TRACKING HUMAN BODY MOTION BY USING DEFORMABLE MESH

デフォーマブルメッシュを用いた人体運動のトラッキング

by

LI Xiaolu
李 暁璐

A Master Thesis
修士論文

Submitted to
the Graduate School of Information Science and Technology
the University of Tokyo
on February 3, 2006
in Partial Fulfillment of the Requirements
for the Degree of Master of Information Science and Technology
in Information and Communication Engineering

Thesis Supervisor: Professor Katsushi Ikeuchi　池内 克史　教授

ABSTRACT

Our laboratory is promoting a project for digitally archiving tangible and intangible cultural heritage objects, and traditional Japanese dance is one of the subjects we are working on currently.

To archive traditional Japanese dances, we first acquire the dance motion using an optical motion capture system, where a human dancer needs to put on markers all over his or her body, and the geometry-and-appearance is taken separately. The main drawback of this system is that the motion and the geometry-and-appearance cannot be obtained simultaneously. To overcome this problem, we have developed a system that can obtain both information simultaneously.

The proposed system includes two main stages: 3D shape reconstruction and human motion tracking.

A sequence of 3D volumetric representations of a human body is reconstructed from multiple video streams, and the reconstructed data are further refined with the voxel coloring method. The texture images or appearance, are also acquired at the same time.

To estimate the motion of the human body, we proposed to utilize a deformable human body model that is composed of a kinematic skeleton model and a surface skin model. For each patch of the surface, the system finds the correspondence in the observed data and estimates the pose and the joint angles of the body iteratively. The surface deforms naturally by following the motion of the bones, thus the correspondences are robustly found, especially around the joints.

The proposed techniques are verified in simulation as well as in a real world setting and are proved to track a variety of whole body motions reliably. By combining the textured 3D data and the motion obtained from tracking, the motion and the geometry-and-appearance are archived successfully.

Acknowledgements

First, I would like to express my sincere gratitude to my current thesis advisor, Prof. Ikeuchi Katsushi, for his generous support and guidance, through my Master's course at the University of Tokyo.

I am very grateful to Dr. Kouichi Ogawara, for giving me many ideas in this thesis and much useful advice for this study.

I am very grateful to members of Robotics Group, for allowing me to share an enjoyable space and good time with them.

I am very grateful to all secretaries of our laboratory, for their kindness and help with my life.

I am also very grateful to all the members of the Ikeuchi Laboratory, for helping me much in various ways. The environment here is so stimulating and challenging; to have these two years with these people will be a treasure for my whole life.

I am grateful to all my friends outside the lab, for their constant help and encouragement during my good days and bad.

Lastly, I would like to thank my family for their support and patience during these many years and, I hope, in the years to come.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Tracking the human body, also called motion capture or posture estimation, is a problem of estimating the parameters of the human body model (such as joint angles) as the position and configuration of the tracked body change over time.

A reliable motion capture system would be valuable in many applications. One class of applications is those where the extracted body model parameters are used directly, for example, to interact with a virtual world, drive an animated avatar in a video game, or for computer graphics character animation.

Another class of applications uses extracted parameters to classify and recognize people, gestures, or motions, such as surveillance systems, intelligent environments, or advanced user interfaces (sign language translation, gesture-driven control, gait, or pose recognition). Finally, the motion parameters can be used for motion analysis in applications such as personalized sports training, choreography, or clinical studies of orthopedic patients.

The goal of this study of human body tracking is twofold:

- To preserve the traditional Japanese dance as a part of archiving our intangible cultural heritage archiving
- To contribute to the preservation of traditional dances by developing a dancing humanoid robot that can imitate human dances finally

This study is confined to the first goal and includes the following topics in the field of computer vision: 3D shape reconstruction, and motion tracking shown as areas 2 and 3 in Fig. 1.1.

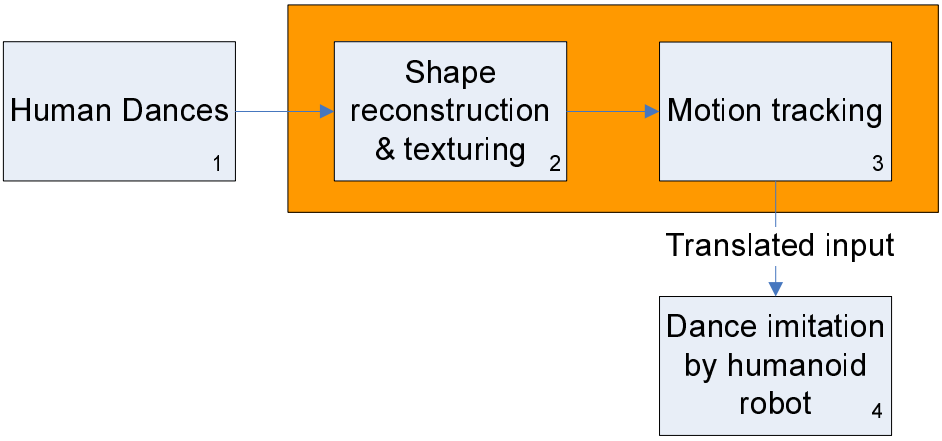


Figure 1.1: The target of the study

In this chapter, we discuss the background and purpose of this study, and related works. We also present a brief overview of our system.

1.1 Background

Cultural heritage is transmitted from generation to generation. It provides people with a sense of identity and continuity, and develops cultural diversity and human creativity. Many facets of cultural heritage in the world contribute to our knowledge of human culture and life in past times. However, due to their age, environment, or political reasons, many of them are in danger of being destroyed. In particular, many important intangible cultural properties such as traditional dances throughout the world are losing lost because of the lack of the successors. Therefore, actions, such as recovering technologies, are being taken to protect these properties.

Recently, there has been an attempt to preserve these properties by the use of computer vision technology. Digital archiving technology is one of the effective ways to record these properties forever and to possibly recreate them in the future. There are arguments that we could just use video systems to archive this goal. However, video data often lack the information needed for a full appreciation of the subject.

Our digital archiving projects of cultural properties including tangible [11] and intangible ones such as folk dances was initiated and expanded to include imitation by a humanoid robot. [19] [20].

One typical method for digitizing human motions is to use a commercial motion capturing system. However, such a motion capturing system provides only low-level movement information, with limits to the clothes and space. Further more, texture information, such as the texture of a kimono in the case of folk dances, cannot be observed and acquired at the same time.

With these considerations in mind, we track the human body using multi-view cameras instead. We assume that the body model of the tracked person is known and placed close to the true position in the beginning of the tracking process. Then we estimate the model parameters in time to reflect the motion of the person.

The purpose of our research is to develop a technique that can acquire the information about geometry-and-appearance and the human body motion simultaneously. Two main procedures undertaken in this study are to obtain the human body shape data with color and to track motions.

1.2 Related Works

Human motion tracking systems can be classified as ”with-markers” and ”without-markers”. Of with-markers systems, there are mechanical, electromagnetic, or optical systems.

These systems typically require the person to wear special markers, body suits or gloves. In an optical system, the person wears optical markers on the body to whose 3D locations a kinematics skeleton is fitted. As human skeleton is a highly articulated structure, and twists and rotations make the movement fully three-dimensional. But because of the occlusion problem, it leads to inconsistent and unreliable tracking of the human body.

The disadvantages of marker-based tracking systems include:

- Identification of standard bony landmarks can be unreliable.
- The soft tissue overlying bony landmarks can move, giving rise to noisy data.
- The marker itself can wobble due to its own inertia.

In the past few years, marker-less systems using only cameras have received much more attention.

Algorithms have been developed that take input from one or multiple cameras working in 2D image plane. This approach can be catalogued with or without explicit shape

models. However, the cases of tracking without models cannot estimate joint angles, so that topic is omitted here.

Methods with models rely on the pre-designed models or require a significant amount of a priori knowledge to generate the model. Assembled simple shape primitives, such as cylinders or ellipsoids, are mostly employed. A common trend has been to use novel 3-D models, which has led to more robust results.

Rehg and Kanade [24] have developed a system for tracking a 3D articulated model of a hand based on a layered template representation of self-occlusions.

Kakadiaris and Metaxas [13] [14] have developed a system for 3D human body model acquisition and tracking using three cameras placed in a mutually orthogonal configuration. The person under observation is requested to perform a set of movements according to a protocol that incrementally reveals the structure of the human body. Once the model has been acquired, the tracking is performed using the physics-based framework. Based on the expected body position, the difference between the predicted and actual images is used to calculate forces that are applied to the model. The dynamics are modeled using the extended Kalman filter. The tracking result, a new body pose, is a result of the applied forces acting on the physics-based model. The problem of occlusions is solved by choosing from the available cameras those that provide visibility of the part and observability of its motion, for every body part at every frame.

Gavrila and Davis [7] used a human body model with 17 DOF and composed of tapered super-quadrics to track (multiple) people in 3D. They use a constant acceleration kinematics model to predict positions of body parts in the next frame. Their locations are then adjusted using the undirected normalized chamfer distance between image contours and contours of the projected model (in multiple images). The search is decomposed in stages: they first adjust positions of the head and the torso, then arms and legs.

Deutscher et al. [5] developed a system using Kalman filter to handle high dimensional configuration space of human motion capture. It uses a continuation principle, based on annealing, to gradually introduce the influence of narrow peaks in the fitness function. Two image features are used in combination: edges and foreground silhouettes. Good tracking results are achieved using this approach.

Delamarre and Faugeras [4] describe an algorithm that computes human body contours based on optical flow and intensity. Then, forces are applied that attempt to align the outline of the model to the contours extracted from the data. This procedure is repeated until a good agreement is achieved.

Hunter et al. [10] developed an algorithm based on the Expectation Maximization (EM) procedure that assigns foreground pixels to body parts and then updates body part positions to explain the data. An extra processing step, based on virtual work static equilibrium conditions, integrates object kinematic structure into the EM procedure, guaranteeing that the resulting posture is kinematically valid.

Wren et al. [30] introduce a statistical description of the whole scene making use of color, driven by 2Dblob features from multiple cameras that are probabilistically integrated into a 3D human body model. Also, the system includes a feedback from the 3D body model to the 2D feature tracking by setting the appropriate prior probabilities using the extended Kalman filter.

However, in the algorithms that work with data in 2-D image planes, the 3D body model is repeatedly projected onto the image planes to be compared against the extracted image features. Another problem in working with the image plane data is that different body parts appear in different sizes and may be occluded depending on the relative position of the body to the camera and on the body pose.

Recently, sequences of shape-from-silhouette models have been considered as input data for human motion estimation. Ellipsoidal body models, kinematic skeletons, or a pre-defined kinematics model with triangular mesh surface representation are adopted to be fitted to the volumetric data.

The first attempt at using voxel data obtained from multiple cameras to estimate body pose was reported by Cheung et al. [3]. A simple six-part body model is fitted to the 3D voxel reconstruction. The tracking is performed by assigning the voxels in the new frame to the closest body part from the previous frame and by recomputing the new position of the body part based on the voxels assigned to it. This simple approach did not guarantee that two adjacent body parts would not drift apart. Also it could lose track easily for moderately fast motions.

Cheung et al. [2] improved it to use colored surface points (CSPs) to segment the volumetric body data into rigidly moving body parts, based on the results of the previous frames, and took advantage of the constraint of equal motion of parts at their coupling joints to estimate joints positions of an actor.

