part of the weight curve can be replaced with different kinds of functions. The color information is only auxiliary since it's not necessary that different body parts have dissimilar color. Therefore, we restrict the highest "bonus" for similar color to be one and the severest "penalty" for dissimilar color to be no less than -1. After constructing the adjacency graph based on position and color information, we increase the weight of nodes with the largest degree in a similar way to shrink them toward the origin. The experimental results are shown below.



(a)



(b)



(c)

Figure 3-14 The segmentation result using position and color information (a)(b) transformation and shrinking results in 6-D eigenspace (c) the segmentation result

In *Figure 3-14*, nodes which correspond to the head remain a distinct branch. The color information between torso and head is usually different and hence avoid them shrinking together. We apply this idea to the segmentation of visual hull in the 3-D space.

In summary, our body part segmentation method is briefly described as below:

A. Modified Laplacian Eigenmap

A-1、 Graph Construction

Given n voxels $v_1, \ldots, v_n$ in the visual hull, we construct an adjacency graph G for the voxels. In our case, the six adjacent neighbors of each voxel are checked. E is defined as the adjacency matrix of the graph. Each element E_{ij} of E records the relationship between Node i and Node j. For position information, if two nodes are 6-adjacent neighbors, E_{ij} gets one point. For color information, we project each voxel into the image planes and calculate the similarity between their colors. In our simulation, there are eight cameras in total in the scene. If for some camera the value of d, as defined in *Eq. 3-13*, is less than some threshold, one vote is recorded. Therefore, at most 8 votes can be recorded for each 6-adjacent neighbor of a node. The extra bonus on E_{ij} is added based on the following rule:

$$E_{ij} = E_{ij} + \begin{cases} a & total\_votes = 8 \\ b & th_1 \le total\_votes < th_2 \\ c & total\_votes < th_3 \end{cases} \qquad \textit{Eq. 3-15}$$

$0 < b < a \le 1 \quad -1 < c < 0$

Since the original value of E_{ij} is at most 1 for the position information, the bonus for the E_{ij} due to the color information is restricted to be no more than 1. Please note that the color information is only auxiliary. This is because different colors don't necessarily mean different body parts. Here, we simply use color information to prevent a mistaken shrinkage of the head part.

A-2、 Shrinking of Nodes

After having constructed the adjacency graph, we impose more weights on those voxels that have more connections to their neighbors. For a Node i, its degree is defined as $\sum_{j,j\neq i} E_{ij}$. Once we increase the weights of these nodes that have the largest degree, the transformation of these nodes will shrink toward the origin in the eigenspace. Also, their neighbors are drawn towards the origin as well.

B. Body Parts Segmentation

The following steps B-1 and B-2 are the same as Sundaresan's method. However, we proposed a more efficient termination method based on the modified LE.

B-1、 Spline Initialization

The starting point of a branch is the node that farthest from the origin. P nearest points are selected to initialize the piecewise spline fitting. Then we perform PCA on these (P+1) points to acquire the principal direction. The site values of (P+1) points are calculated by projecting them into the principal direction. Finally, a 6-D spline is used to fit the site value to minimize the fitting error *Eq. 3-8*.

B-2、 Spline Propagation

Starting from the end of the (P+1) nodes, the nearest N points are selected. A new principal direction is calculated based on these N nodes. If the angle between the previous and the present principal direction is less than some threshold, we can just use the previous one to evaluate the site values. Unlike Sundaresan's method, we don't have to count the number of outliers. The site values are also fitted using a 6-D spline.

B-3、 Spline Termination

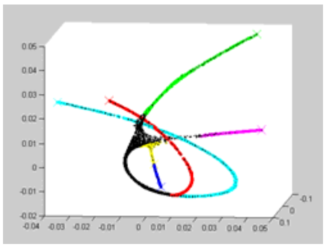
The process of spline propagation continues until the nodes are around the origin in the eigenspace. If the distance between the end of the spline and the origin is less than a pre-defined threshold, we terminate the growth process and complete the segmentation of one branch.

Based our modified body parts segmentation method, the experimental results are shown below.

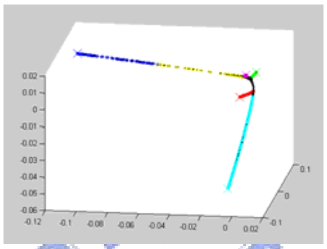
(a)

🟦 1st body part
🟩 2nd body part
🟦 3rd body part
🟥 4th body part
🟪 5th body part
🟨 6th body part
⬛ unfitted

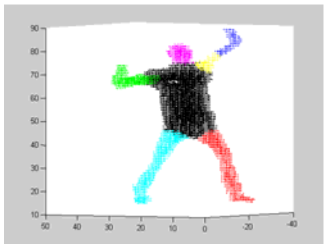
(b)



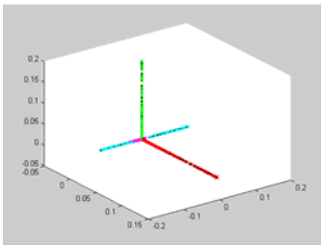
(c)



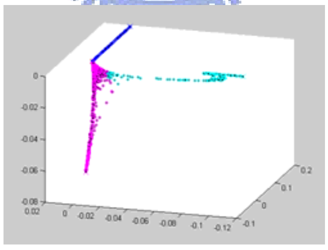
(d)



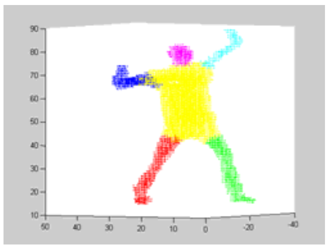
(e)



(f)



(g)



(h)

Figure 3-15 Comparison between Sundaresan's segmentation method and ours (a) one of the eight input images (b) the color representation for the segmentation results (c) the segmentation result in the first three dimension of the eigenspace using original LE (d) the segmentation result in the last three dimension of the eigenspace using original LE (e) the segmentation result in the normal space using original LE (f) the segmentation result in the first three dimension of the eigenspace based on our modified method (g) the segmentation result in the last three dimension of the eigenspace based on our modified method (h) the segmentation result in the normal space using our modified method

In *Figure 3-15*, we show the comparison between the Sundaresan's method and ours. It can be seen that our method map the trunk part to the origin of the eigenspace. Hence, after spline fitting, the trunk is well detected and the limb parts of the human body, especially the left arm, can be successfully extracted.

3.1.5 Skeleton Extraction

Once the segmentation of human body parts is done, we can extract the skeleton of the visual hull. Skeleton extraction has the advantage of feature reduction. Furthermore, skeletons encode the information of kinematic motion and don't deform for any pose. Therefore, we extract the skeleton data from the visual hull to analysis the motion of the human body. So far there has been some research about the skeleton extraction. As mentioned in Chapter 2, [7] applied the medial axis technique to extract the skeleton. Some research also uses the 3-D thinning methods to achieve the same goal. However, these methods are too sensitive to the minor change in the visual hull. The extraction results are noisy and have to be trimmed. Sundaresan proposed a more efficient way based on the Laplacian Eigenmap segmentation results [6]. Our skeleton extraction technique modifies Sundaresan's method to deal with short and thick braches more properly.

3.1.5.1 Sundaresan's Method

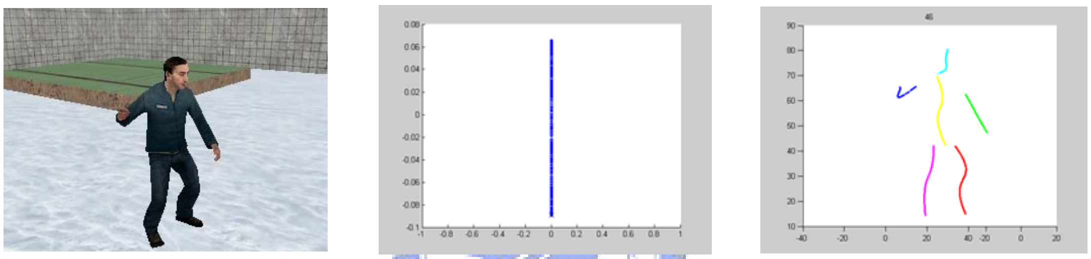
Sundaresan utilizes the site values which are obtained from spline fitting process to extract the skeleton from visual hulls. The transformation results of Laplacian Eigenmap encode the geometric relation between voxels in the normal space. For example, the fingertips usually correspond to the starting points of the branches. This is because Laplacian Eigenmap manages to make the nodes which are neighbors be close to each other and those which have the strongest connectivity scatter near the origin in the meanwhile. Therefore, the transformation results for the torso part distribute near the origin and the position of the node along the branch infers its geometric relation. The relation is also what site values encode. For the voxels of each body part, a 3-D cubic spline ***h*** is used to minimize the fitting error and extract the skeleton. The fitting error is evaluated as:

$$\sum_{v_i \in \text{some body part}} \left\| \boldsymbol{v}_i - \boldsymbol{h}(s_i) \right\|^2 \qquad \textit{Eq. 3-16}$$

However, as mentioned earlier, the site values of thick body parts are hard to computed since the transformation is not a smooth curve. Therefore, Sundaresan's method has difficulties dealing with the skeleton extraction for the torso part. We modify Sundaresan’s method to “individual skeleton extraction” which is stated in the next section.

3.1.5.2 Modified Skeleton Extraction Method

Inspired by Sundaresan’s method, we modify it to be suitable for every body part no matter how thick it is. Each body part is transformed into a 1-D eigenspace using LE separately. Other manifold learning techniques can be used as well. The smallest nonzero eigenvalue represents the most important dimension which corresponds to the trend of the body part. 1-D spline fitting and site value calculation are performed along this dimension. Since the transformation is in a 1-D eigenspace, our method has the advantage of being capable of dealing with every body part. The skeleton extraction is then performed by finding a 3-D spline ***h*** which minimizes *Eq. 3-16*. The experimental results are shown below.



(a) (b) (c)

Figure 3-16 The skeleton extraction result using proposed method (a) one of the eight input images (b) spline fitting and site value calculation in the 1-D eigenspace (c) the extracted skeleton

3.2 PSO Based Pose Estimation

After skeleton extraction, the posture of the specific person is estimated using a prior human model. The joints of the human model have their individual degrees of freedom (DOF). In total, there are usually 20 or more parameters in total. Thence, the fitting of the human model to the skeleton data is an optimization problem in a very high dimensional search space. In this case, it is very challenging to simultaneously find the optimal solution for the current skeleton data and to ensure the temporal smoothness over time. In the following section, we will discuss the adopted 3-D human model and the proposed pose estimation technique.

3.2.1 3-D Human Model

Our human model is a 3-D skeleton model. Here, we adopt the popular twists and exponential products formulation which is popular among the human models. This mathematical framework helps us in describing the kinematic chains in the human body. We will introduce the background of twists formulation firstly and then discuss our human model.

3.2.1.1 Twists and Exponential Products Formulation

The concept of twists and exponential products for kinematic chains is introduced by Murray [23] and is generalized to the application of 3-D human models by Mikic [13]. We can view the articulated structure of the human body as kinematic chains and the twists framework can be used to describe the rotation of the joints. It gives a concise description of the motion parameters and the motion constraints. Moreover, the relationship between the parameters and the position of the points in the model is simple. In this section, we present the concept of the twists framework based on Mikic's formulation.

Consider the rotation of a rigid object about a fixed axis as the simplest case. Given that the direction of the axis is a unit vector $\boldsymbol{\omega} \in \Re^3$ and $\mathbf{g} \in \Re^3$ is a point on the axis. After rotation by θ radians, the position of a point on the object is denoted as $\mathbf{p}(\theta)$. Its velocity is then expressed as

$$\boldsymbol{p}'(\theta) = \boldsymbol{\omega} \times (\boldsymbol{p}(\theta) - \boldsymbol{g}) \qquad \textit{Eq. 3-17}$$

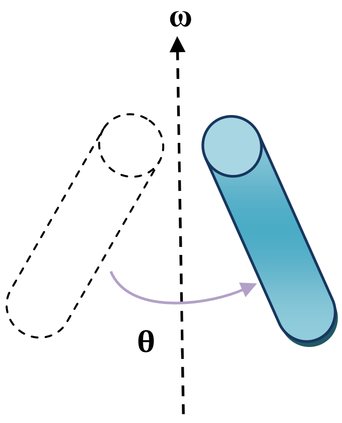


Figure 3-17 An object rotates θ radians about a fixed axis **ω**

We can reformulate *Eq. 3-17* in the form of the matrix and homogeneous coordinate representation:

$$\begin{bmatrix} p'(\theta) \\ 0 \end{bmatrix} = \begin{bmatrix} \tilde{\omega} & -\omega \times g \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p(\theta) \\ 1 \end{bmatrix} \triangleq \tilde{\xi} \begin{bmatrix} p(\theta) \\ 1 \end{bmatrix} \qquad \textit{Eq. 3-18}$$

where $\omega \times x = \tilde{\omega} x,\ \forall x \in \Re^3$. $\tilde{\xi}$ is defined as the twist which describes the rotation related to **ω** and **g**. It can be shown that the solution to *Eq. 3-18* is

$$\begin{bmatrix} p(\theta) \\ 1 \end{bmatrix} = \begin{bmatrix} e^{\tilde{\omega}\times\theta} & (\mathbf{I} - e^{\tilde{\omega}\times\theta})(\omega \times (-\omega \times g)) + \omega\omega^T(-\omega \times g)\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p(0) \\ 1 \end{bmatrix} \qquad \textit{Eq. 3-19}$$
$$\triangleq e^{\tilde{\xi}\theta} \begin{bmatrix} p(0) \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p(0) \\ 1 \end{bmatrix}$$

where

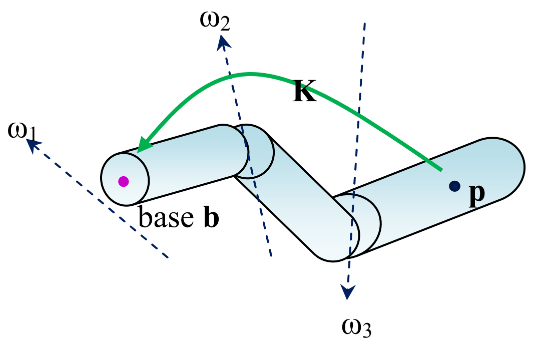
$$R = e^{\tilde{\omega}\theta} = \mathbf{I} + \frac{\tilde{\omega}}{\|\omega\|}\sin(\|\omega\|\theta) + \frac{\tilde{\omega}^2}{\|\omega\|^2}(1 - \cos(\|\omega\|\theta)) \qquad \textit{Eq. 3-20}$$

$e^{\tilde{\xi}\theta}$ is viewed as the exponential map from the initial position of the point **p** on the object to its new position after rotating θ radians. Besides, $e^{\tilde{\omega}\theta}$ corresponds to the rotation matrix **R** of the rigid object and **t** is the translation vector.