R. Kehl et al. [15] presented an approach for full body pose tracking using stochastic sampling. They adopted a deformable body model, using volumetric reconstruction data to fit the model. The color information is needed in their system to improve the speed and robustness of the tracking.

1.3 System Overview

In this study, we present a marker-less human body tracking system. The whole system process is shown in Fig. 1.2.

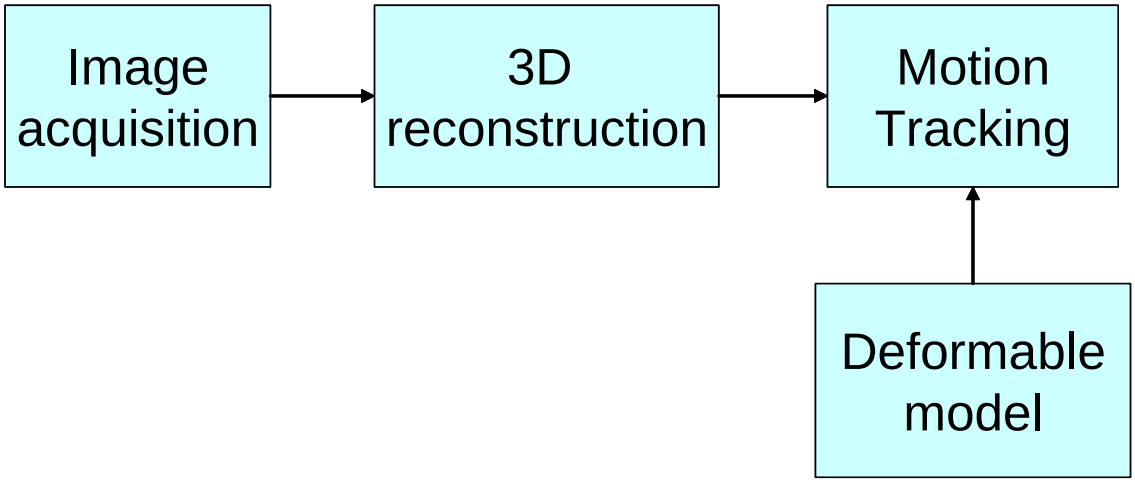


Figure 1.2: The system flowchart

First, human body motion is acquired from multiple synchronized video cameras.

Instead of the approach of working directly with the image data, we use 3D shape data of the human body at each frame as input to the model tracking.

This is carried out with a 3D shape reconstruction process. Silhouette images are acquired after background segmentation process of the input images; a human body shape is reconstructed using a visual hull method. Furthermore, taking advantage of the color information of the voxel data, a voxel coloring method is adopted to obtain a relatively precise reconstruction data set, which leads to a more robust result of the tracking stage. At the same time, we obtain texture information of a relatively precise 3D human body reconstruction shape with tracking.

Then we generate an articulated human body model for tracking. We adopt a two-layered articulated human body model with kinematic skeletal structure inside and deformable surface mesh outside, which makes the model act naturally and leads to more robust tracking results than the general segment model.

At the final stage, we fit the human body model to the reconstructed data, to estimate the pose and the joint angles of a human body.

In the following chapters, details of each process are described.

The remainder of this thesis is organized as follows.

In Chapter 2, related works about 3D shape reconstruction and motion tracking are introduced.

In Chapter 3, algorithm for computing 3D shape reconstructions is presented.

Chapter 4, focuses on the human body model and describes our tracking method.

In Chapter 5, experiments are shown and results are examined.

Finally, in Chapter 6, summary and future works are presented.

Chapter 2

3D Shape Reconstruction

In computer vision, a persistent problem is the creation of a 3D model of a shape from the given 2D images of the shape. Applications include robot navigation, virtual reality, games, and special effects for motion pictures. To analyze and archive folk dances, we use 3D shape reconstruction to reconstruct human body shape from static 2D images and then track human body motion from time sequences using these reconstructed shape data.

One of the traditional shape reconstruction methods called multi-view stereo is based on image matching [18] [32], using either intensity-based methods or feature-based methods. The method computes correspondences across images and then recovers 3D structure by triangulation and surface fitting. Some of the disadvantages of this approach are that (1) views must often be close together (i.e., small baseline) to make correspondence techniques effective; (2) correspondences must be maintained over many views spanning large changes in viewpoint; (3) A parameterized surface model is needed to fit to the 3D points in order to obtain the final dense surface reconstruction for sparse features; and (4) There are no explicit solutions in handling of occlusion differences between views.

Shape from silhouette (SFS) is an alternative approach of 3D shape reconstruction. Based on computations in three-dimensional shape space, it uses silhouettes from multiple input images to construct the volume that is consistent in such images. This allows widely-separated views, avoids the disadvantages of stereo methods but require calibrated input images.

Because of the above advantages of SFS, and the fact that a relatively large space is needed for our target to track the human body motion, we choose the second method to compute the shape reconstruction using multiple cameras.

There are many advantages to estimating 3D shape using SFS, such as the fact that silhouettes are readily and easily obtainable and the implementation is generally straightforward. On the other hand, the volume only approximates the true 3D shape depending on the number of views. In particular, dependence only on silhouette images fails to capture the concave patches that are not observable in any silhouette. Thus, we consider using color information to improve the reconstruction results. A color-consistency carving method called voxel coloring [26] is used to output a better 3D shape.

The voxel coloring method, like other shape from silhouette methods, operates in an initialized 3D volume space, which should be the superset of the object to be reconstructed. The selection of the init volume is crucial to save the computation; mostly, a visual hull method is used to do this.

The camera images are first segmented using the algorithm described in [9], which eliminates shadows and produces good quality silhouettes. We will explain it later in Section. 2.2.

We use volumetric representation for the reconstructed 3D shape data, which is the input of the tracking step.

A texture mapping method is also used to acquire the color information of the human body shape.

2.1 Overview

The whole process is shown in Fig. 2.1. There are 5 topics described in this chapter as follows:

- Background subtraction
  As the initial input data, silhouette images are generated by the color images captured from cameras using background subtraction.

- Visual hull reconstruction
  Given the silhouette images retrieved from the former step, the visual hull, which is the maximal solid volume, is reconstructed. The visual hull is produced by the intersection of the input silhouette cones. The resulting intersection volume is a conservative bound on the object's shape.

- Voxel Coloring reconstruction
  The Voxel Coloring method assigns color information to voxels (points) in a 3D

volume, and checks the voxel color to see if it is consistent with all the input color images that can see the voxel. Thus, a more precise reconstructed data can be acquired.

- Mesh simplification
  Mesh representation is used for the reconstructed 3D shape and simplification is to save the computation later in the tracking.

- Texture mapping
  In this step, all original color images captured from cameras are combined together to produce the final textured model.

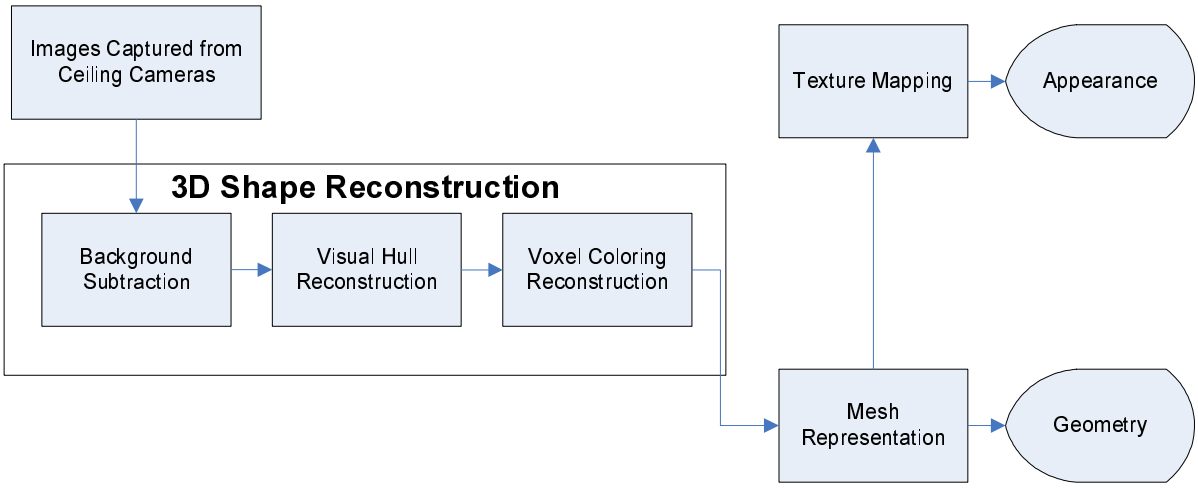


Figure 2.1: Reconstruction Process

The details are described in the following sections.

2.2 Background Subtraction

The basic scheme of background subtraction is to subtract the image from a reference image that models the background scene.

Basically, the basic steps of algorithm are as follows:

- Background modeling, to construct a reference image representing the background

- Threshold selection, to determine the threshold in the subtraction operation
- Subtraction operation

The simplest background model assumes that the brightness of each background pixel varies independently, according to normal distribution. The background characteristics can be calculated by accumulating several dozens of frames, as well as their squares. That means, for a pixel i located in (x, y), finding sum values $\boldsymbol{S}_i$ and a sum of squares of the values $\boldsymbol{Sq}_i$. Then arithmetic mean is calculated as

$$\boldsymbol{m}_i = \frac{\boldsymbol{S}_i}{\boldsymbol{N}} \tag{2.1}$$

where $\boldsymbol{N}$ is the number of the frames collected, and standard deviation as

$$\boldsymbol{\sigma}_i = sqrt(\frac{\boldsymbol{S}_i}{\boldsymbol{N}} - (\frac{\boldsymbol{Sq}_i}{\boldsymbol{N}})^2) \tag{2.2}$$

Then a pixel in a certain frame is regarded as foreground if condition

$$\|m_i - p_i\| \geq \boldsymbol{C}\sigma_i \tag{2.3}$$

is met, where $\boldsymbol{C}$ is a certain constant.

To obtain the background model, any objects should be put away from the camera for a few seconds, so that a whole image from the camera represents subsequent background observation.

However, the above technique cannot get rid of the influence of the shadows [9]. To improve this, we do the following:

First, generate a statistics model for every pixel in the images.

As shown in Fig. 2.2, for a pixel i, let E_i represent the pixel's expected RGB color in the reference image; let I_i denote the pixel's RGB color value in the image to subtract from the background. To measure the distortion of I_i from E_i, two components CD and α are decomposed as:

$$\boldsymbol{CD}_i = \|\boldsymbol{I}_i - \boldsymbol{\alpha}_i\boldsymbol{E}_i\| \tag{2.4}$$

$$\boldsymbol{\alpha}_i = \arg\min(\boldsymbol{I}_i - \boldsymbol{\alpha}_i\boldsymbol{E}_i)^2 \tag{2.5}$$

Analysis of CD and α operates the background subtraction.

To decide the statistics model of pixel i, we capture the reference background image over a number of static background frames as the simplest method mentioned above.