We can generalize this formulation to an open kinematic chain with m connected links. These links have their own rotation axes and different rotation angles. Let **K**(0) be the rigid body transformation which describes the position of the point on the last object of the chain, in terms of the base of the chain for the initial configuration. After the chain rotates $\Theta=[\theta_1\ \theta_2\ \dots\ \theta_m]^T$ for each link, the transformation **K**(Θ) can be reevaluated as:

$$K(\Theta) = e^{\tilde{\xi}_1\theta_1} e^{\tilde{\xi}_2\theta_2} \cdots e^{\tilde{\xi}_m\theta_m} K(0) = \begin{bmatrix} R(\Theta) & t(\Theta) \\ 0 & 1 \end{bmatrix}, \qquad \textit{Eq. 3-21}$$

where **R**(Θ) is the rotation matrix and **t**(Θ) is the translation matrix. The exponential product shown in the *Eq. 3-21* can be used to describe the position of the points on an open kinematic chain properly.



$$\begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \boldsymbol{b} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{b} \\ 1 \end{bmatrix}$$

Figure 3-18 The rigid body transformation of an open kinematic chain

3.2.1.2 3-D Skeleton Model

In our model, a human body is a 3-D skeleton composed of 12 segments and 23 parameters. as illustrated in *Figure 3-19*. It is based on the twists formulation and the position of each point in the model can be described using exponential products.

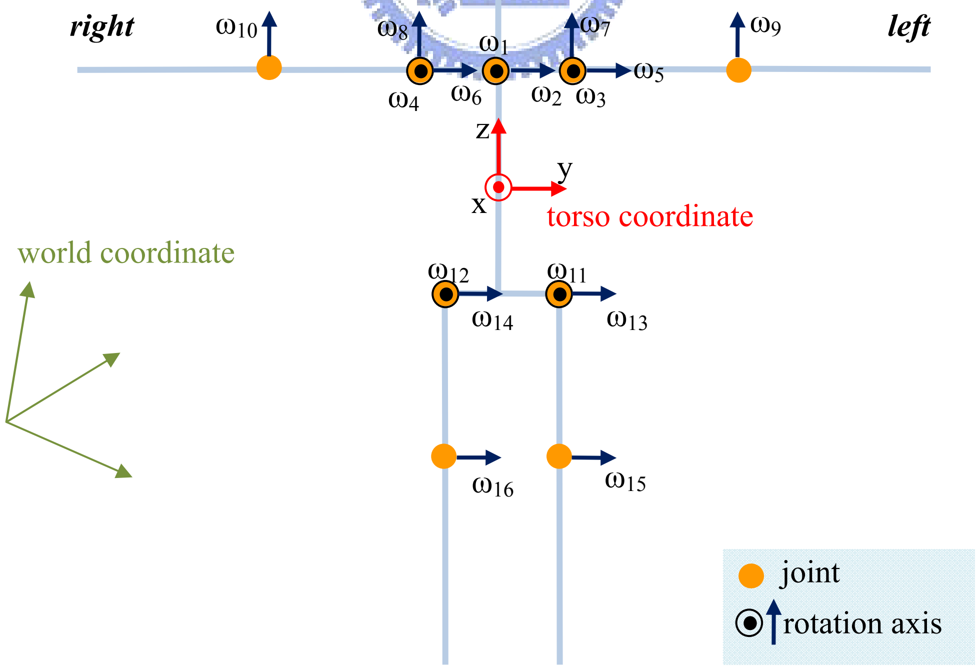


Figure 3-19 3-D human skeleton model

Table 3-1 The notation of the lengths of the body parts

Body part	Length
Torso	L_0
Head	L_1
Upper arm(left/right)	L_2
Lower arm(left/right)	L_3
Thigh(left/right)	L_4
Calf(left/right)	L_5
Shoulder	L_6
Hip	L_7

The human skeleton model refers to and is modified from Mikic's 3-D human shape model [13]. It consists of 16 rotation axes for 9 joints and the rotation is formulated using the twists framework that is referred to the torso coordinate system. For the rotation axes, their corresponding rotation angles are denoted as θ_1, θ_2, …, θ_{16}, separately. The lengths of the body parts are decided beforehand based on the stature of the target person. To describe the position of each point in the model, we exploit the rigid body transformation and twists formulation. Therefore, we have to define the initial body configuration to manipulate the model by 16 rotation angles. The initial configuration is defined as the pose which is shown in *Figure 3-19*. The initial values of the parameters and the positions of the joints are listed in *Table 3-2*.

The positions of the points in the human model can be calculated using twits and exponential products formulation. The values of the positions and rotation parameters are based on the torso-centered coordinate system. Based on the concepts of the twist framework, the position of each point in the human body can be described by the rigid body transformation. For example, the position of the wrist in terms of the torso-centered coordinate is decided by the kinematic chain which consists of the lower arm, the upper arm and the shoulder. The transformation is defined by the function $\mathbf{K}(0)$ in the initial configuration. When the body starts to move, the transformation can be adjusted to $\mathbf{K}(\Theta)$ via exponential products formulation. Furthermore, we have to decide the transformation between the world and torso-centered coordinate systems to transform the torso-centered system to the world coordinates. The transformation is determined by the rotation and the translation of the torso center relative to the origin of the world coordinate. In Mikic's design, the rotation axis of the torso is an arbitrary unit vector ω_0 in the world coordinates [13]. However, it is not easy to control the orientation of the human model. The orientation of the human model determines which part is the right hand side. This information is important since the motion constraints for the right side and the motion constraints for the left side are somewhat different. For example, the allowed rotation angle for the right elbow is [0,π]. However, it is [-π,0] for the left elbow. With the motion constraints, we can make our pose estimation more natural and reasonable. Unfortunately, the Mikic's method does have the problem in defining the orientation of the model. In *Figure 3-20*, we illustrate an example to explain this issue.

Table 3-2 The values of the parameters for the initial configuration

Joints	Position (torso-centered coordinate)	Rotation axis	Rotation angle
neck	$p_1 = [0 \quad 0 \quad L_0/2]^T$	$\omega_1=[1\ 0\ 0]^T$	$\theta_1=0$
		$\omega_2=[0\ 1\ 0]^T$	$\theta_2=0$
left shoulder	$p_2 = [0 \quad L_6/2 \quad L_0/2]^T$	$\omega_3=[1\ 0\ 0]^T$	$\theta_3=0$
		$\omega_5=[0\ 1\ 0]^T$	$\theta_5=0$
		$\omega_7=[0\ 0\ 1]^T$	$\theta_7=0$
right shoulder	$p_3 = [0 \quad -L_6/2 \quad L_0/2]^T$	$\omega_4=[1\ 0\ 0]^T$	$\theta_4=0$
		$\omega_6=[0\ 1\ 0]^T$	$\theta_6=0$
		$\omega_8=[0\ 0\ 1]^T$	$\theta_8=0$
left elbow	$p_4 = [0 \quad L_2 + L_6/2 \quad L_0/2]^T$	$\omega_9=[0\ 0\ 1]^T$	$\theta_9=0$
right elbow	$p_5 = [0 \quad -(L_2 + L_6/2) \quad L_0/2]^T$	$\omega_{10}=[0\ 0\ 1]^T$	$\theta_{10}=0$
left hip	$p_6 = [0 \quad L_7/2 \quad -L_0/2]^T$	$\omega_{11}=[1\ 0\ 0]^T$	$\theta_{11}=0$
		$\omega_{13}=[0\ 1\ 0]^T$	$\theta_{13}=0$
right hip	$p_7 = [0 \quad -L_7/2 \quad -L_0/2]^T$	$\omega_{12}=[1\ 0\ 0]^T$	$\theta_{12}=0$
		$\omega_{14}=[0\ 1\ 0]^T$	$\theta_{14}=0$
left knee	$p_8 = [0 \quad L_7/2 \quad -(L_4 + L_0/2)]^T$	$\omega_{15}=[0\ 1\ 0]^T$	$\theta_{15}=0$
right knee	$p_9 = [0 \quad -L_7/2 \quad -(L_4 + L_0/2)]^T$	$\omega_{16}=[0\ 1\ 0]^T$	$\theta_{16}=0$

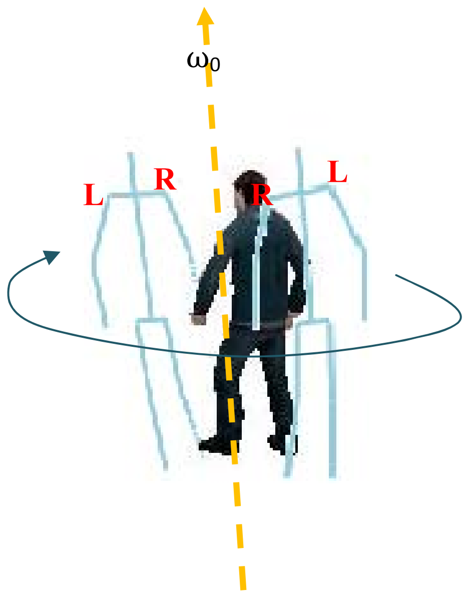


Figure 3-20 The weakness of Mikic's method

As shown in *Figure 3-20*, if the initial pose estimation is in the back of the target person, it has the wrong body orientation. When we rotate the human model to the right orientation, it causes even larger fitting error since the model rotates about the axis with an inclined angle. Therefore, the system may pick up the one with wrong orientation for our pose estimation result. To determine the right side from the left side, Mikic switches the right side and left side of the human model and compares their fitting errors. For the case in *Figure 3-20*, it happens that the smaller fitting error actually corresponds to the wrong decision. A more natural thinking is that if we can make the model self-spin, the orientation of the human body can be easily manipulated and decided. Hence, we redefine the rotation axis of the human body as the torso stick of the model. Furthermore, the neck position and torso center control the incline of the human model. With self spin, the human model can easily spin to the correct orientation to obtain less fitting error. To sum up, the rotation axis of the human body is decided by the positions of the neck joint and the torso center. That is, the orientation is performed by rotating the human model about the z axis of the torso coordinate.

The total number of the human model parameters is 23 in our case, which include $\hat{\boldsymbol{p}}_0$ (the position of the torso center respect to the world coordinate), $\hat{\boldsymbol{p}}_1$ (the position of the neck joint respect to the world coordinate), θ_0 (the spin angle) and 16 rotation angles for each joints. In the following context, we use ^ to indicate the positions respect to the world coordinate. Otherwise, the positions are in terms of the torso-centered coordinate system. The orientation of the torso is decided by the rotation matrix related to $\hat{\boldsymbol{p}}_0$ and $\hat{\boldsymbol{p}}_1$:

$$\boldsymbol{\omega}_0 = \hat{\boldsymbol{p}}_1 - \hat{\boldsymbol{p}}_0 \qquad \textit{Eq. 3-22}$$

$$\boldsymbol{R}_0 = \mathbf{I} + \frac{\tilde{\boldsymbol{\omega}}_0}{\|\boldsymbol{\omega}_0\|}\sin(\|\boldsymbol{\omega}_0\|\theta_0) + \frac{\tilde{\boldsymbol{\omega}}_0^{\,2}}{\|\boldsymbol{\omega}_0\|^2}(1-\cos(\|\boldsymbol{\omega}_0\|\theta_0)) \qquad \textit{Eq. 3-23}$$

Eq. 3-23 is derived from *Eq. 3-20*. Furthermore, we can apply the concept of exponential product to the calculation of the positions of each point in the human model. Consider the position **p**(0) of an arbitrary point in the model for the initial configuration. The rotations which influent the position of p is called the significant rotations by Mikic's definition [13]. For example, if p refers to the position of the fingertip, the significant rotations include the rotation angles of the wrist, elbow, and shoulder. According to *Eq. 3-21*, we can find the new position of the point p after the pose change from the initial configuration. Assuming there are m significant rotations for p, the new position of p will be:

$$\begin{aligned}\begin{bmatrix}\boldsymbol{p}(\Theta)\\ 1\end{bmatrix} &= \boldsymbol{K}(\Theta)\begin{bmatrix}\boldsymbol{p}(0)\\ 1\end{bmatrix}\\ &= e^{\tilde{\xi}_1\theta_1}e^{\tilde{\xi}_2\theta_2}\cdots e^{\tilde{\xi}_m\theta_m}\begin{bmatrix}\boldsymbol{p}(0)\\ 1\end{bmatrix}\end{aligned} \qquad \textit{Eq. 3-24}$$

$$\textit{where } \Theta = \begin{bmatrix}\theta_1 & \theta_2 & \dots & \theta_m\end{bmatrix}^T$$

Finally, we need to transform the torso-centered coordinate system to the world coordinate system via the rotation matrix of the torso. Hence, *Eq. 3-24* becomes:

$$\begin{aligned}\begin{bmatrix}\hat{\boldsymbol{p}}(\Theta)\\ 1\end{bmatrix} &= e^{\tilde{\xi}_0\theta_0}\begin{bmatrix}\boldsymbol{p}(\Theta)\\ 1\end{bmatrix}\\ &= e^{\tilde{\xi}_0\theta_0}e^{\tilde{\xi}_1\theta_1}e^{\tilde{\xi}_2\theta_2}\cdots e^{\tilde{\xi}_m\theta_m}\begin{bmatrix}\boldsymbol{p}(0)\\ 1\end{bmatrix}\end{aligned} \qquad \textit{Eq. 3-25}$$

We can reformulate *Eq. 3-25* in the form of Cartesian coordinate:

$$\hat{\boldsymbol{p}}(\Theta) = \boldsymbol{R}_0(\boldsymbol{R}_1(\boldsymbol{R}_2(\dots(\boldsymbol{R}_m\boldsymbol{p}(0)+\boldsymbol{t}_m)+\dots)+\boldsymbol{t}_2)+\boldsymbol{t}_1)+\hat{\boldsymbol{p}}_0 \qquad \textit{Eq. 3-26}$$

3.2.2 PSO Based Pose Estimation

PSO (Particle Swarm Optimization) has the advantages of being capable of dealing with nonconcave and nonlinear cost functions. Moreover, its computational cost is usually very light. This PSO method provides a powerful tool for dealing with an optimization problem in a high dimensional search space. Inspired by [8], we apply PSO to the fitting of the 3-D skeleton model to the extracted skeleton data.