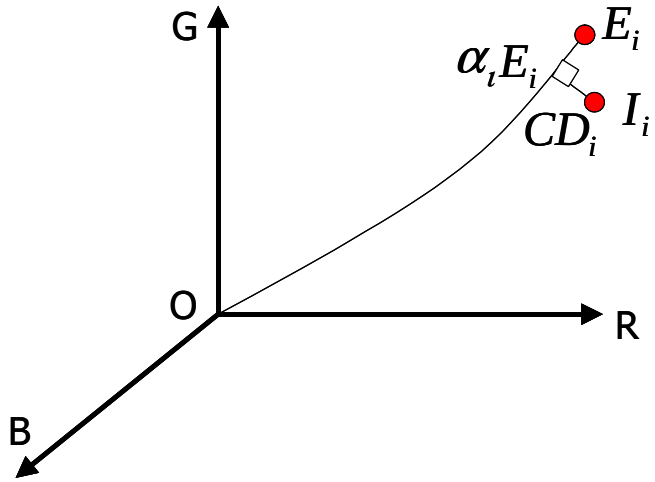


Figure 2.2: Color model in the three-dimensional RGB color space

A pixel is modeled as $\langle \boldsymbol{E}_i, \boldsymbol{s}_i, \boldsymbol{a}_i, \boldsymbol{b}_i \rangle$, where $\boldsymbol{E}_i$ is the expected color value, and $\boldsymbol{s}_i$ is the standard deviation of color value shows as

$$\boldsymbol{E}_i = [\mu_R(i), \mu_G(i), \mu_B(i)] \tag{2.6}$$

$$\boldsymbol{s}_i = [\sigma_R(i), \sigma_G(i), \sigma_B(i)] \tag{2.7}$$

where $\mu_(i)$, $\mu_R(i)$, $\mu_R(i)$ are the arithmetic means, and $\sigma_R(i)$, $\sigma_R(i)$, $\sigma_R(i)$ are the standard deviation of pixel i over N frame of the background frames, calculated as 2.1 and 2.2.

$\boldsymbol{a}_i$ is the variation of the brightness distortion and $\boldsymbol{b}_i$ is the variation of the birghtness distortion of pixel i, given by

$$\boldsymbol{a}_i = RMS(\boldsymbol{a}_{in}) = \sqrt{\frac{\Sigma_{n=1}^{N}(\boldsymbol{\alpha}_{in} - 1)^2}{\boldsymbol{N}}} \tag{2.8}$$

$$\boldsymbol{b}_i = RMS(\boldsymbol{CD}_{in}) = \sqrt{\frac{\Sigma_{n=1}^{N}(\boldsymbol{CD}_{in})^2}{\boldsymbol{N}}} \tag{2.9}$$

As mentioned above, the different pixels yield different distributions of $\boldsymbol{\alpha_i}$ and $\boldsymbol{CD_i}$, to normalize $\boldsymbol{\alpha_i}$ and $\boldsymbol{CD_i}$ as

$$\hat{\alpha}_i = \frac{\alpha_i - 1}{a_i} \tag{2.10}$$

$$\hat{CD}_i = \frac{CD_i}{b_i} \tag{2.11}$$

is necessary.

Based on these definitions, a pixel is classified as

$$\begin{cases} F : \hat{CD}_i \geq \tau_{CD}, & else \\ B : \hat{\alpha}_i \leq \tau_{\alpha 1} & and \quad \hat{\alpha}_i \geq \tau_{\alpha 2}, \quad else \\ S : \hat{\alpha}_i \leq 0 & else \\ H : otherwise \end{cases} \tag{2.12}$$

where τ_{CD}, $\tau_{\alpha 1}$ and $\tau_{\alpha 2}$ are selected threshold values used to determine the similarities of the chromaticity and brightness between the background image and the current image, which can be selected by a statistical learning procedure automatically at the detection rate of r.

Fig. 2.3 shows the segmentation result for a frame captured from four different views.

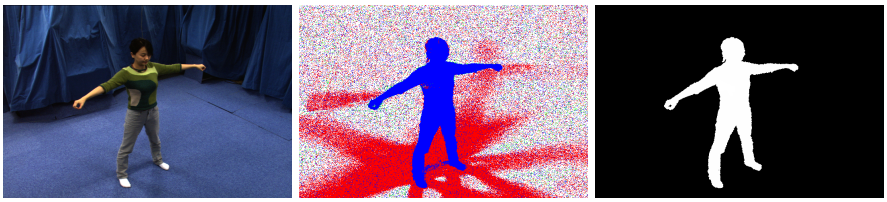


Figure 2.3: The result of background subtraction

2.3 Visual Hull Reconstruction

A silhouette image is a binary image, with the value at a point indicating whether or not the visual ray from the optical center through that image point intersects an object surface in the shape. Thus each pixel is either a silhouette point or a background point. The binary images can be obtained by segmentation algorithms like background subtraction. Combined with calibration information for each camera, each point in a

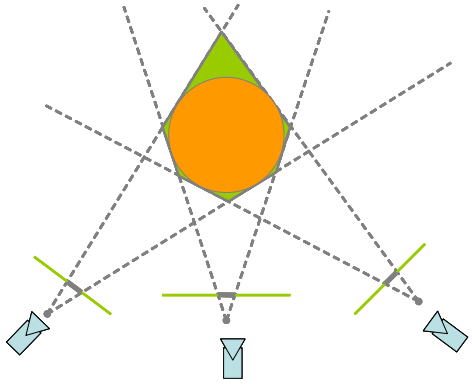


Figure 2.4: Object(orange color) and its visual hull(green color) by the silhouettes

silhouette defines a ray in shape space that intersects the object at some unknown depth along this ray. The union of these visual rays for all points in the silhouette defines a generalized cone within which the 3D object must lie.

Given a set of calibrated and silhouette-segmented images, the visual hull is the maximal solid volume that is consistent with each silhouette. The visual hull is given by the intersection of the input silhouette cones. The resulting intersection volume is a conservative bound on the object's shape (Shown in Fig. 2.4)

With the known camera parameter, we project all voxels to the input silhouette images, so we can get the locations of each voxel on silhouette images. Voxels that project outside the object's silhouette in one of the images cannot belong to the visual hull.

Octree [23] is one of the best known approaches to implement the visual hull. The volume of interest is first represented by one cube, which is progressively subdivided into eight subcubes. Once it is determined that a subcube is entirely inside or entirely outside the 3D reconstruction, its subdivision is stopped. Cubes are organized in a tree, and once all the leaves stop dividing, the tree gives an efficient representation of the 3D shape.

Octree subdivides the initial cube volume into eight subcubes progressively (Fig. 2.6). It is easy to manage the voxel size by defining subdivision depth, and it is good to perform a distributed computation to speed up the subdivision.

As a result, a volumetric shape composed of small voxels that is similar to the original 3D object is reconstructed. The precision of the reconstruction depends on the size of a voxel: the smaller a voxel is, the higher the precision we can get.

2.4 Voxel Coloring Reconstruction

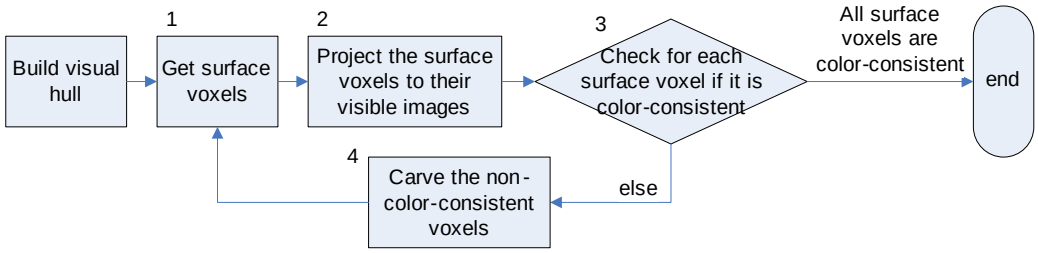


Figure 2.5: Procedure of Voxel coloring

The basic idea of voxel coloring reconstruction is to decide if the voxel of an object is photo-consistent.

photo-consistency, introduced by Seitz [26], is used to distinguish surface points from other points in a shape. The photo-consistency of all visible surface points with respect to each image is defined. A point on a scene surface is photo-consistent with a set of images if, for each image in which that point is visible, the image irradiance of that point is equal to the intensity at the corresponding image pixel. Usually a Lambertian reflectance model is considered to calculate photo-consistency.

Voxel coloring algorithms begin with a reconstruction volume of initially opaque voxels that encompasses the shape to be reconstructed. As the algorithm runs, opaque voxels are tested for photo-consistency, and those that are found to be inconsistent are carved. The algorithm stops when all the remaining opaque voxels are photo-consistent. When these final voxels are assigned the colors they project to in the input images, they form a model that closely resembles the shape.

First a one-pass procedure is introduced and extended to general camera configurations. Then a multi-pass procedure is developed that evaluates each voxel in the current plane of voxels using the subset of cameras and voxels that are in front of that plane. [16] describes this implementation as Space Carving.

The process flow of voxel coloring reconstruction is shown in Fig. 2.5. Because the visual hull is a convex hull of the real object, there must be some voxels on its surface that do not belong to the real object. So, we have to repeat the operations marked 1,2,3,4 in Fig. 2.5 until all surface voxels are photo-consistent. This resembles a ”peeling” process.

As explained above, there are two important factors that greatly affect a lot to the result of voxel coloring reconstruction:

- A good method that can access the voxels in the correct order, from the surface to the centroid of a given 3D volume.
- An efficient method that values the ”photo-consistency.”

The first factor indicates that volume data structure and its scanning algorithm have to be optimized easily for ”peeling” the visual hull. The second one indicates that color information is not only the RGB value of every pixel. It has to be devised to let a non-surface voxel shows larger variance. How we choose the two factors is introduced in the following subsections.

Volume Data Structure

Octree data structure, as we introduced above, because of its efficiency in speed and space, is widely used in the visual hull reconstruction. It also provides a way to refine the reconstruction result hierarchically.

As shown in Fig. 2.6(A), during the subdivision, when a parent voxel is divided into eight small child voxels, a link from the parent voxel to a child voxel can be created. To access to a terminal child voxel, we can simply follow this link progressively to the end.

For voxel coloring implementation, octree handles the images as a z-buffer plane. It projects all the voxels to an image, and sorts the voxels that are projected to the same pixel by their z-value. The z-value is calculated as the distance from a voxel to the camera screen.

For each pixel, following the sorted order, a list of voxels that are projected on this pixel can be created. The first element of the list is a surface voxel. If this voxel is carved, then the second element becomes the new surface voxel. For octree structure, there are only links between the voxels with different division levels.

Despite Octree’s advantage, we found it much harder to control all voxels in the terminal division level.

Another data structure often used is a voxel array. It divides the initial space volume into small voxels with the same size directly, and puts them into an array in the order of each voxel’s position. We can access a voxel by specifying the index of the voxel on the array. Fig. 2.6(B) shows the array data structure and the calculation of the voxel index.

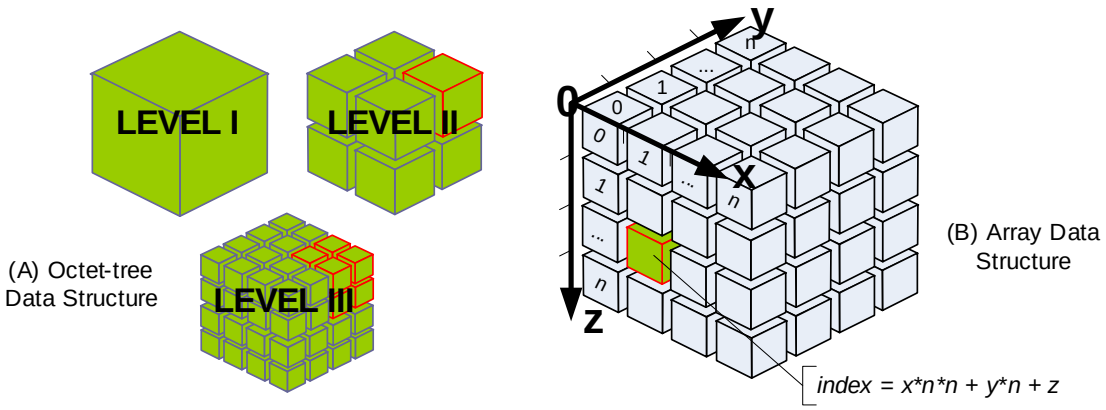


Figure 2.6: Two data structures

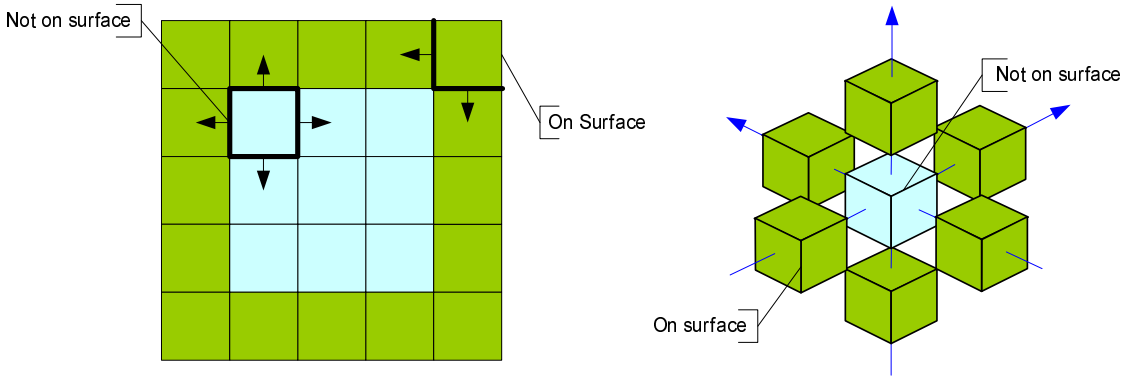


Figure 2.7: Determining the Surface Voxels

By combining the advantages of these two data structures, we decide to use octree data structure in the process of visual hull reconstruction, then load all the voxels to an array. Considering our method of peeling the visual hull, we adopted the voxel array for voxel coloring reconstruction, which can examines photo-consistency of voxels easily. Here we give a brief introduction to our implementation.

We improved the implementation of finding the surface voxels. As can be seen in Fig. 2.7, a voxel that is completely surrounded by its neighbor voxels cannot be a surface voxel. For a 3D voxel, there are 6 surrounding voxels that are shown in the right part of Fig. 2.7. So we determine whether a voxel is on the surface by checking its 6 neighbor voxels' existences instead of projecting all voxels to the color images. This

method saves much computation.

Calculate photo-consistency

Seitz and Dyer [26] give the original description of Voxel Coloring. In their formulation, the set of pixels in image j from voxel i that is visible is computed. The union of all views defines the color of that voxel:

$$\Pi_i = \bigcup_{j=0}^{m} \pi_j^i$$

where m is the number of images from which the voxel can be seen. To determine if the voxel is photo-consistent, the likelihood ratio test of this set is computed and thresholded. Therefore, if

$$(\boldsymbol{m} - 1)\,\sigma_{\Pi_i} \leq \tau$$

the voxel is labeled consistent. (where τ is the threshold parameter)

However, thresholded variance has its difficulty associated with photo-consistency measurement. A histogram-based photo-consistency method has been proposed [27]. Instead of pooling all of the pixels from all of the views of a given voxel, as was done in [26], this method use a series of tests on image pairs.

Found the set pi_j^i of pixels from image j that are visible from a voxel i. We test the consistency of the voxel by comparing all pairs of such sets:

$$\forall_{k,l} \pi_k^i \approx \pi_l^i \,(\boldsymbol{k} \neq \boldsymbol{l}) \tag{2.13}$$

where π_k^i and π_l^i are non-empty.

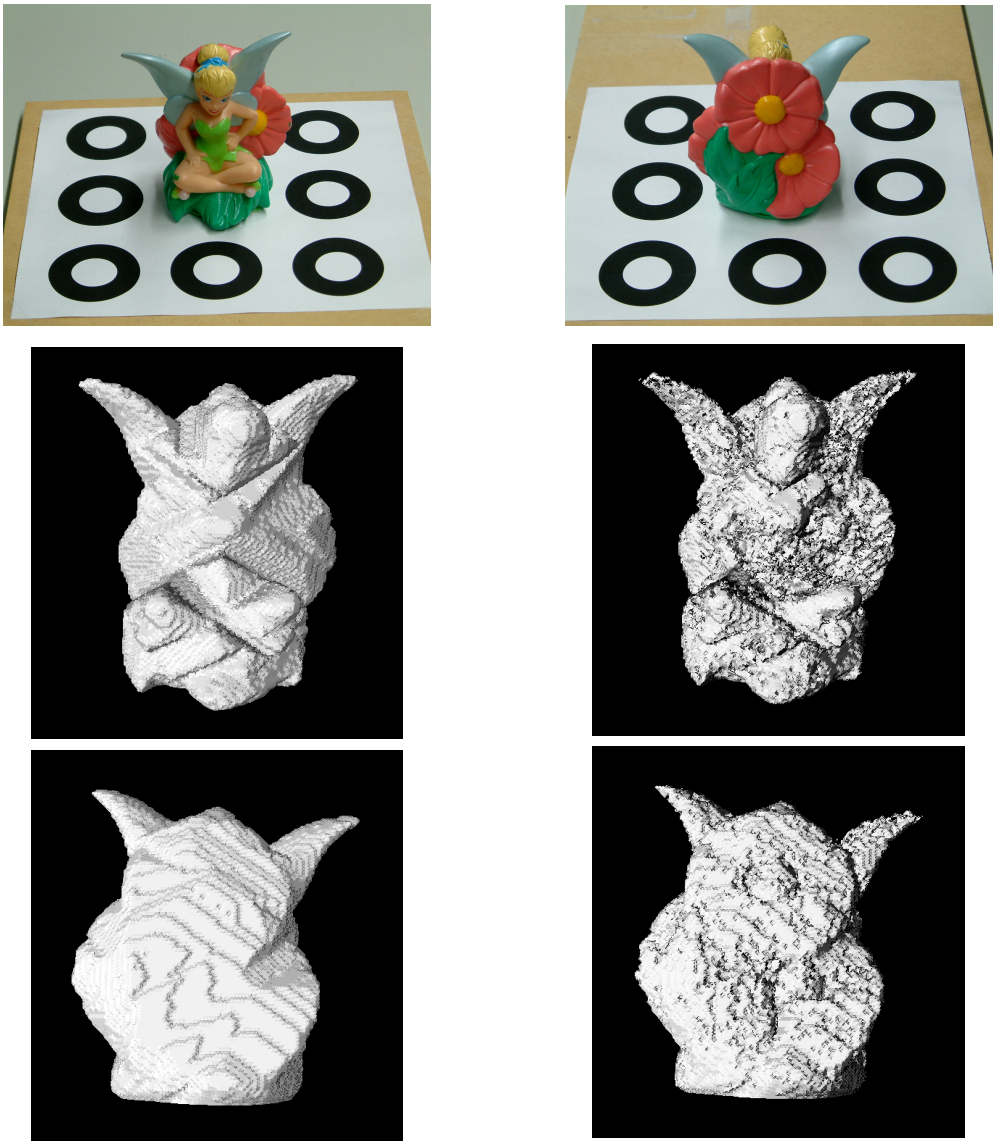
A voxel is announced consistent if all of the histograms of all views of the voxel intersect:

$$\forall_{k,l} Hist\left(\pi_k^i\right) \cap Hist\left(\pi_l^i\right) \neq \Phi \quad (\boldsymbol{k} \neq \boldsymbol{l}) \tag{2.14}$$

Pairs of histograms intersect if at least one pair of corresponding bins has a non-zero count. Therefore, a single pair of views can cause a voxel to be declared inconsistent if the colors they see at the voxel do not overlap.

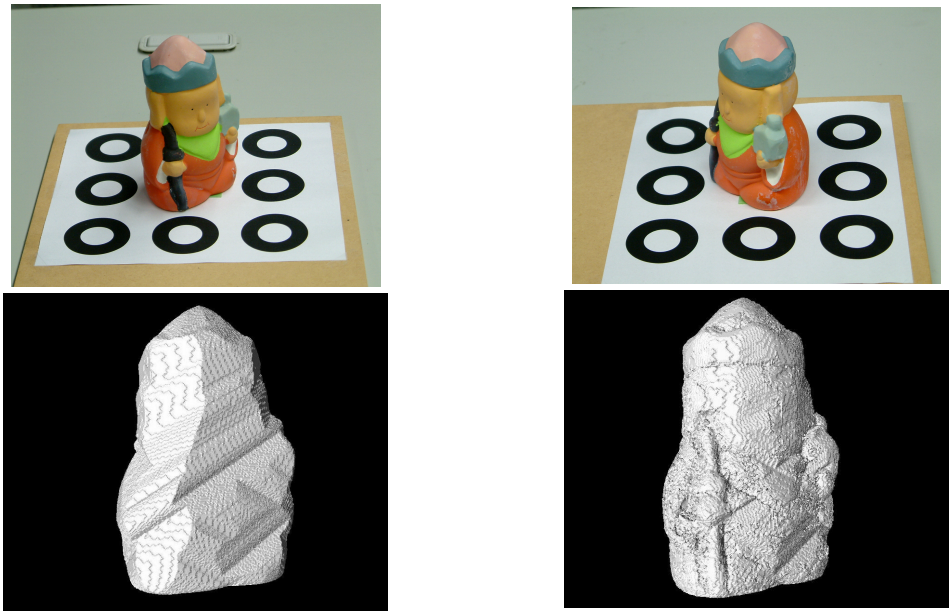
We implemented this histogram method over RGB color space and found that 8 bins per channel of RGB (total 512 bins (8x8x8)) are adequate for acceptable reconstructions.

We have run a comparison between the visual hull method and voxel coloring method. For this comparison, we have used 16 images of the object taken from different directions. Fig. 2.8 and Fig. 2.9 show two example images from this dataset.



Visual Hull Voxel Coloring

Figure 2.8: Reconstruction result 1



Visual Hull Voxel Coloring

Figure 2.9: Reconstruction result 2

2.5 Mesh representation

We use mesh representation to the reconstruction shape by applying the Marching-cubes algorithm [17].