3.2.2.1 Evaluation Function

In the process of pose estimation, we fit the 3-D skeleton model defined in *Figure 3-19* to the extracted and labeled skeleton. The skeleton model is composed of 12 line segments while the extracted skeleton data consists of many nodes in the 3-D space. The evaluation function calculates the error between the model and the extracted data. Hence our goal is to find the minima of the evaluation function. The major difference between our method and others is that we've already segmented the different body parts apart. When we fit the model to the extracted data, the correspondence of the body parts is restricted to one-to-one. We summarize the calculation of our evaluation function as below:

1. **The Euclidean distance between the model and the extracted skeleton**
   Since the extracted skeleton data doesn't have shoulder and hip, we only have to consider the other 10 line segments which include head, left/right upper arm, left/right lower arm, torso, left/right thigh, left/right calf. For each extracted body part i, we sample n_i nodes uniformly from m_i nodes for the purpose of speeding up. Then the same amount of nodes is sampled from every line segment. We illustrate the sampling process using *Figure 3-21*.

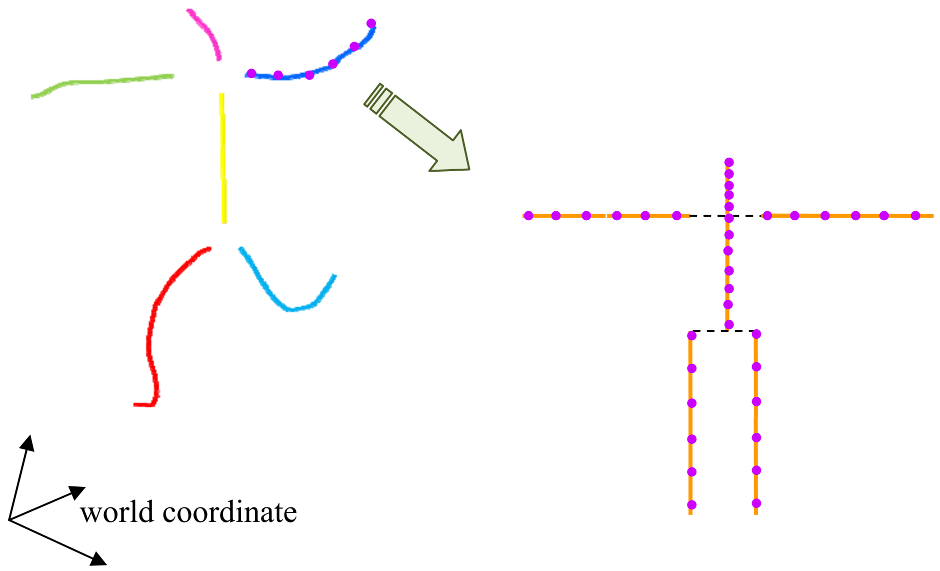


Figure 3-21 For the dark blue skeleton, we sample 6 nodes and use them to calculate the fitting error respect to the 6 different body parts (head, torso, arms, and legs) in the human model separately. 6 points are also marked from each body part segment.

After points sampling, Euclidean distance is calculated for each pair of points. We denote n_i nodes from ith body part of the extracted skeleton as $a_{i1}, a_{i2}, \ldots, a_{in_i}$ and that from jth body part of the human model as $b_{j1}, b_{j2}, \ldots, b_{jn_i}$. The numbers are ordered from one end of the body parts. The first problem is that how do we know the correspondence of the points between the extracted skeleton and the human model. That is, a_{i1} corresponds to b_{j1} or b_{jn_i}. We try these two kinds of association and pick up the one with less error. Therefore, for ith body part of the extracted skeleton, its fitting error corresponds to jth body part of the human model is calculated as:

$$Er_{ij} = \min\left(\sum_{s,t=1}^{s,t=n_i} dist(a_{is}, b_{jt}), \sum_{s,t=1}^{s,t=n_i} dist(a_{is}, b_{j(n_i-t+1)})\right) \qquad \text{Eq. 3-27}$$

2. **Total error**

From the previous step, we will obtain an 6x6 error matrix **Er**. Since we've labeled the body parts for the extracted skeleton, total error is calculated by finding the correspondence between body parts of the model and the extracted data which will acquire the minimum total error. To compare the fitness among different body parts, we should average the values of Er_{ij} by their number of sampled points. As a result, the total error is calculated by choosing a one-to-one mapping between i and j which minimize the sum of the error for 6 body parts.

From the above steps, we can obtain the fitting error given the values of 23 parameters. Next, we apply PSO to find the best match pose for the extracted skeleton.

3.2.2.2 Hierarchical PSO Fitting Process

Our pose estimation is performed by fitting the human model to the extracted skeleton. Since there are 23 parameters in total, it can be modeled as the optimization problem in the high dimensional search space. A swarm of particles and an evaluation function f are defined in the search space with the dimensionality of D. Each particle is represented as a vector $\mathbf{p}_i = [p_{i1}\ p_{i2}\ \ldots\ p_{iD}]^T$ with D elements. Furthermore, every particle has its associative velocity $\mathbf{v}_i=[v_{i1}\ v_{i2}\ \ldots\ v_{iD}]^T$ to guide its motion. In every iteration, the value of the evaluation function is computed and recorded for each particle. Two kinds of information are evaluated. The first kind of the information is the best position so far for each particle, recorded as $\mathbf{b}_i$. This $\mathbf{b}_i$ is to keep the information of self experience. The second kind of the information is the globally best position, denoted as **gb**. **gb** is evaluated by finding the minimal value of f so far. The new location of each particle is then updated using the information of self experience and the globally best position. Gradually, most particles will converge to the optimal position which has the minimal value of $f(\mathbf{p})$. The basic PSO based pose estimation process is described by the pseudo code:

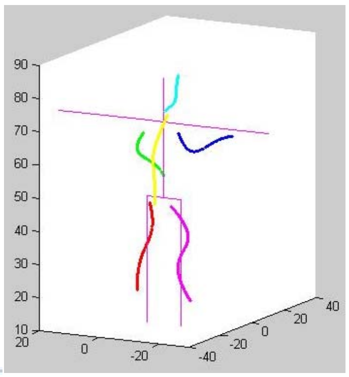
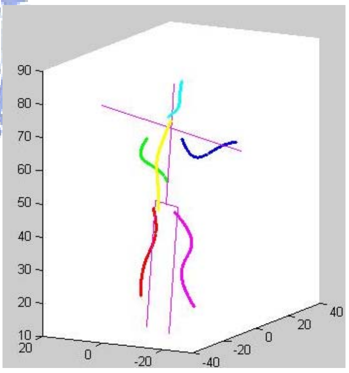
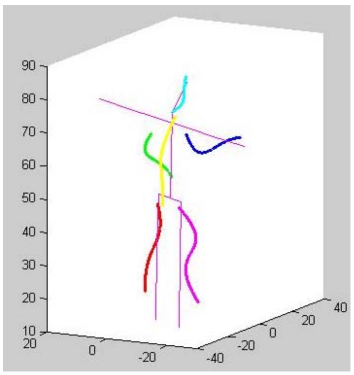
*Let the number of particles to be NP_0 with dimension D. Each dimension represents one parameter for the human model. In our case, D is equal to 23. The allowed values for each particle is [**a**,**b**].*

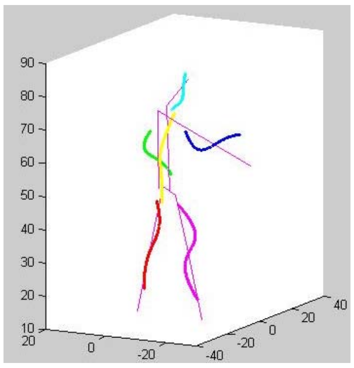
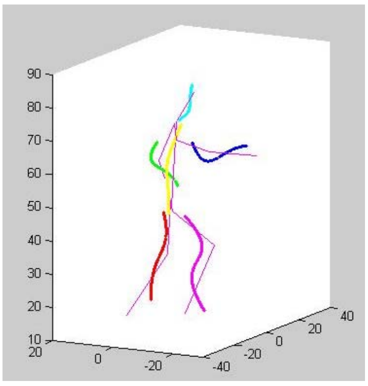
```
for each particle i
    the position of the particle: $\mathbf{p}_i=\mathbf{rand}(\mathbf{a},\mathbf{b})$
    (rand(a,b) means a random scalar or vector in the range of [a,b])
    initialize the velocity of the particle: $\mathbf{v}_i=[0\ 0\ldots0]^T$
    initialize the current best position of the particle: $\mathbf{b}_i = \mathbf{p}_i$
end of for
initialize the current globally best position: $\mathbf{gb} = \arg(\min_{\mathbf{p}_i}(f(\mathbf{p}_i)))$
while the stop condition isn't satisfied
    do
    for each particle i
        update its velocity and position
            $\mathbf{v}_i = t_1*\mathbf{v}_i+t_2*rand(0,1)*(\mathbf{b}_i-\mathbf{p}_i)+t_3*rand(0,1)*(\mathbf{gb}-\mathbf{p}_i)$
            ($t_1$, $t_2$ and $t_3$ are some constants)
            $\mathbf{p}_i = \mathbf{p}_i+\mathbf{v}_i$
    compute the value of fitting error to obtain $\mathbf{e}_i$ for $\mathbf{p}_i$
    if $\mathbf{e}_i < f(\mathbf{b}_i)$
      $\mathbf{b}_i = \mathbf{p}_i$
```

```
        end
        if $\mathbf{e}_i < f(\mathbf{gb})$
            $\mathbf{gb} = \mathbf{p}_i$
        end
end of while
```

Inspired by [1], we fit the model to the extracted skeleton hierarchically. The flow chart is shown below:

Table 3-3 The Hierarchical structure of our fitting process

stage	body parts to be fitted	example
1	torso center	
2	torso center torso orientation	
3	torso center torso orientation head	

4	torso center torso orientation head upper arms 、thighs	
5	torso center torso orientation head upper arms 、thighs lower arms 、calves	

As shown in Table 3-3, we hierarchically fit the human model instead of fitting 23 parameters at one time. At the first stage, we only consider where the best position of the torso center is. Therefore, there are only 3 parameters to be concerned. For the case of fewer parameters, fewer particles are needed and they can converge to the optimal solution more quickly. We can view the output result of this stage as the rough estimation of the position of the torso center. When we proceed to the second stage, we can lessen the search range of the torso center and focus on finding the best orientation. However, we still need to estimate the parameters of the body parts which are evaluated at the previous stages since they can be inaccurate due to being lack of the fitness information of the other body parts. Therefore, our hierarchical structure accumulates the parameters to be estimated. Furthermore, we lessen the range of values of the parameters which are estimated at previous stages. As for the stop criteria of the PSO algorithm, we set a threshold to restrict the number of iteration. The stages with more parameters to be estimated require more particles and iterations to converge to the global optimal solution.

3.2.2.3 Motion Constraints

The rotation of joints of the human body is restricted. If we don't limit the rotation angle for each joint, the PSO estimation result may be unnatural and inconsistent with the allowed body motion. Referring to [13], we set the following constraints on the rotation angles of the joints.

Table 3-4 The motion constraint for each joint

Joint	Rotation angle	Allowed range
Neck	θ_1	[-π/2,π/2]
	θ_2	[-π/2,π/2]
Left shoulder	θ_3	[-π,π]
	θ_5	[-π,π/2]
	θ_7	[-π/2,3*π/4]
Right shoulder	θ_4	[-π,π]
	θ_6	[-π/2,π]
	θ_8	[-3*π/4,π/2]
Left elbow	θ_9	[-π,0]
Right elbow	θ_{10}	[0,π]
Left hip	θ_{11}	[-π/3,π/2]
	θ_{13}	[-π,π/2]
Right hip	θ_{12}	[-π/2,π/3]
	θ_{14}	[-π,π/2]
Left knee	θ_{15}	[0,π]
Right knee	θ_{16}	[0,π]

For Mikic's method, the orientation of the human model is adjusted by switching the angle limits. For example, if we want to switch the right side and left side which are defined in *Figure 3-19*, we have to adjust the angle limits of θ_{15} and θ_{16} to [-π,0]. The orientation with fewer fitting errors is selected. This is bothersome and not intuitive. For our proposed model, the motion constraints remain unchanged even if it goes through self spin. The orientation of the human model can be manipulated easily and straightly.