The Marching-cubes algorithm is used in volume rendering to construct an isosurface from a 3D field of values. The 2D analog would be to take an image, and for each pixel, set it to black if the value is below some threshold, and set it to white if it's above the threshold. Then smooth the jagged black outlines by skinning them with lines. Marching-cubes algorithm tests the corner of each cube (or voxel) in the scalar field as being either above or below a given threshold. This yields a collection of boxes with classified corners. Since there are eight corners with one of two states, there are 256 different possible combinations for each cube. Then, each cube can be replaced with a surface that meets the classification of the cube. The result of the marching cubes algorithm is a smooth surface that approximates the isosurface that is constant along a given threshold.

Lorensen [17] reduces the original 256 combinations of cell state down to a total of 15 combinations, considering cell combinations that duplicate under the following conditions:

- Rotation by any degree over any of the 3 primary axes
- Mirroring the shape across any of the 3 primary axes
- Inverting the state of all corners and flipping the normals of the relating polygons

Kenmochi [31] improves the algorithm to take account of 23 combinations considering rotation only. We take this method to generate the mesh representation.

Furthermore, to reduce calculation cost later in the body tracking, this representation is simplified by applying a mesh simplification method [8]. It introduces a very efficient data structure that can be used to represent a triangle mesh at multiple levels of detail, also allowing progressive refinement and transmission. The basic idea of this technique is to construct multi-resolution representation by iteratively performing edge collapses and storing the simplified mesh along with the inverse of all performed edge collapses (vertex splits).

We show this as in Fig. 2.10.

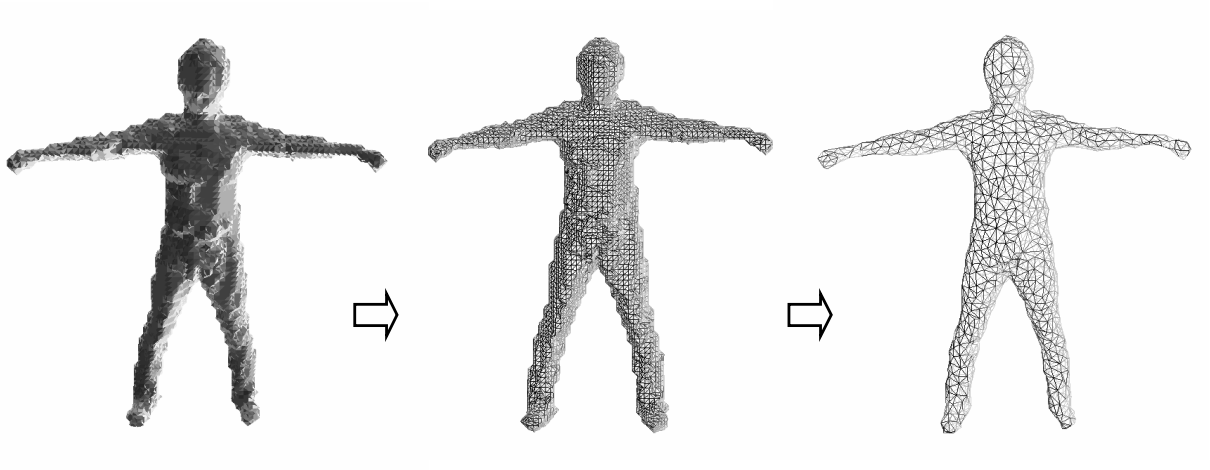


Figure 2.10: Mesh Representation of Body Volume

2.6 Texture mapping

Once we have the reconstructed the 3D human body shape, since the camera parameter is already known, we just combine the color images of each camera together

to produce a final textured model. There is one issue to solve at this stage: we have to decide, for each vertex on the mesh, which image to use if more than one is available.

The image is selected using the following criteria:

- If the mesh vertex visible from the camera, then the corresponding image is a candidate image for that vertex.
- For each candidate image, we can compute the angle between the vertex normal and the viewing direction, and choose the one image whose angle is the smallest.

And we use z-buffer algorithm to deal with the visibility problem (Shown in Fig. 2.11). For the hidden surface, the texture of the corresponding camera is omitted.

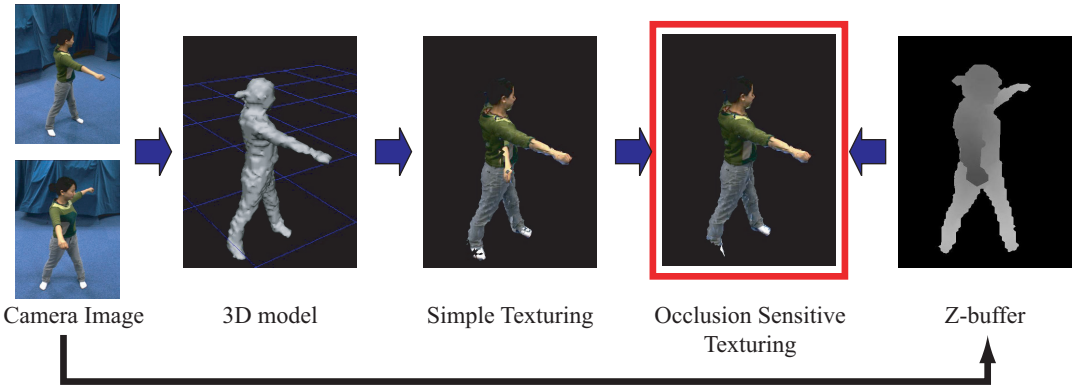


Figure 2.11: Texture mapping using z-buffering

With OPENGL rendering, the results of our multi-view texture mapping are shown in Fig. 2.12.

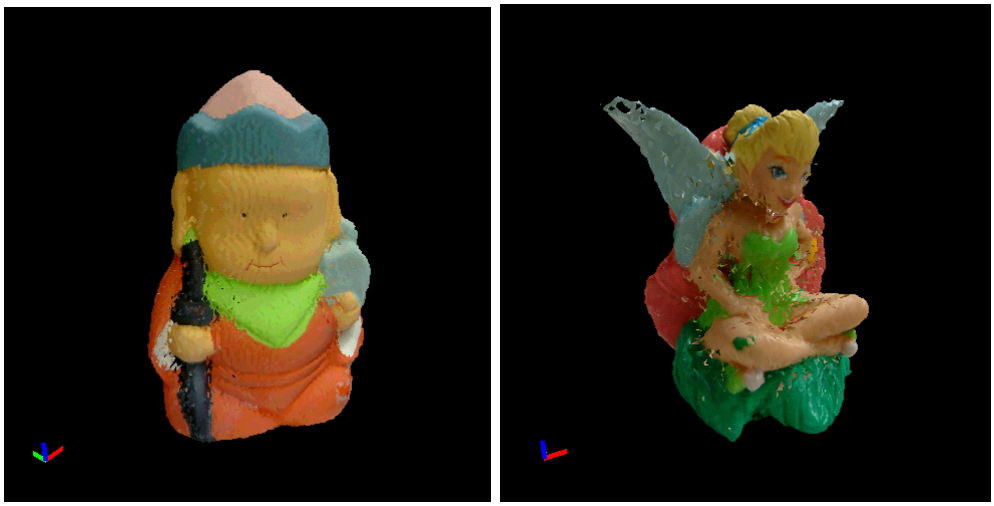


Figure 2.12: Reconstruction results with texture

Chapter 3

Articulated Human Body Model

In designing a robust motion tracking system that produces kinematically valid motion estimates, the choice of a human body model is critical. The basic requirements of the proposed human body model are as following:

- The model must represent the human motion correctly.
- The model is made as simple as possible but estimates robustly.

Regarding the first concept, our robotics group uses motion capturing system VICON [28] to track human body motion. We design a compatible model that can change to the model used in the robotics group.

Regarding the second concept, to save computation, our model consists of a simple structure under constraints, and deformable mesh representation is adopted. We construct our articulated human body model composed of a kinematic skeletal model and a surface mesh model, which leads to a better tracking result than the simple segment model. We will show this in Chapter 5.

3.1 Kinematic Skeleton Model

The kinematic skeleton model is represented by skeletal links in hierarchical structure are shown in Fig. 3.1. We set the base of the model (Root) between the upper and lower body and define links with no joints like chests and hips. The depth is up to six. We define the relationship as that, between two neighbor levels, the link in higher level is the parent of the link in lower level, the link in lower level is the child of the link in the higher level, and the leaf link has no children.

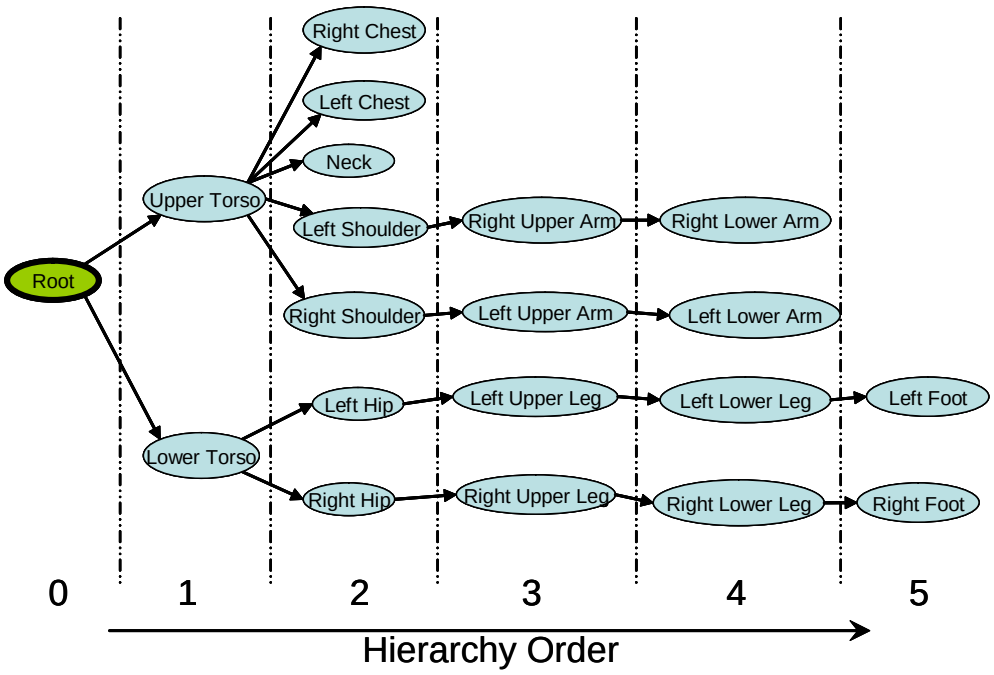


Figure 3.1: Hierarchial Structure of Skeleton Model

A connection between links indicates a joint, and 14 joints with accumulated 29 degrees of freedom (DOF) are modeled as shown in Fig. 3.2: a 3-DOF joint for the torso, a 3-DOF joint for each shoulder and hip, a 2 DOF for the collarbone, a 2 DOF for each foot, a 2-DOF joint for the neck and a 1-DOF joint for each of the knees and the elbows.

In addition to 29 DOF for joints, the skeleton model has an extra 6 DOF, 3 DOF for translation and 3 DOF for rotation, which determines the pose of the entire human body.

Our kinematic skeleton model is defined in a file containing the configuration between skeletal links. In order to find a transformation from each link to the origin, we adopt Denavit Hartenberg [12] representation to describe the kinematics of the links connected together by various joints.