3.2.2.4 Fine Tune the Pose Estimation Results

It is necessary to fine tune the PSO pose estimation results to make it more accurate. At section 3.2.2.2, we have used PSO to fit the model to the extracted skeleton. However, the number of parameters to be fitted is gradually getting larger. For higher dimension search space, PSO requires more particles and iterations to achieve the globally optimal solution. This will spend a lot of time converging to the global minimum. To speed up our system, we don't use mass particles and many iterations at the first time of pose estimation. The estimation results of section 3.2.2.3 are viewed as an initial guess of the body pose. Then the results are fine tuned for each body parts separately. When we adjust single body part, fewer parameters have to be considered and hence fewer particles are required and they converge to the optimal solution more easily and faster. Therefore, the positions and orientations of the torso, head, arms, and legs are fine tuned one by one to obtain more robust estimation result.

3.2.2.5 Temporal Consistency

The pose of the target person is estimated and the results are refined for each frame. Besides the use of the PSO method, we also need to ensure the temporal consistency between frames. The motion changes between the current frame and its previous frame should be smoothly changing. To ensure the temporal consistency, we propagate the values of the estimated parameters from the current frame to the next frame. In other words, we restrict the values of the parameters for the next frame to be within some range around the estimated results at current frame. However, since an incorrect estimation may also propagate over time, we add a re-initialization mechanism for each frame. When the fitting error is larger than some pre-defined threshold, we will reinitialize the whole pose estimation process based on the current frame only. This can prevent the propagation of errors.

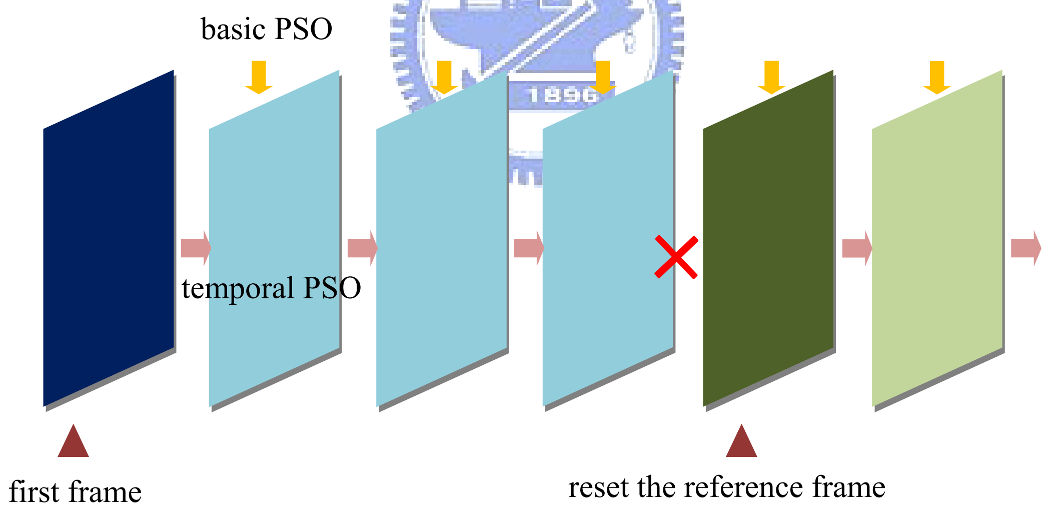


Figure 3-22 The mechanism that ensures the temporal consistency

In *Figure 3-22*, the basic PSO means the ordinary PSO with motion constraints. Temporal PSO means the one with both motion and temporal constraints. For each frame after the reference frame, the errors of basic PSO and temporal PSO are compared with each other. If the error of temporal PSO is greater than that of basic PSO by some threshold, the propagation process is cut off and the reinitialization mechanism is invoked to reset the reference frame. By doing this, we can ensure the temporal consistency and avoid error propagation at the same time.

Chapter 4. Experimental Results

The performance of our proposed method is evaluated via some sequences generated by OVVV. In these sequences, we use different actions to test our system. The names of sequences are listed below:

sequence NO.	sequence name	number of frames	camera configuration
s1	Wave_SMG1	38	*Figure 3-5*
s2	swing	168	*Figure 3-5*
s3	ThrowItem	400	*Figure 4-1*
s4	MeleeAttack01	190	*Figure 4-1*
s5	sitcouch1	170	*Figure 4-1*
s6	d1_t03_Tenements_Look_Out_Door_Close	780	*Figure 4-1*
s7	luggagepush	645	*Figure 4-1*
s8	walk_all	305	*Figure 4-1*
s9	d3_c17_03_throw_from_tower	1110	*Figure 4-1*

In these sequences, the target person performs different actions. These are built-in actions in OVVV. We use the script sequence in the Hammer tool, which is developed by the Valve company, to edit and control the motion of the target person. Then the virtual video tool offered by ObjectVideo is used to generate the scenarios.

Eight virtual cameras are set up to capture the images. We use two different setups of these cameras to test our system. One setup has been shown in *Figure 3-5*, while the other is shown below.

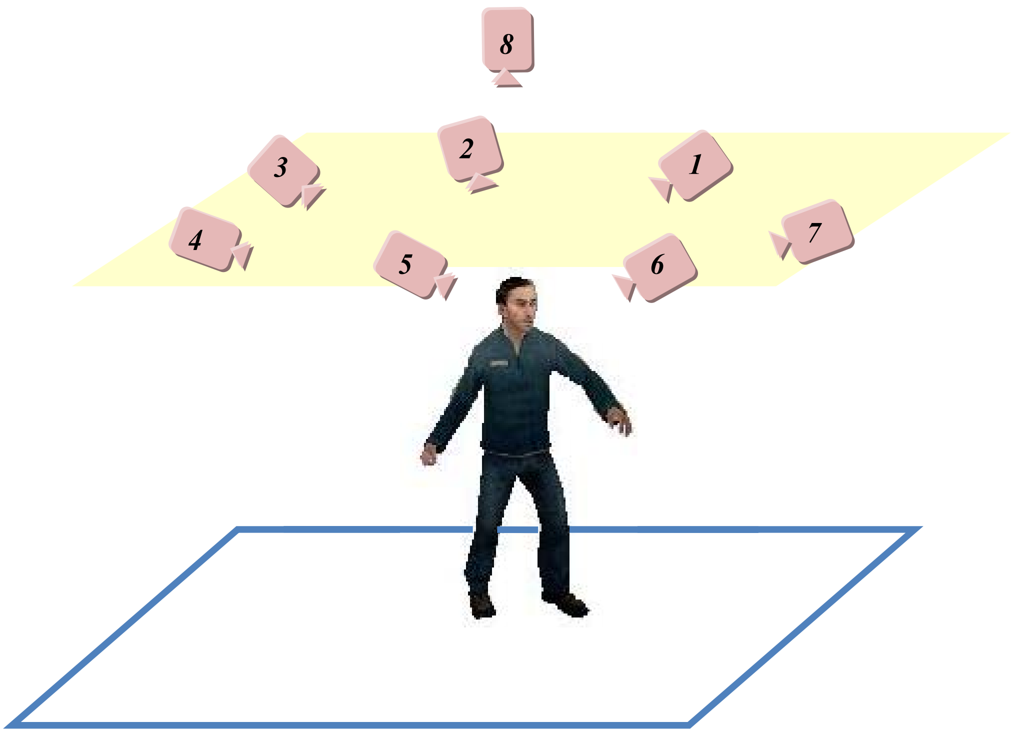
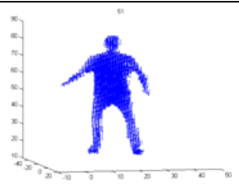
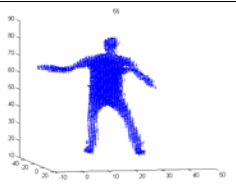
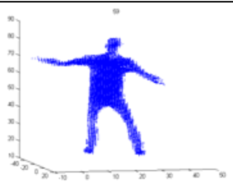
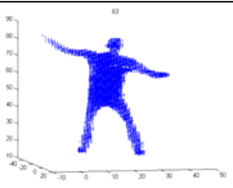
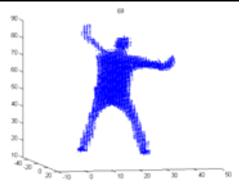
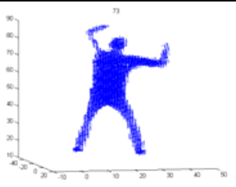
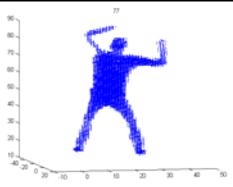
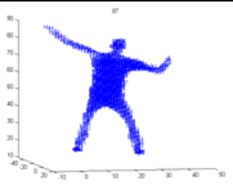
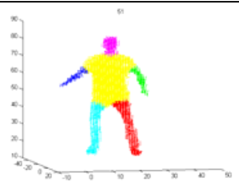
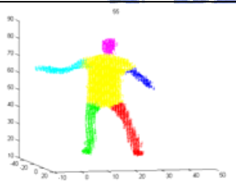
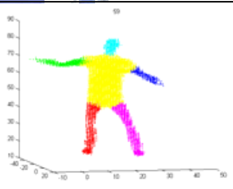
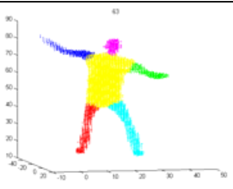
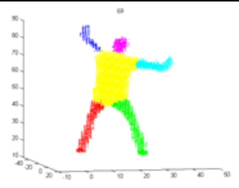
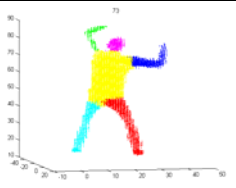
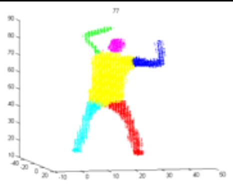
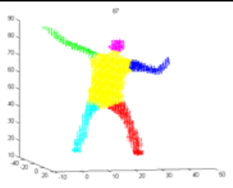
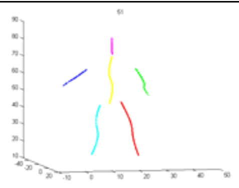
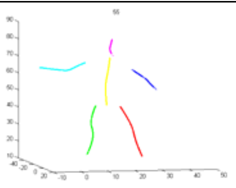
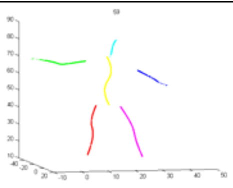
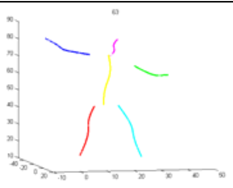


Figure 4-1 The second setup of eight cameras for oursynthetical environment

According to [17], the most favorable camera configuration for the case of eight cameras is shown in *Figure 4-1*. If compard to the configuration in *Figure 3-5*, this configuration may generate visual hulls with fewer artifacts. In each sequence, we adjust the rotation and zoom angles of the eight cameras to ensure each camera can shoot the whole person. In the capturing of a single video, these parameters of cameras are fixed and well calibrated. We use the ground truth offered by OVVV to obtain the external and internal parameters of these cameras. The frame size is 320× 240. Since the images are captured by connecting to the server of OVVV through internet, the frame rate is influenced by the working state of the computer and the internet. In our experiments, we set the frame rate to be approximately 30 frames per second. Some experimental results are shown below:

A s1: Wave_SMG1

The target person waves his hands in this sequence. For the sequence s1, we show the detailed results of visual hull reconstruction, body parts segmentation, skeleton extraction, and pose estimation.

time 51	time 55	time 59	time 63
time 69	time 73	time 77	time 87
			
time 51	time 55	time 59	time 63
			
time 69	time 73	time 77	time 87
			
time 51	time 55	time 59	time 63
			
time 69	time 73	time 77	time 87
			
time 51	time 55	time 59	time 63

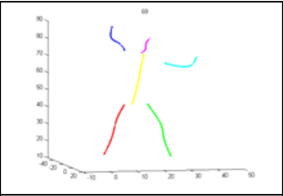
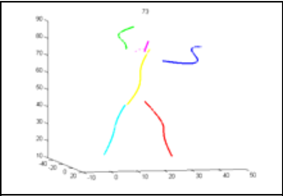
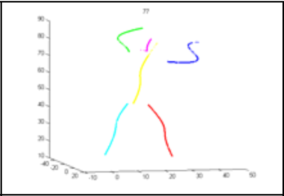
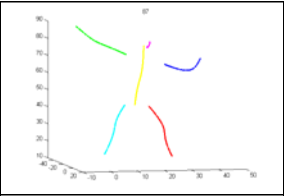
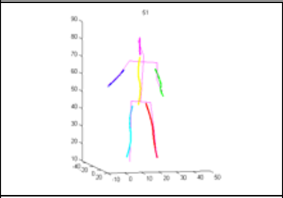
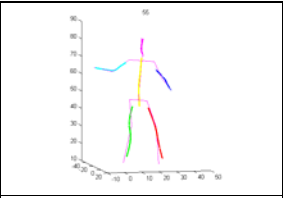
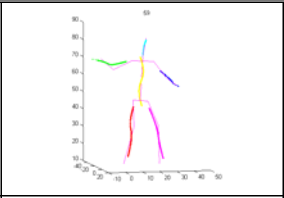
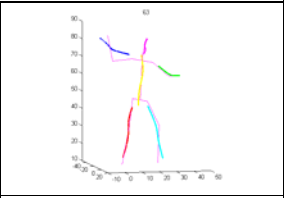
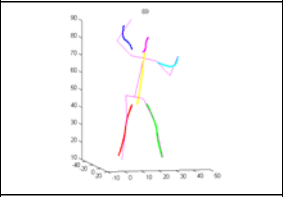
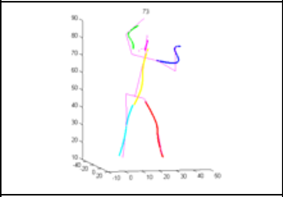
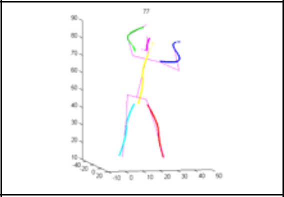
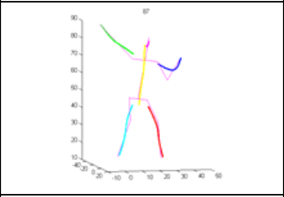
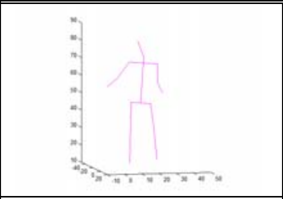
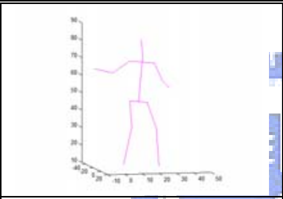
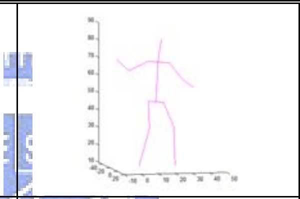
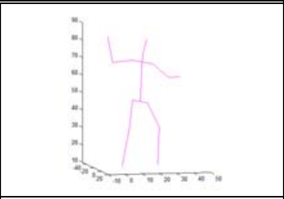
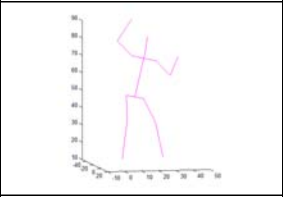
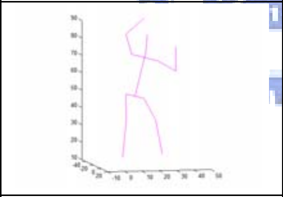
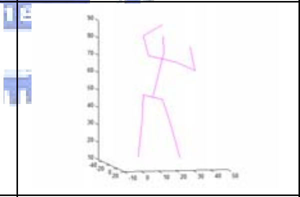
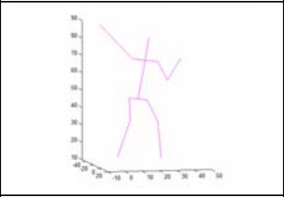
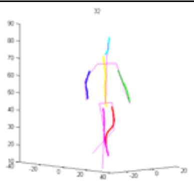
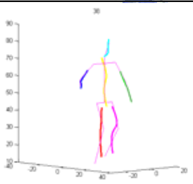
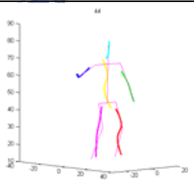
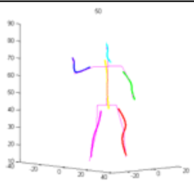
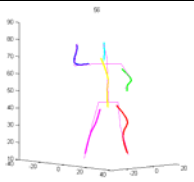
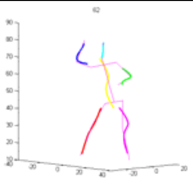
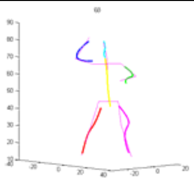
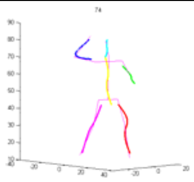
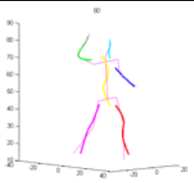
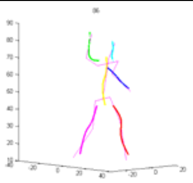
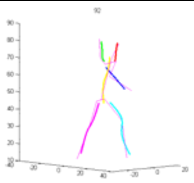
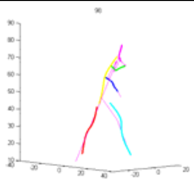
			
time 69	time 73	time 77	time 87
			
time 51	time 55	time 59	time 63
			
time 69	time 73	time 77	time 87
			
time 51	time 55	time 59	time 63
			
time 69	time 73	time 77	time 87

Figure 4-2 The pose estimation results of s1. The first two rows are the 8 different frames captured by the eight different cameras at a specific time instant. The 3rd and 4th rows are the reconstructed visual hulls. The 5th and 6th rows are the segmented visual hulls. The extracted skeletons are plotted in the 7th and 8th rows. The fitting results are shown in the 9th and 10th rows. The last two rows are the pose estimation results.

B. s2: swing

In this sequence, the target person throws a ball. To save space, we only show the results of skeleton extraction and pose estimation. The reconstructed visual hulls and the labeling of body parts are similar to that of s1.

time 32	time 38	time 44	time 50
time 56	time 62	time 68	time 74
time 80	time 86	time 92	time 98
time 110	time 130	time 160	time 190
			
time 32	time 38	time 44	time 50
			
time 56	time 62	time 68	time 74
			
time 80	time 86	time 92	time 98

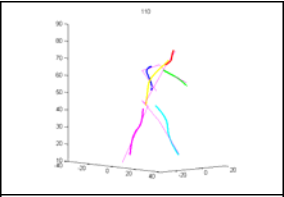
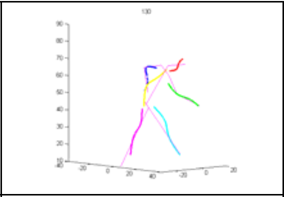
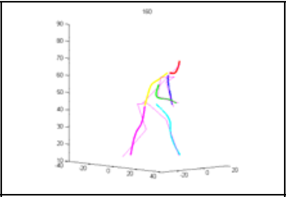
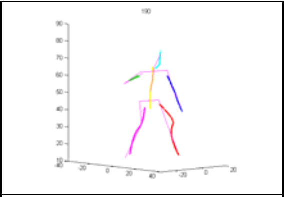
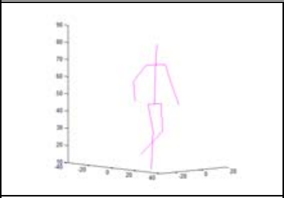
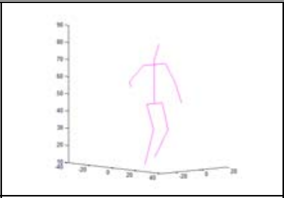
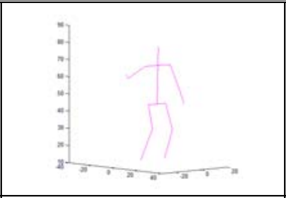
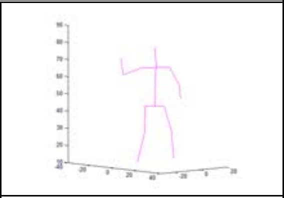
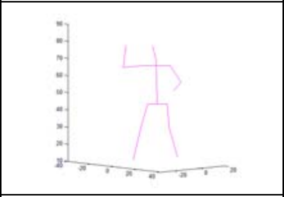
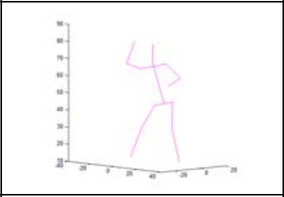
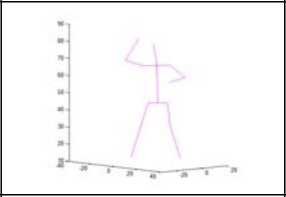
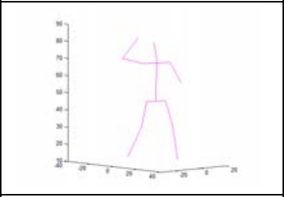
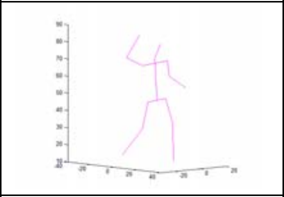
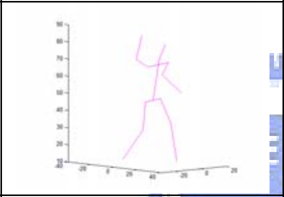
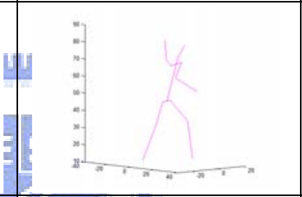
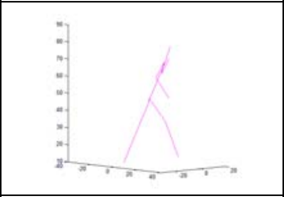
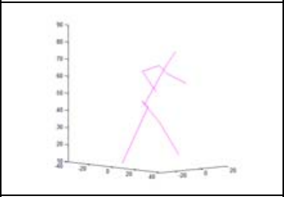
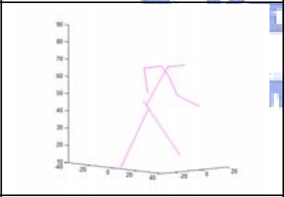
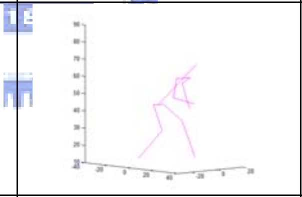
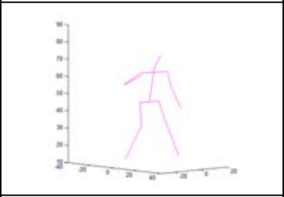
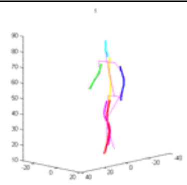
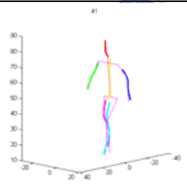
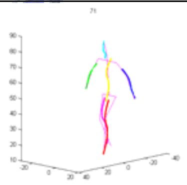
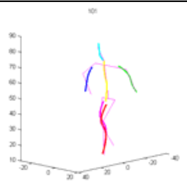
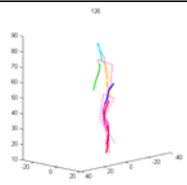
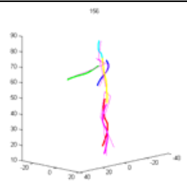
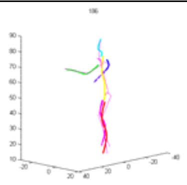
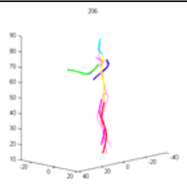
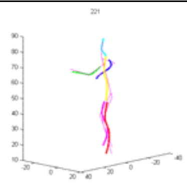
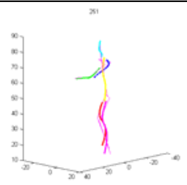
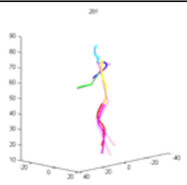
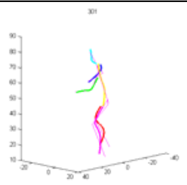
			
time 110	time 130	time 160	time 190
			
time 32	time 38	time 44	time 50
			
time 56	time 62	time 68	time 74
			
time 80	time 86	time 92	time 98
			
time 110	time 130	time 160	time 190

Figure 4-3 The pose estimation results of sequence s2. The 1st to 4th rows are the input images from one of the eight cameras. The 5th to 8th rows show the extracted skeleton and the fitting results. Pose estimation results are shown in the last four rows.

C. s3: ThrowItem

In this sequence, the target person throws something. We show the results of pose estimation below.

time 1	time 41	time 71	time 101
time 126	time 156	time 186	time 206
time 221	time 251	time 281	time 301
time 321	time 351	time 381	time 401