To perform the kinematic analysis, we attach a coordinate frame $o_i x_i y_i z_i$ denoted by Σ_i to link i. This means when joint i is actuated, link i and its attached frame Σ_i experience a resulting motion. Σ_0 is referred to the initial link, the base of the articulated model.

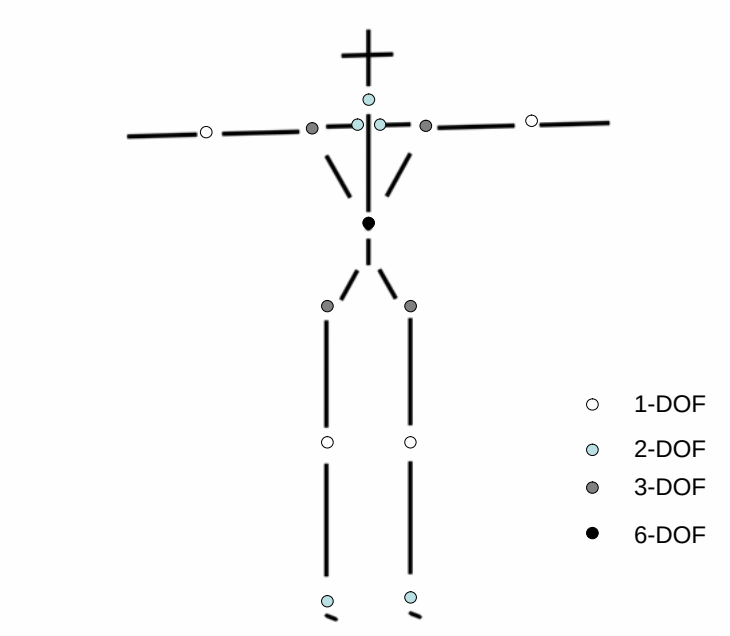


Figure 3.2: Kinematic Skeleton Model

Suppose transformation matrix $\mathbf{A}_i$ expresses the position and orientation of Σ_i with respect to the neighbored coordinate frame Σ_{i-1}.

$$\boldsymbol{A}_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

where the four quantities θ_i, a_i, d_i, α_i are parameters associated with link i and joint i. θ_i, a_i, d_i, α_i are

- Rotation around $\boldsymbol{z}_{i-1}$ with angle θ_i
- Translation along $\boldsymbol{z}_{i-1}$ with displacement d_i
- Translation along $\boldsymbol{x}_{i-1}$ with displacement a_i
- Rotation around $\boldsymbol{x}$ with angle α_i

Furthermore, transformation matrix of $\mathbf{A_j}$ with respet to $\mathbf{A}_i$ is denoted by $\mathbf{T}^i_j$.

$$\mathbf{T}^i_j = \mathbf{A}_{i+1}\mathbf{A}_{i+2}\cdots\mathbf{A}_{j-1}\mathbf{A}_j \quad if \quad i < j \tag{3.2}$$

An example of an arm is taken to further explain the kinematics of the body model (Fig. 3.3). Table 3.1 shows these link parameters in Fig. 3.3. With these parameters transformation matrix $\mathbf{A}_1$ of coordinate frame Σ_1 to Σ_0 is

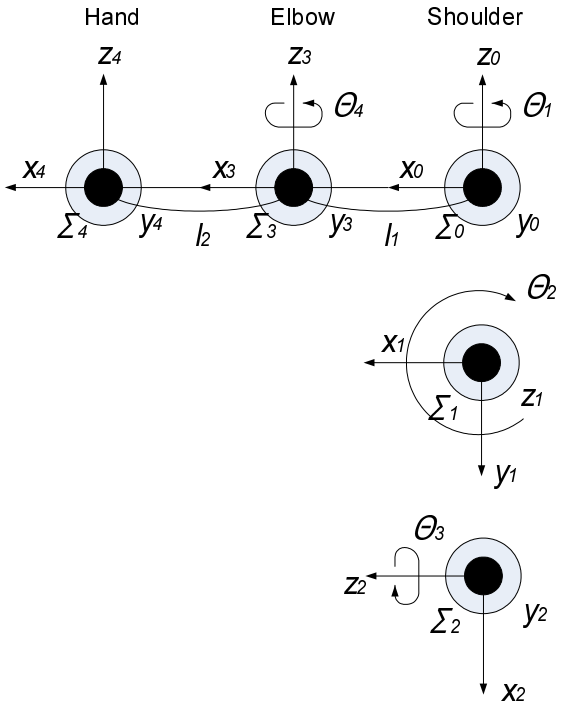


Figure 3.3: Example of Joint of an Arm

Table 3.1: Link parameter

Joint	θ	d	α	a
1	θ_0	0	$-\pi/2$	0
2	θ_1	0	$\pi/2$	0
3	θ_2	0	$\pi/2$	l_0
4	θ_3	0	0	l_3

$$\boldsymbol{A}_1 = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1\cos\left(-\frac{\pi}{2}\right) & \sin\theta_1\sin\left(-\frac{\pi}{2}\right) & l_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1\cos\left(-\frac{\pi}{2}\right) & -\cos\theta_1\sin\left(-\frac{\pi}{2}\right) & l_1\sin\theta_1 \\ 0 & \sin\left(-\frac{\pi}{2}\right) & \cos\left(-\frac{\pi}{2}\right) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

$$= \begin{pmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & l_1\cos\theta_1 \\ \sin\theta_1 & 0 & \cos\theta_1 & l_1\sin\theta_1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.4}$$

The position and orientation of each joint are determined uniquely by such a matrix.

For instance, to calculate the position of elbow joint, the transformation matrix **T** should be:

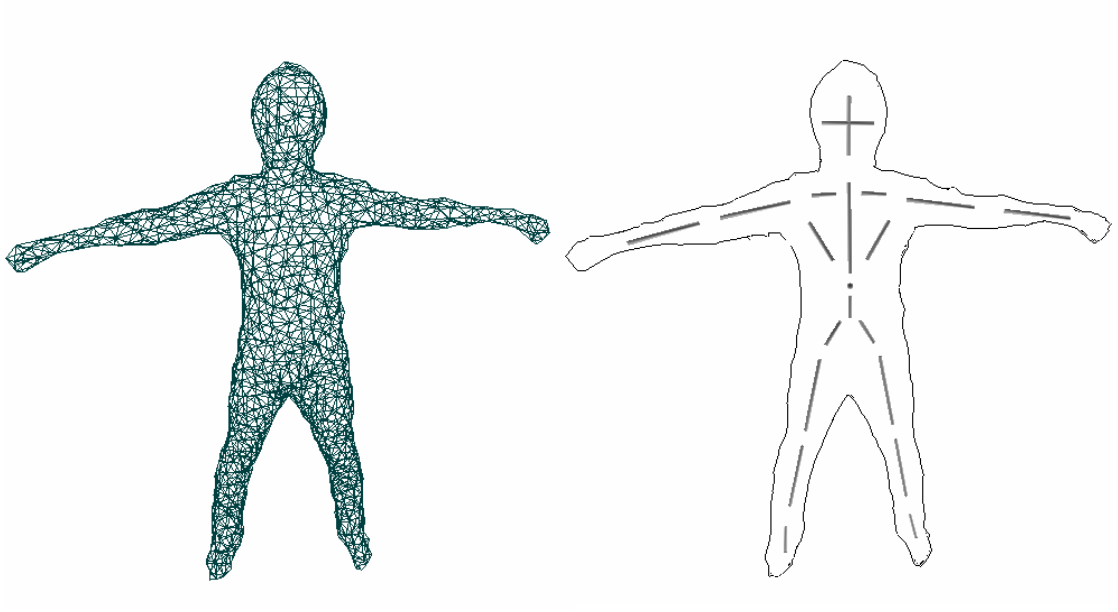
$$\begin{aligned} \boldsymbol{T} &= \boldsymbol{A}_1\boldsymbol{A}_2\boldsymbol{A}_3 \\ &= \begin{pmatrix} C_1 & 0 & -S_1 & l_1C_1 \\ S_1 & 0 & C_1 & l_1S_1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_2 & 0 & S_2 & 0 \\ S_2 & 0 & -C_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_3 & 0 & -S_3 & l_2C_3 \\ S_3 & 0 & C_3 & l_2S_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

The position of the elbow joint in Σ_0 is calculated by multiplying **T** to origin $(0, 0, 0, 1)^{\mathrm{T}}$ in Σ_3.

3.2 Surface Mesh Model

To allow the model to undergo realistic physical deformations, it is necessary to consider the skin surface bound to the skeleton.

In this study, the surface model of a human body is constructed by the reconstruction of a real human body. The subject adopts an initialization pose standing upright with its arms and legs spread in the pose shown in Fig. 3.4.



mesh view edge view with skeleton inside

Figure 3.4: initial pose of the human body model

The kinematic skeleton model and the surface mesh model overlap, and each vertex in the surface model is linked to the corresponding bones. When a bone rotates about a joint, all the linked vertices move to a new location. To determine this new location of all the relative links, weighted interior division is adopted.

The calculation process is as follows:

1. For each vertex i, calculate the shortest path length x_i to the link l using the dijkstra algorithm [6].

   The dijkstra algorithm, also called dijkstra shortest path algorithm, is an algorithm that solves the single-source shortest path problem for a directed graph with non-negative edge weights.

The input of the algorithm consists of a weighted directed graph G and a source vertex s in G. We will denote V the set of all vertices in the graph G. Each edge of the graph is an ordered pair of vertices (u, v) representing a connection from vertex u to vertex v. The set of all edges is denoted E. $w(u, v)$ is the non-negative cost of moving from vertex u to vertex v . The cost of an edge can be thought of as the distance between those two vertices. The cost of a path between two vertices is the sum of costs of the edges in that path. For a given pair of vertices s and t in V, the algorithm finds the path from s to t with the lowest cost (i.e., the shortest path).

Notice that the path should not penetrate any mesh (Fig. 3.5).