			
time 1	time 41	time 71	time 101
			
time 126	time 156	time 186	time 206
			
time 221	time 251	time 281	time 301

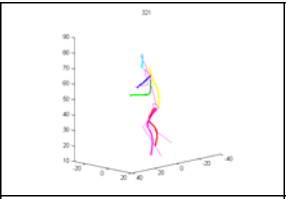
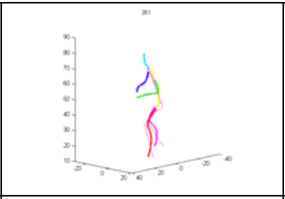
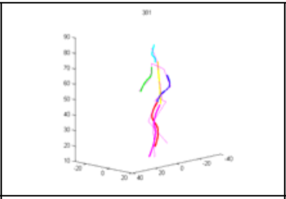
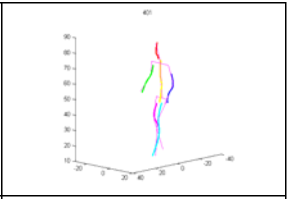
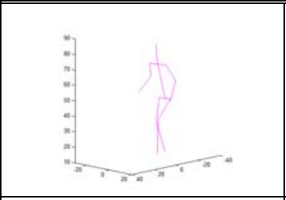
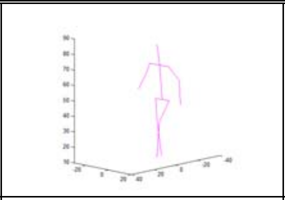
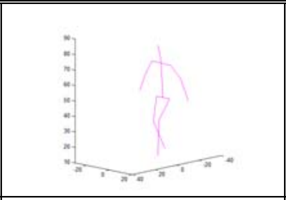
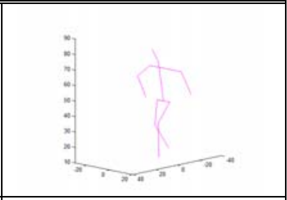
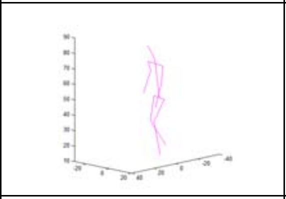
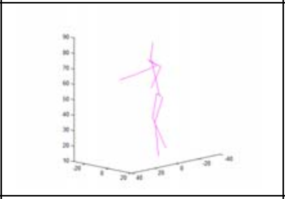
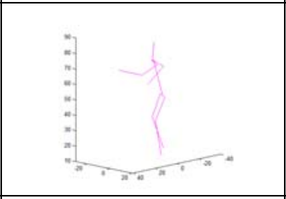
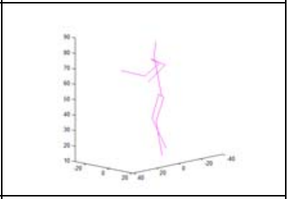
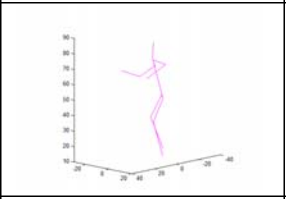
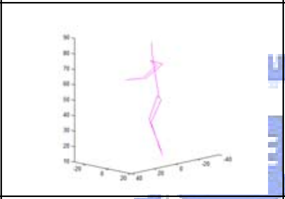
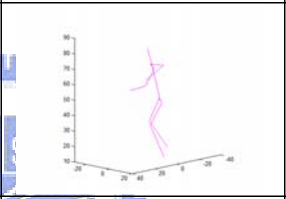
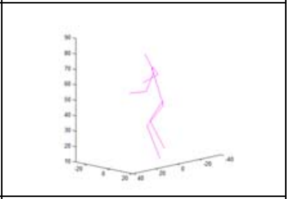
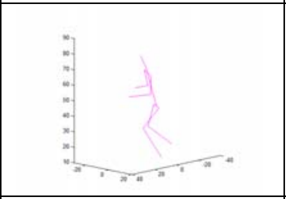
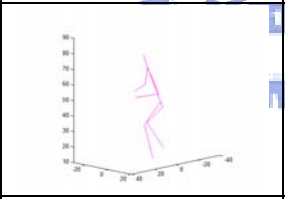
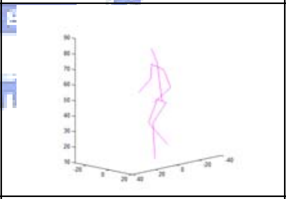
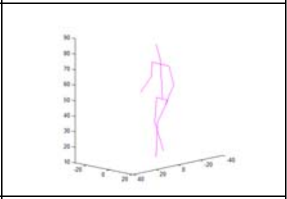
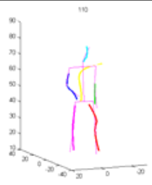
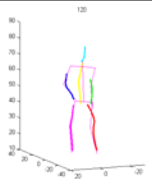
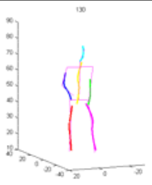
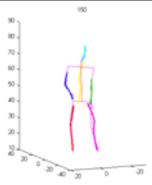
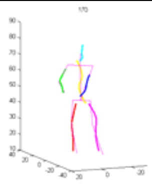
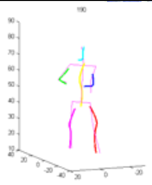
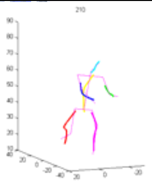
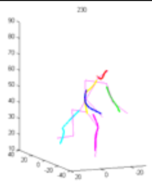
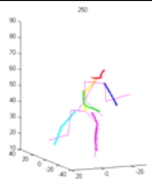
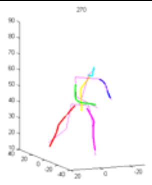
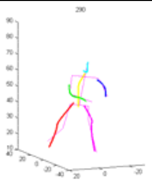
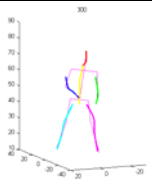
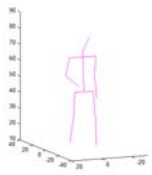
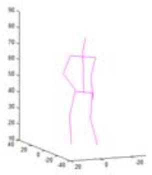
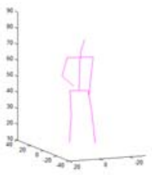
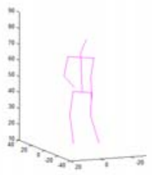
			
time 321	time 351	time 381	time 401
			
time 1	time 41	time 71	time 101
			
time 126	time 156	time 186	time 206
			
time 221	time 251	time 281	time 301
			
time 321	time 351	time 381	time 401

Figure 4-4 The pose estimation results of sequence s3. The first four rows are the input images from one of the eight cameras. The 5th to 8th rows show the skeleton extraction and the model fitting results. The last four rows are the results of pose estimation

D. s4: MeleeAttack01

The target person is struggling in this sequence. The pose estimation results are shown below:

time 110	time 120	time 130	time 150
time 170	time 190	time 210	time 230
time 250	time 270	time 290	time 300
			
time 110	time 120	time 130	time 150
			
time 170	time 190	time 210	time 230
			
time 250	time 270	time 290	time 300
			
time 110	time 120	time 130	time 150

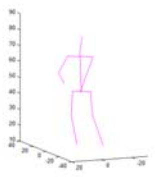
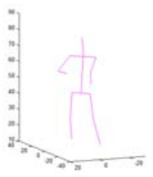
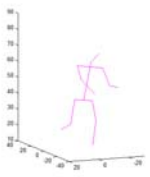
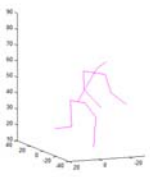
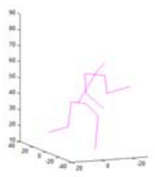
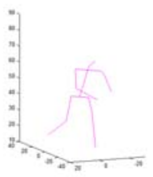
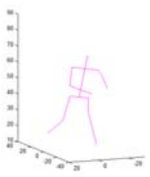
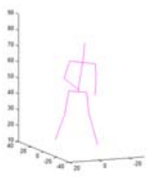
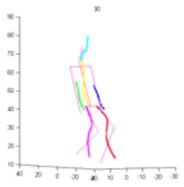
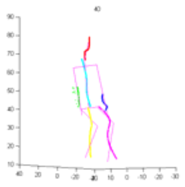
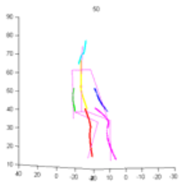
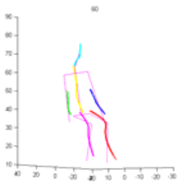
			
time 170	time 190	time 210	time 230
			
time 250	time 270	time 290	time 300

Figure 4-5 The pose estimation results of sequence s4. The first three rows are the input images from one of the eight cameras. The extracted skeleton and its model fitting results are shown in the 4th to 6th rows. The last three rows show the pose estimation results.

E. s5: sitcouch1

The performer is going to sit on a couch in this sequence. We show below the pose estimation of this sequence.

time 30	time 40	time 50	time 60
time 80	time 100	time 110	time 130
time 150	time 170	time 190	time 200
			
time 30	time 40	time 50	time 60

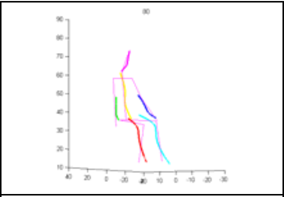
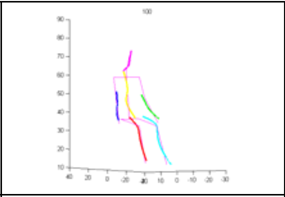
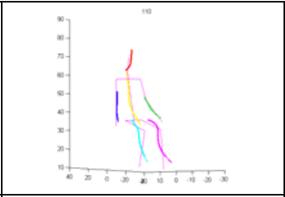
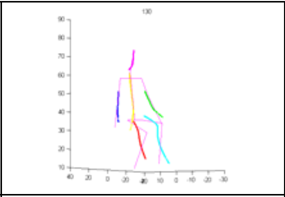
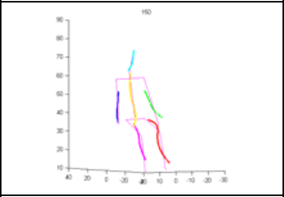
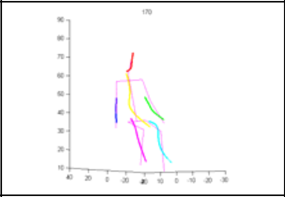
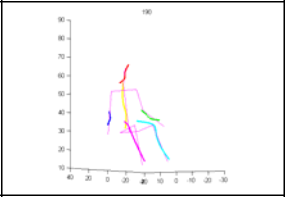
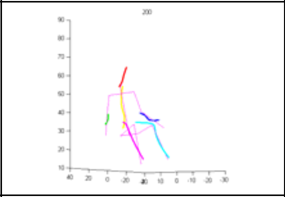
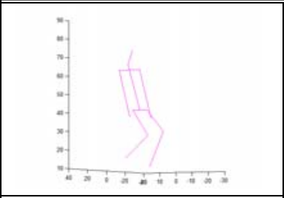
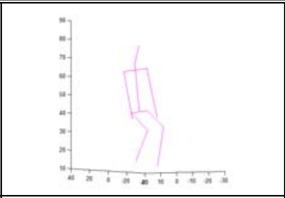
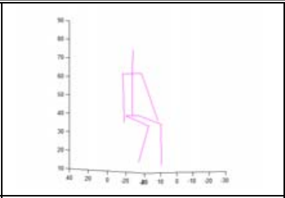
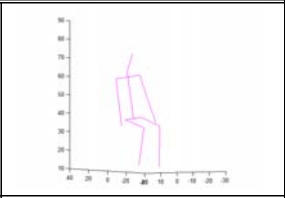
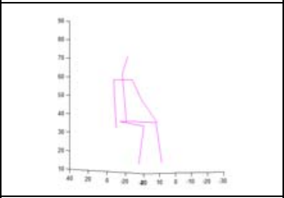
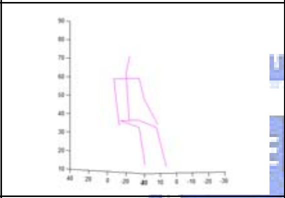
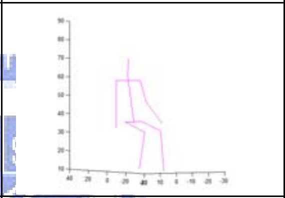
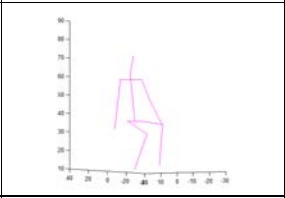
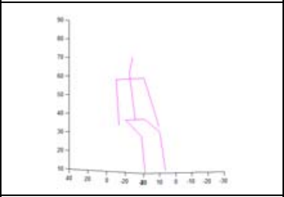
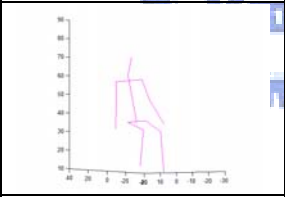
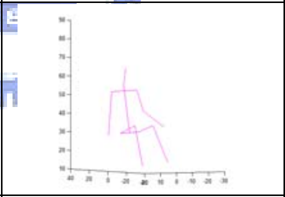
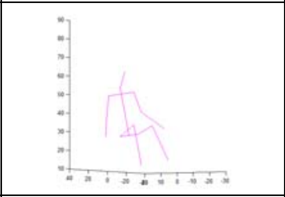
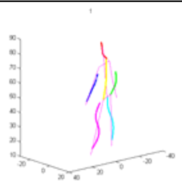
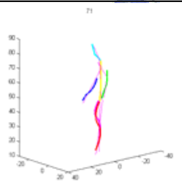
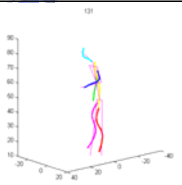
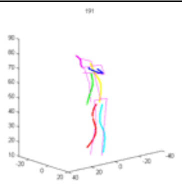
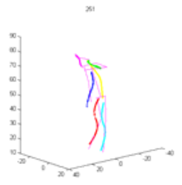
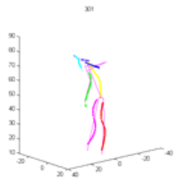
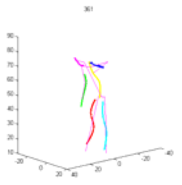
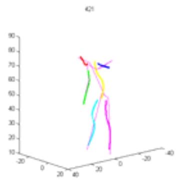
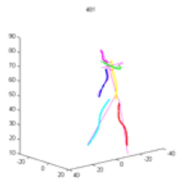
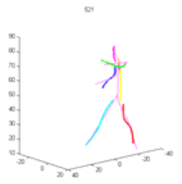
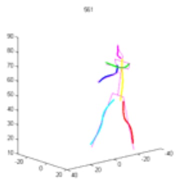
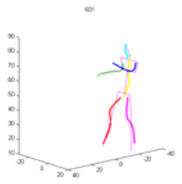
			
time 80	time 100	time 110	time 130
			
time 150	time 170	time 190	time 200
			
time 30	time 40	time 50	time 60
			
time 80	time 100	time 110	time 130
			
time 150	time 170	time 190	time 200

Figure 4-6 The pose estimation results of sequence s5. The 1st to 3rd rows show the input images from one of the eight cameras at 12 different time instants. The 4th to 6th rows are the extracted skeleton and the fitting results. The last three rows show the pose estimation results.

F. s6: d1_t03_Tenements_Look_Out_Door_Close

In this sequence, the target person looks out the door and then closes the door. The pose estimation results are shown below:

time 1	time 71	time 131	time 191
time 251	time 301	time 361	time 421
time 481	time 521	time561	time 601
time 621	time 681	time 741	time 781
			
time 1	time 71	time 131	time 191
			
time 251	time 301	time 361	time 421
			
time 481	time 521	time561	time 601

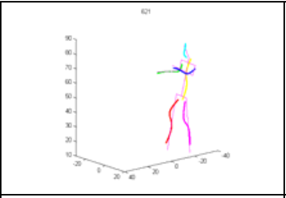
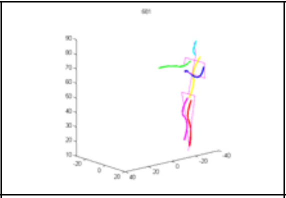
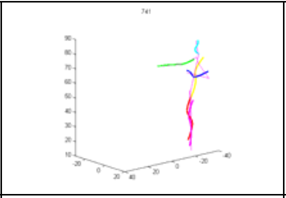
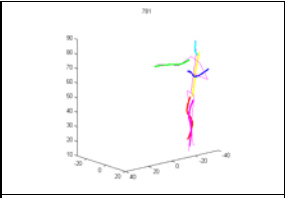
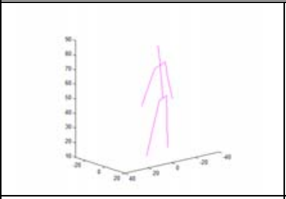
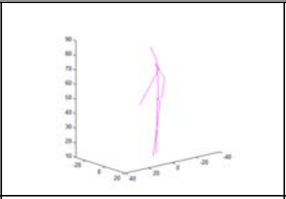
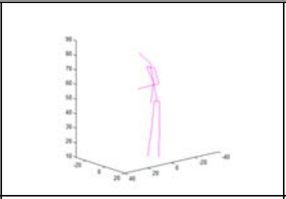
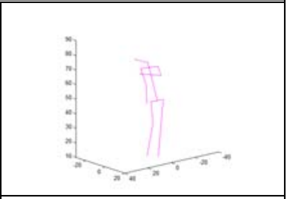
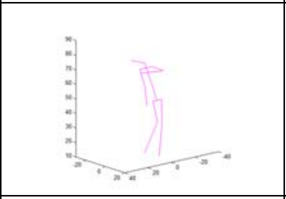
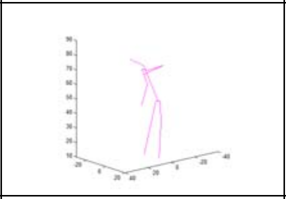
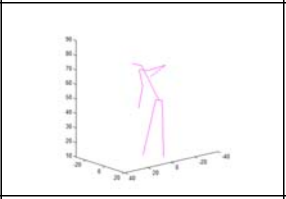
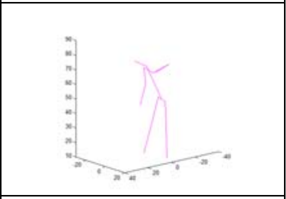
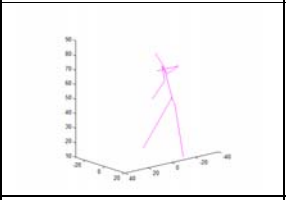
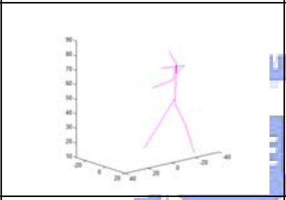
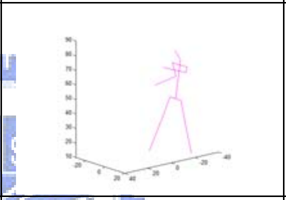
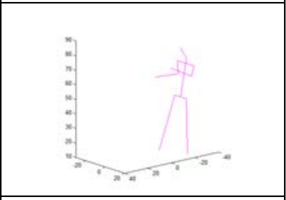
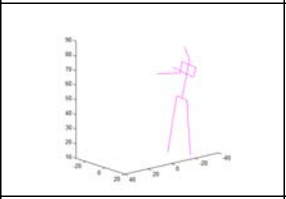
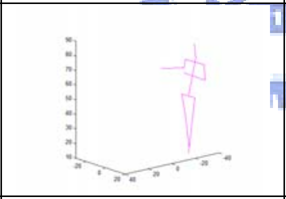
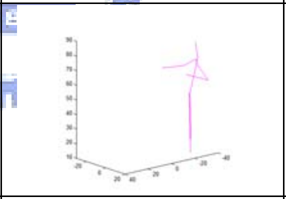
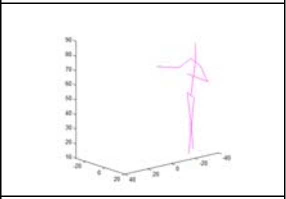
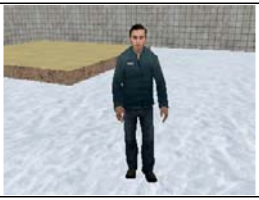
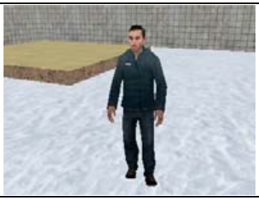
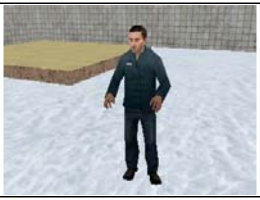
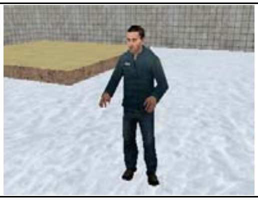
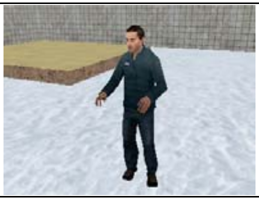
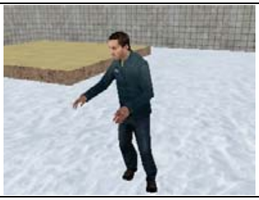
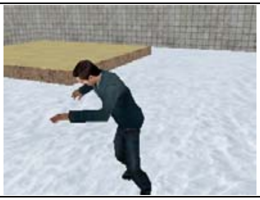
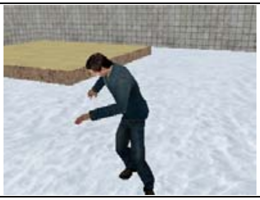
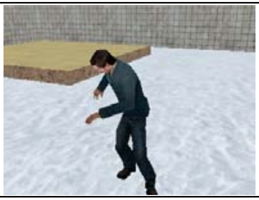
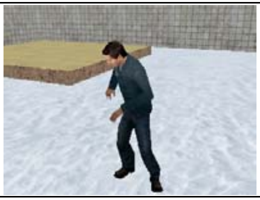
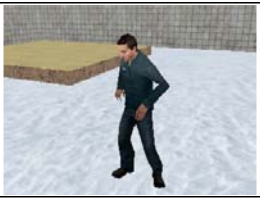
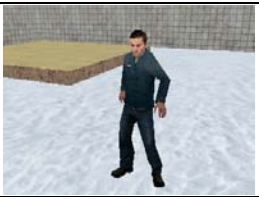
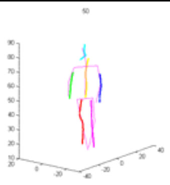
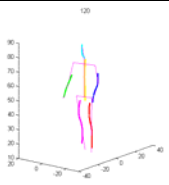
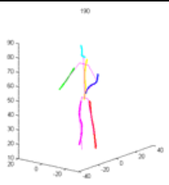
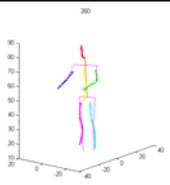
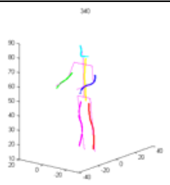
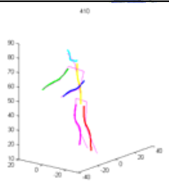
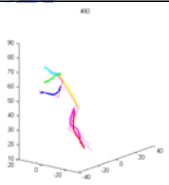
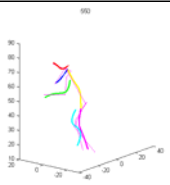
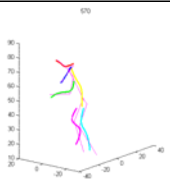
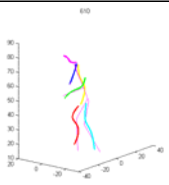
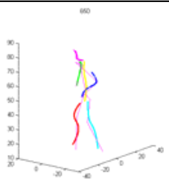
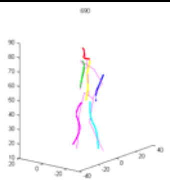
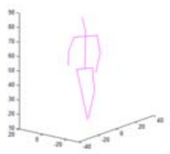
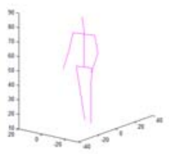
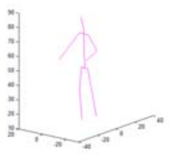
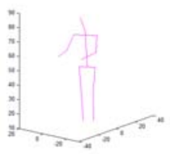
			
time 621	time 681	time 741	time 781
			
time 1	time 71	time 131	time 191
			
time 251	time 301	time 361	time 421
			
time 481	time 521	time561	time 601
			
time 621	time 681	time 741	time 781

Figure 4-7 The pose estimation results of sequence s6. Input images from one of the eight cameras are shown in the first four rows. 5th to 8th rows are the results of the extracted skeleton and model-fitting. The last four rows show the results of pose estimation.

G. s7: luggagepush

The target person is pushing a piece of luggage in this sequence. The pose estimation results are shown below:

			
time 50	time 120	time 190	time 260
			
time 340	time 410	time 480	time 550
			
time 570	time 610	time 650	time 690
			
time 50	time 120	time 190	time 260
			
time 340	time 410	time 480	time 550
			
time 570	time 610	time 650	time 690
			
time 50	time 120	time 190	time 260

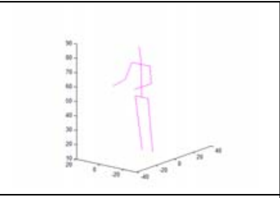
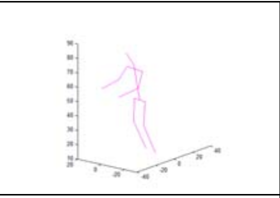
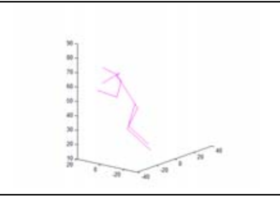
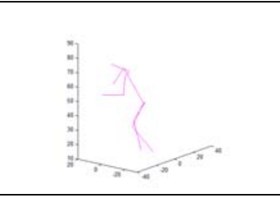
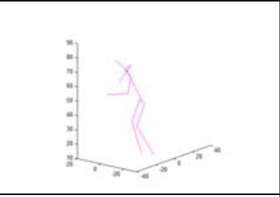
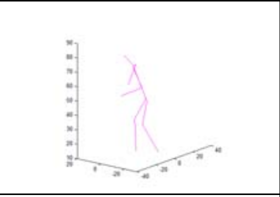
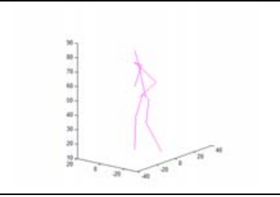
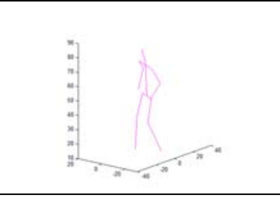
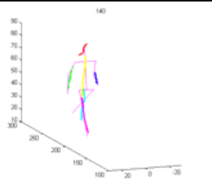
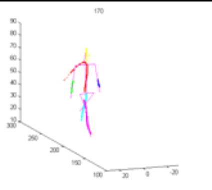
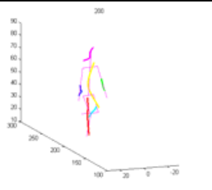
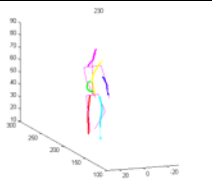
			
time 340	time 410	time 480	time 550
			
time 570	time 610	time 650	time 690

Figure 4-8 The pose estimation results of s7. The first three rows are the input images from one of the eight cameras. The 4th to 6th rows show the extracted skeleton and model-fitting results. The pose estimation results are shown in the last three rows.

H. s8: walk_all

The target person takes a walk in this sequence. It's a challenging sequence since the hands get partially occluded. It can be seen that our system can still generate robust results for this sequence.

time 140	time 170	time 200	time 230
time 260	time 320	time 350	time 380
time 400	time 415	time 430	time 445
			