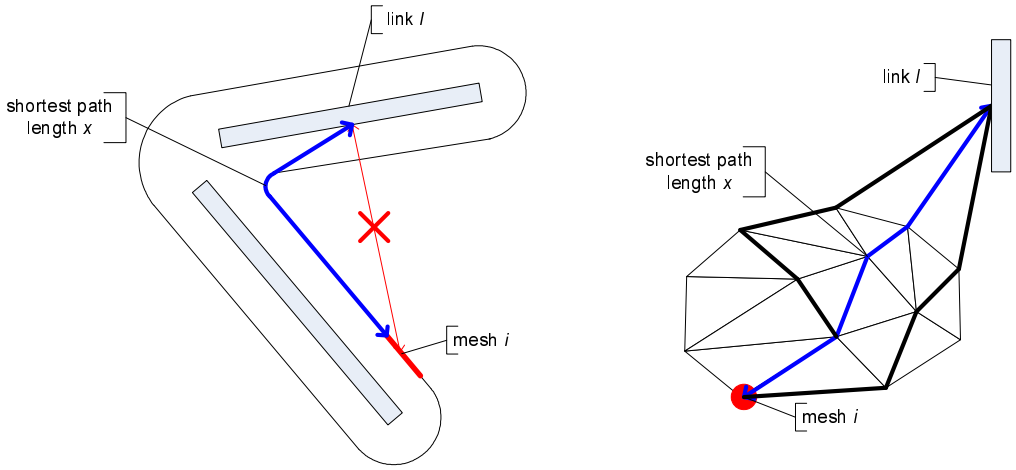


Figure 3.5: Weight interior divison of skin attechment

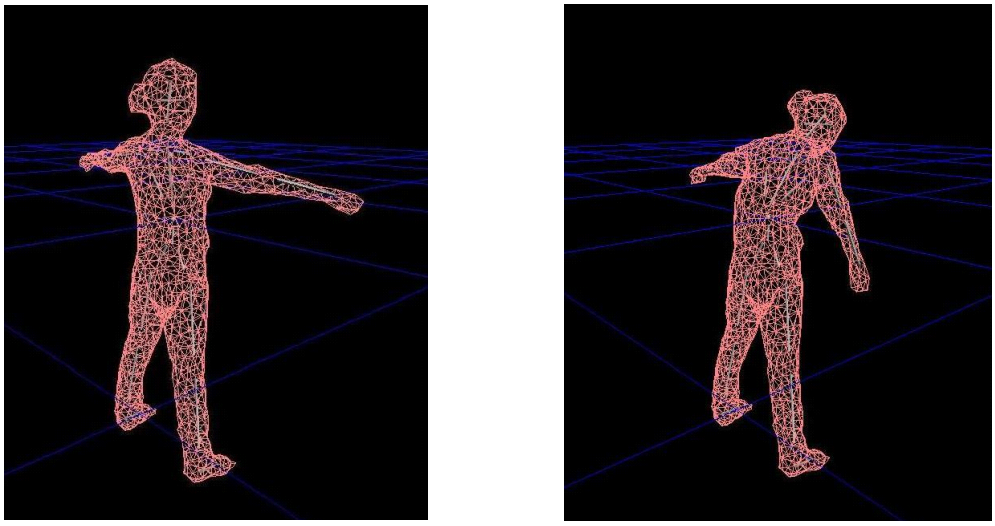
2. Determine the weight of vertex i to the link l using the exponential function as $Weight_{il} = ae^{-bx_i}$.

3. Normalize $Weight_{il}$ to satisfy that for all the links, $\Sigma_l Weight_{il} = 1$.

The normalized weight will be used later to estimate the joint angle. The calculation of the weight is done only once at the beginning to construct the articulated body model. This technique allows surface vertices to be influenced by not just one link, and avoids unnatural deformations appearing near the joints. Thus the mesh model deforms smoothly around each joint (Fig. 3.6).

As described above, the human body model is generated following these steps:

1. Generate the kinematic skeletal model: set the joint location, link the parent and child joint, and define the DOF constraint of each joint.



One side View Deformed View

Figure 3.6: Surface Mesh Model

2. Generate surface mesh model: capture the images of a human body with the initial pose from cameras; reconstruct the human body shape and represent it with meshes.

3. Align the skeletal model and surface mesh model: make sure they are at the same position and that the surface mesh overlaps the bone entirely

4. Generate correspondence of the skeletal and mesh models to complete the articulated human body model

Chapter 4

Human Body Motion Tracking

The pose and joint angle of a reconstructed human body volume are estimated by fitting the reconstructed data with the articulated body model in 3D space. The goal is to minimize the distance between the surface of the 3D model and the 3D reconstruction.

With regard to localization techniques between a rigid 3D geometric model and 3D range data, Besl et al. proposed the iterative closest point (ICP) algorithm, which iteratively searches for the corresponding model vertex to each 3D range point. Then the optimal correspondence, i.e., the pose of the model, is calculated by solving the least squares method using all the correspondences [1].

However, ICP is sensitive to outliers, so Wheeler et al. proposed a 3D template matching (3DTM) technique that utilizes M-estimator from robust statistics to exclude the effect of outliers [29]. This technique has been used, for example, to track a manipulated object in 3D space [22].

[21] extended 3DTM to handle an articulated object model so that the joint angles as well as the pose of a hand can be robustly estimated by applying an articulated hand model to a reconstructed hand volume.

We extend this work to the human body case, which has a more complicated shape and joint structure. The tracking framework will be described in the subsequent sections.

4.1 Objective Function

Once the reconstructed 3D volumetric shape has been computed, we need to locate the correspondences between the reconstructed data points and points on the human

body model.

Our objective function could be the minimum of the sum of the distances of model vertices to the corresponding reconstruction voxels.

Generally, for one point m_i on the model, we define the closest point d_i to point m_i on the reconstruction data the corresponding point of point m_i.

Given the large number of model vertices, however, a subset $\boldsymbol{M}$ of tracking vertices $\boldsymbol{m}_i$ is sampled. For each $\boldsymbol{m}_i$ on the current model configuration $\boldsymbol{p}_i$, the corresponding point of the reconstructed data $\boldsymbol{v}_i$ is sought.

Mathematically, the objective function is:

$$F(\boldsymbol{p}_i) = \sum_{M} \|\boldsymbol{m}_i - \boldsymbol{v}_i\|^2 \tag{4.1}$$

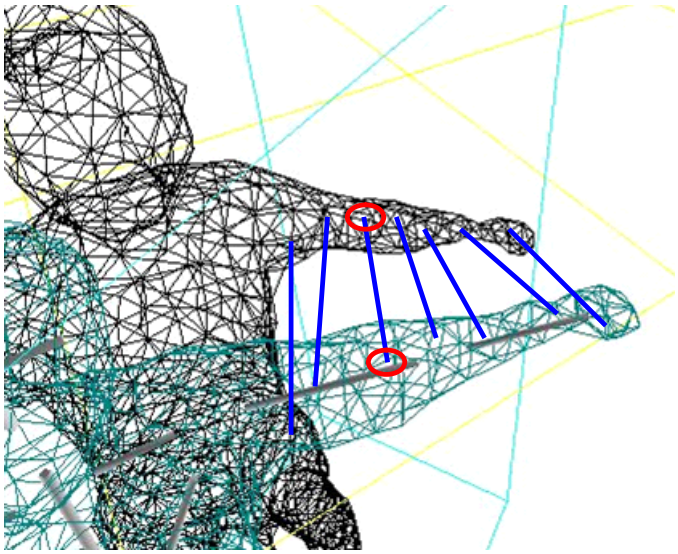


Figure 4.1: Illustration of the objective function in the case of one arm

Fig. 4.1 illustrates the objective function with corresponding points. Solid blue lines indicate the assignment of tracking vertices on the model to their closest points on reconstructed data.

Correspondence Search

Computing nearest neighbor correspondences efficiently between the human body model points and reconstructed shape points will aid our search for the correct pose

estimation. A quick local search is desired and Kd-tree is adopted.

K-d (k-dimensional) tree is a generalization of binary-search trees in two or more dimensions. The k-d tree is created by recursively splitting a data set down the middle of its dimension of greatest variance (Fig. 4.2). The splitting continues until the number of data points in each leaf node is less than the threshold. The result is a tree of depth $O(logn)$ where n is the number of points stored in the k-d tree.

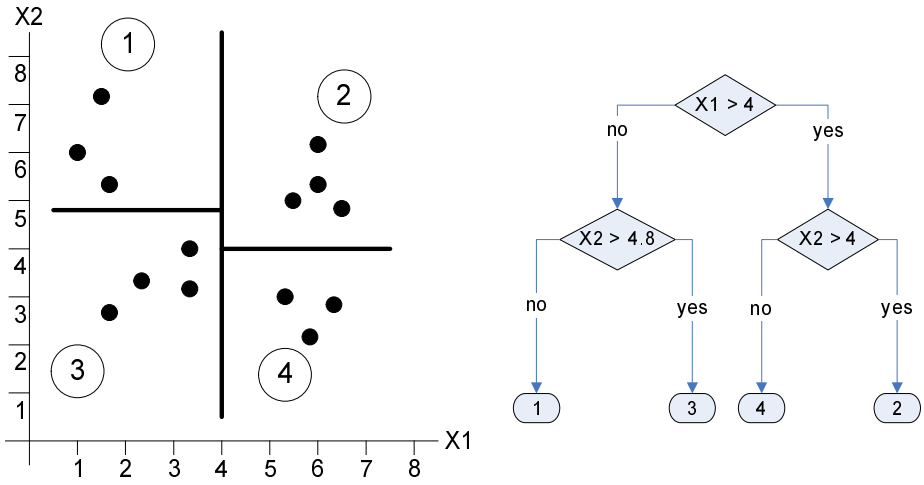


Figure 4.2: Kd-tree construction and search in 2D space. Points are divided in the above order. The leaf node has the coordinates of points, while the others have the information of the axis and the value where the data points are divided

In most cases, calculating only the distances between the k-d tree nodes is enough to locate corresponding points. But considering some special cases, such as, the initial position estimate is too far or, perhaps, the current pose estimate is closer to an object other than the desired object, then the 3-dimensional k-d tree may not be sufficient (Fig. 4.3(a)).

Thus, an extension of using surface normal similarity, in which the surface normal of the model point should be similar to the normal of its matching image point, is considered (Fig. 4.3(b)). The ideal dissimilarity measure for comparing two unit vectors (normals) is the angle between the two vectors.

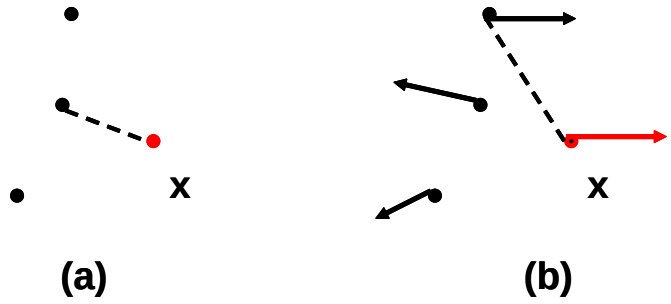


Figure 4.3: An example of nearest distance correspondence (a) and nearest distance + normal correspondence (b)

4.2 3DTM

We introduce the 3DTM algorithm in this section.

The point $\boldsymbol{r}_i$ in the reconstructed volume corresponding to a point $\boldsymbol{m}_i$ in the body model can be represented with two parameters, $\boldsymbol{t}$, a translation vector, and $\boldsymbol{R}$, a rotation matrix as Eq.(4.2).

$$\boldsymbol{r}_i = \boldsymbol{R}\boldsymbol{m}_i + \boldsymbol{t} + \beta \tag{4.2}$$

where β is random 3D noise. If β follows a Gaussian distribution, $< \boldsymbol{R}, \boldsymbol{t} >$ can be estimated by minimizing Eq.(4.3) by solving the least squares method.

$$f(\boldsymbol{R}, \boldsymbol{t}) = \sum_i \|\boldsymbol{R}\boldsymbol{m}_i + \boldsymbol{t} - \boldsymbol{r}_i\|^2 \tag{4.3}$$

where $\boldsymbol{R}$ is represented using a unit 4-vector called a quaternion. Appendix B shows the detail about quaternion representation.