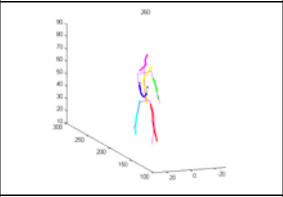
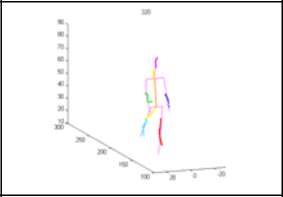
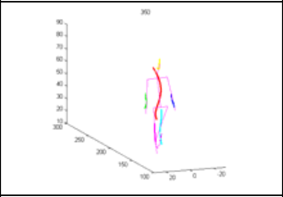
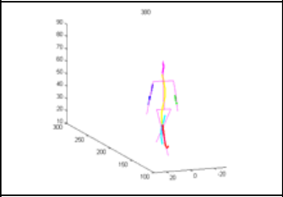
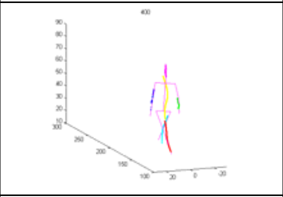
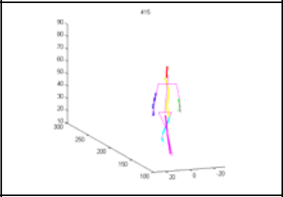
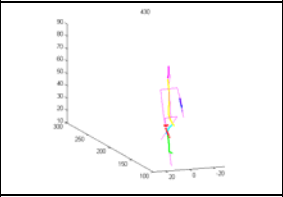
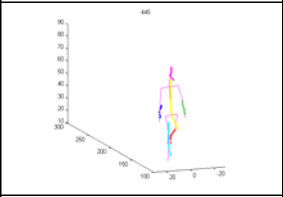
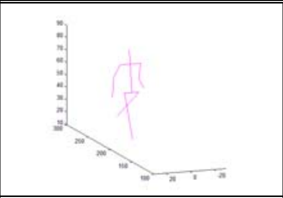
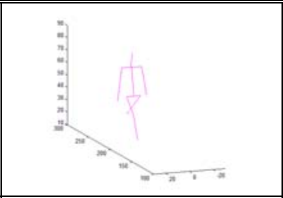
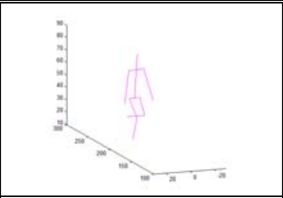
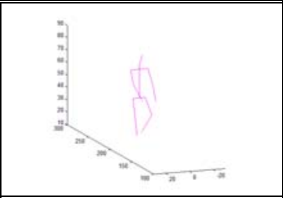
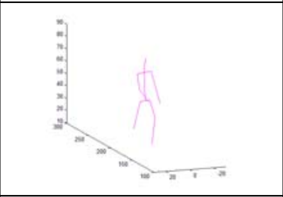
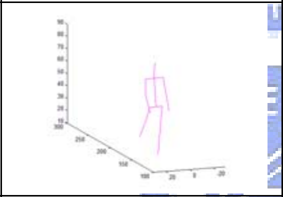
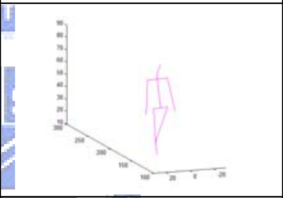
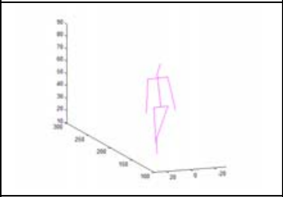
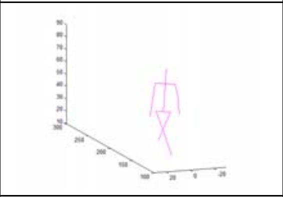
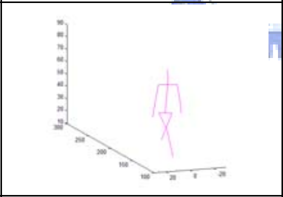
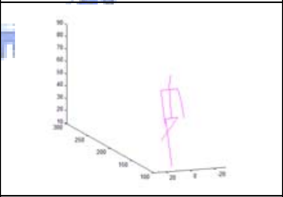
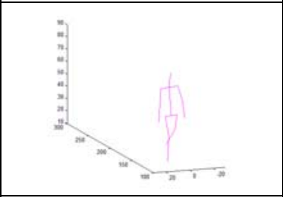
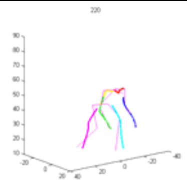
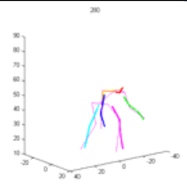
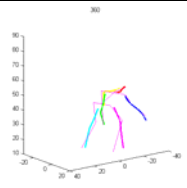
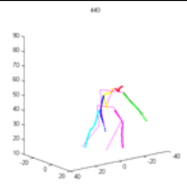
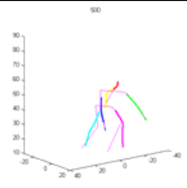
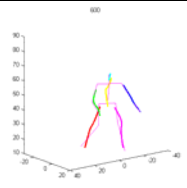
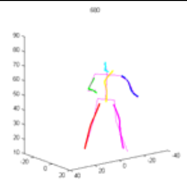
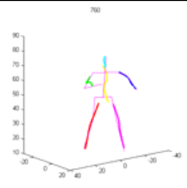
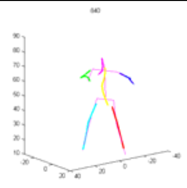
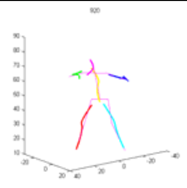
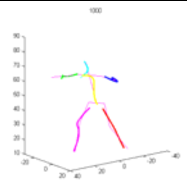
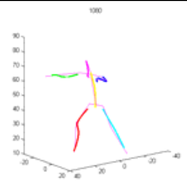
time 140	time 170	time 200	time 230
			
time 260	time 320	time 350	time 380
			
time 400	time 415	time 430	time 445
			
time 140	time 170	time 200	time 230
			
time 260	time 320	time 350	time 380
			
time 400	time 415	time 430	time 445

Figure 4-9 The pose estimation results of sequence s8. The 1st to 3rd rows show the input images from one of the eight cameras. The 4th to 6th rows are the extracted skeleton and model-fitting results. The pose estimation results are shown in the last three rows.

I. s9: d3_c17_03_throw_from_tower

In this sequence, the target person lifts something from the ground and then drops it. This sequence is also challenging since in the initial pose the person bends his back and hangs his arms (*Figure 4-10*, time 220). This pose is quite different from a standing posture. It can be seen that our system can still deal with this sequence well.

time 220	time 280	time 360	time 440
time 500	time 600	time 680	time 760
time 840	time 920	time 1000	time 1080
time 1160	time 1200	time 1240	time 1280
			
time 220	time 280	time 360	time 440
			
time 500	time 600	time 680	time 760
			
time 840	time 920	time 1000	time 1080

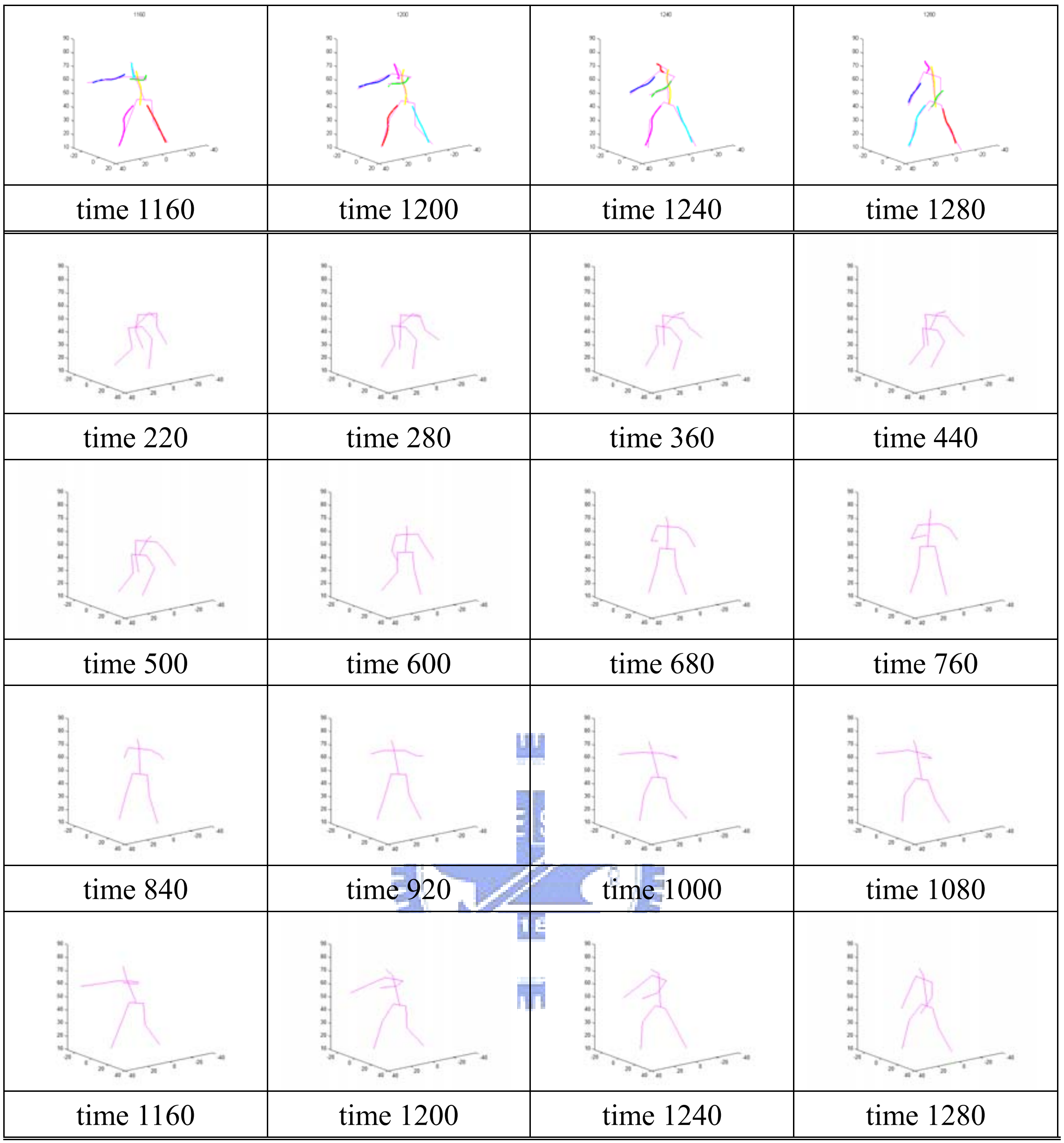


Figure 4-10 The pose estimation results of sequence s9. The first four rows are the input images from one of the eight cameras. The next four rows show the extracted skeleton and the fitting results. The last four rows are the pose estimation results.

Chapter 5. Conclusions

In this thesis, we proposed a model-based pose estimation technique for multiple camera motion capture system. The whole process, which includes initialization and pose estimation, is automatic and markerless. For system initialization, we reconstruct the 3-D visual hull from multiple foreground silhouettes. We segment the human body in the eigenspace, and then extract the skeletons to reduce the dimension of the feature space. In the initialization stage, no prior model is needed. Furthermore, we modify the Laplacian Eigenmap to make the body parts segmentation easier than Sundaresan's method [6]. After system initialization, a prior 3-D human model is fitted to the extracted skeleton based on the PSO algorithm. Our human model allows self-spin and combines motion constraints with the pose estimation in a more natural way. Moreover, the temporal smoothing process is also achieved by a parameter propagation mechanism. The experiment results show that our system can handle the pose estimation problem for various kinds of actions.

Reference

[1] http://www.math.umn.edu/~wittman/mani/

[2] http://mathworld.wolfram.com/Manifold.html

[3] A. Agarwal and B. Triggs, "3-D Human Pose from Silhouettes by Relevance Vector Regression," IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 882-888, June, 2004.

[4] A. Elgammal, and C. S. Lee, "Inferring 3-D Body Pose from Silhouettes Using Activity Manifold Learning, " IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 681-688, June, 2004.

[5] A. Laurentini, "How Many 2-D Silhouettes Does It Take to Reconstruct a 3-D Object?" Computer Vision and Image Understanding, vol. 67, no.1, pp. 81-87, 1997.

[6] A. Sundaresan, and R. Chellappa, "Model Driven Segmentation of Articulating Humans in Laplacian Eigenspace," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.

[7] C. Menier, E. Boyer, and B. Raffin,"3-D Skeleton-based Body Pose Recovery," International Symposium on 3-D Data Processing, Visualization and Transmission, pp. 389-396, 2006.

[8] C. Robertson and E. Trucco, "Human Body Posture via Hierarchical Evolutionary Optimization," British Machine Vision Conference, vol. 3, pp. 999-1008, 2006.

[9] C. W. Chu, O. C. Jenkins, and M. J. Mararić, "Markerless Kinematic Model and Motion Capture from Volume Sequences," IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 475-482, June, 2003.

[10] F. Cuzzolin, D. Mateus, E. Boyer, and R. Horaud, "Robust Spectral 3-D-Bodypart Segmentation along Time," International Conferences on Computer Vision 2nd Workshop on Human Motion, pp. 196-211, 2007.

[11] G. R. Taylor, A. J. Chosak, and P. C. Brewer, "OVVV: Using Virtual Worlds to Design and Evaluate Surveillance Systems, "IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, June 2007.

[12] I. H. Chen and S J. Wang, "Efficient Vision-Based Calibration for Visual Surveillance Systems with Multiple PTZ Cameras", in Proceeding of IEEE International Conference on Computer Vision Systems, Jan. 5-7, 2006.

[13] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman, "Human Body Model Acquisition and Tracking Using Voxel Data," International Journal of Computer Vision, vol.53, pp. 199-223, 2003.

[14] J. B. Tenenbaum, V. D. Silva, and J. C. Langford, “A Global Geometric Framework for Nonlinear Dimensionality Reduction,” Science, vol. 290, pp. 2319-2323, Dec., 2000.

[15] J. Deutscher, A. Blake, and I. Reid, “Articulated Body Motion Capture by Annealed Particle Filtering,” Computer Vision and Image Understanding, vol. 2, pp. 126-133, June, 2000.

[16] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” Proceedings of the IEEE International Conference on Neural Networks, vol. 4, pp. 1942-1948, 1995.

[17] L. Mündermann, S. Corazza, A. M. Chaudhari, E. J. Alexander, T. P. Andriacchi, “Most Favorable Camera Configuration for a Shape-from-Silhouette Markerless Motion Capture System for Biomechanical Analysis,” Proceedings of Society of Photo-Optical Instrumentation Engineers, pp. 278-287, Jan., 2005.

[18] M. Belkin and P. Niyogi, “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation,” Neural Comput., pp. 1373-1396, 2003.

[19] M. Piccardi and T. Jan, “Mean-shift background image modeling,” International Conference on Image Processing, vol. 5, pp. 3399–3402, Oct. 2004

[20] N. R. Howe, “Silhouette Lookup for Automatic Pose Tracking,” Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops, pp. 15-22, June, 2004.

[21] R. Kehl, L. V. Gool, “Markerless Tracking of Complex Human Motions from Multiple Views,” Computer Vision and Image Understanding, vol. 104, pp. 190-209, 2006.

[22] R. Kehl, M. Bray, and L.V. Gool, “Full Body Tracking from Multiple Views Using Stochastic Sampling,” IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 129-136, June, 2005.

[23] R. Murray, Z. Li, S. Sastry, “A Mathematical Introduction to Robotic Manipulation,” CRC Press, 1993.

[24] R. Szeliski, “Rapid Octree Construction from Image Sequences, ” CVGIP: Image Understanding, vol. 58, pp. 23-32, 1993.

[25] S. T. Roweis and L. K. Saul, “Nonlinear Dimensionality Reduction by Locally Linear Embedding,” Science, vol. 290, pp. 2323-2326, Dec., 2000.

[26] T. B. Moeslund, A. Hilton, and V. Krüger, “A Survey of Advances in Vision-based Human Motion Capture and Analysis,” Computer Vision and Image Understanding, vol. 104, pp. 90-126, 2006.

[27] Y. Sagawa, M. Shimosaka, T. Mori, and T. Sato, “Fast Online Human Pose Estimation via 3-D Voxel Data,” Proceedings of IEEE Conference on Intelligent Robots and Systems, pp. 1034-1040, 2007.