But the real error distribution usually doesn't follow a Gaussian distribution, and the effect of outliers makes the localization process unstable. Therefore, Wheeler proposed a technique to apply M-estimator to estimate the real error distribution [29]. M-estimator is a generalized form of the least squares method and is formulated as Eq.(4.4).

$$E(\boldsymbol{p}) = \sum_i \rho(z_i) \tag{4.4}$$

where $\boldsymbol{p} = (\boldsymbol{t}^{\mathrm{T}}, \boldsymbol{q}^{\mathrm{T}})^{\mathrm{T}}$ is the position and the rotation of the rigid object model and $\boldsymbol{q}$ is a 4 D.O.F. quaternion vector which is equivalent to $\boldsymbol{R}$. $\rho(z_i)$ is an arbitrary function of the error $z_i (= \beta^2)$.

$\boldsymbol{p}$ which minimizes $E(\boldsymbol{p})$ can be solved from Eq.(4.5).

$$\frac{\delta E}{\delta \boldsymbol{p}} = \sum_i \frac{\delta \rho(z_i)}{\delta z_i} \frac{\delta z_i}{\delta \boldsymbol{p}} \tag{4.5}$$

Here, we introduce $w(z)$ as a weight function which represents an error term as Eq.(4.6).

$$w(z) = \frac{1}{z} \frac{\delta \rho}{\delta z} \tag{4.6}$$

With Eq.(4.6), Eq.(4.5) can be rewritten as Eq.(4.7). If we ignore the fact that $w(z)$ is a function of z, this is a form of weighted least squares.

$$\frac{\delta E}{\delta \boldsymbol{p}} = \sum_i w(z_i) z_i \frac{\delta z_i}{\delta \boldsymbol{p}} \tag{4.7}$$

In this study, Lorentzian distribution is chosen as a probability distribution of error to exclude the effect of outliers, and the weight function is defined as follows:

$$w(z) = \left(1 + \frac{1}{2}\left(\frac{z}{\sigma}\right)^2\right)^{-1} \tag{4.8}$$

Eq.(4.7) can be solved by conjugate gradient search algorithm, and $\boldsymbol{p}$, which minimizes the error, is obtained.

4.3 Extension to an Articulated Model

In section 4.2, $E(\boldsymbol{p})$ is described as Eq.(4.9).

$$E(\boldsymbol{p}) = \sum_i \rho\left(\left\|\boldsymbol{R}\boldsymbol{m}_i + \boldsymbol{t} - \boldsymbol{r}_j\right\|^2\right) \tag{4.9}$$

To handle an articulated model, the above equation is rewritten as Eq.(4.10).

$$E(\boldsymbol{p}, \boldsymbol{\theta}) = \sum_i \rho\left(\left\|\boldsymbol{R}\boldsymbol{m}_i(\boldsymbol{\theta}) + \boldsymbol{t} - \boldsymbol{r}_j\right\|^2\right) \tag{4.10}$$

$$\begin{pmatrix} \boldsymbol{m}_i(\boldsymbol{\theta}) \\ 1 \end{pmatrix} = \prod_l \boldsymbol{T}_l(\theta_l) \cdot \begin{pmatrix} \boldsymbol{m}_i \\ 1 \end{pmatrix} \tag{4.11}$$

where $T_l(\theta_l)$ is a 4×4 homogeneous matrix which takes l-th joint angle and converts a point in the coordinate frame of a child link to that of the parent link and $\prod_l$ means from the root of the model to the current link.

Eq.(4.11) shows the case of which not utilizes deformable mesh model. As for the deformable mesh model, the weight of meshes to each link calculated in Section 3.2 is considered and Eq.(4.11) can be rewritten as Eq.(4.12).

$$\begin{pmatrix} \boldsymbol{m}_i(\boldsymbol{\theta}) \\ 1 \end{pmatrix} = \sum_{l=1}^{S_l} w_l \prod_{j \in S_l} \boldsymbol{T}_j(\theta_j) \cdot \begin{pmatrix} \boldsymbol{m}_i \\ 1 \end{pmatrix} \tag{4.12}$$

where w_l is the mesh weight to the link l, and S_l is the set of links.

4.4 Estimation Order

The process of estimating the position and joint angle of the human body in our system is called ”Ordered fitting”. Here, ”ordered” means it will fit the links in the order of the hierarchy level shown as the number at the bottom of Fig. 3.1. The base of the model is noted as level 0, and the last child is as level n. Here n=6.

The ordered fitting process can be separated into 2 steps as following: position fitting and joint angle fitting (Fig.4.4).

1. Position Fitting
   In the position fitting step, translation and rotation are performed on the body model to move the model towards the given posture data, in other words, to locate the position of the link with level 0. Meshes around the links in level 1, and links without 0 DOF as right chest, left chest, right hip and left hip are used to calculate the similarity between model and data.

2. Joint Angle Fitting
   In joint angle fitting step, the joint angle of each link is fitted to the given data. It process in the order of level 1, level 2, ... to level 6. Links in the same hierarchy level are processed simultaneously in separated fitting steps.

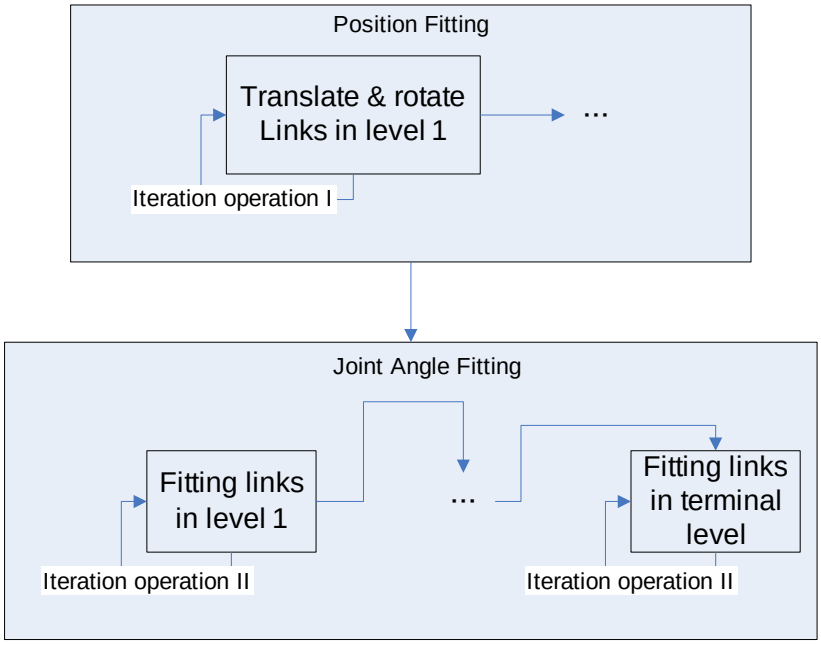


Figure 4.4: Ordered Fitting Process

The estimation process of both the position fitting and joint angle fitting can be written as follows.

1. **repeat**
2. $\boldsymbol{p} = \boldsymbol{p}', \boldsymbol{\theta} = \boldsymbol{\theta}'$
3. Calculate $E(\boldsymbol{p}, \boldsymbol{\theta})$
4. Calculate gradient $\frac{\partial E(\boldsymbol{p},\boldsymbol{\theta})}{\partial \boldsymbol{p}}$
5. Estimate new $\boldsymbol{p}'$ which minimizes $E(\boldsymbol{p}', \boldsymbol{\theta})$
6. Calculate $E(\boldsymbol{p}', \boldsymbol{\theta})$
7. Calculate gradient $\frac{\partial E(\boldsymbol{p}',\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$
8. Estimate new $\boldsymbol{\theta}'$ which minimizes $E(\boldsymbol{p}', \boldsymbol{\theta}')$
9. **until** $|E(\boldsymbol{p}', \boldsymbol{\theta}') - E(\boldsymbol{p}, \boldsymbol{\theta})| < \varepsilon$

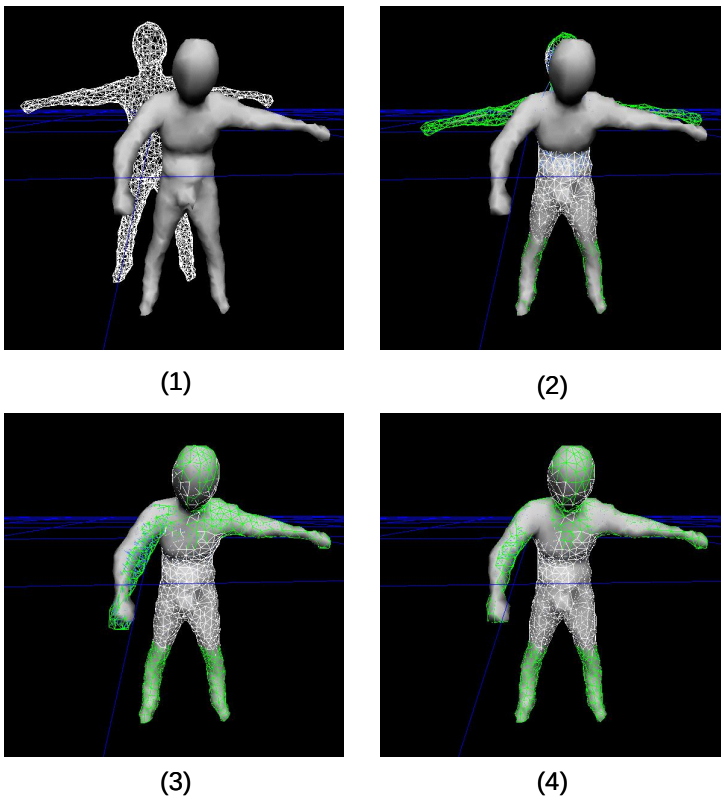


Figure 4.5: Fitting Procedure: (1) shows the initial pose of the model and (wire frame) recostruted data (surface); (2) shows the position fitting of the whole model; (3) shows the joint angle fitting (4) show the result of the whole fitting process

The example of estimation procedure is shown as in Fig. 4.5.

For the complex kinematic structure of the human body, some heuristics methods are implemented to improve the estimation results.

Improvement

For an articulated model with more than one link, especially the case of human body model, to estimate the joint angle of each $link_i$, determination of which meshes to use is crucial.

Using all the meshes of the model will cause the problem shown in Fig. 4.6 (a). Notice that the meshes reach to balance.

So we adopt only the nearest meshes to $link_i$, to solve this problem shown in 4.6 (b).

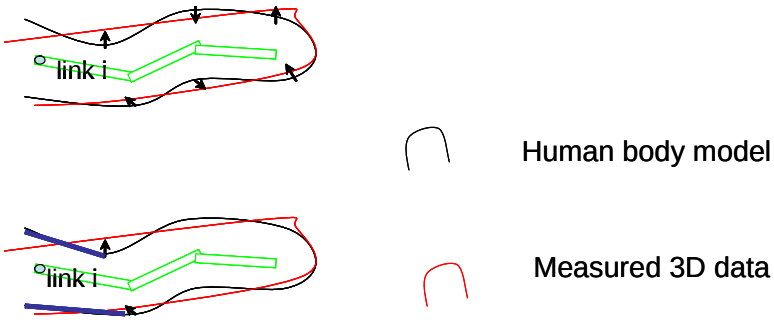


Figure 4.6: meshes contributed to the link

Sometimes the meshes around a single link are too few to calculate the similarity, as the cases like body twist. So we decide that when fitting a link *l*, all meshes around *l*'s children links are also used as *l*'s nearest meshes to evaluate the angle of link *l*. We call it "P-C relation".

The effect of this relation is shown in Fig. 4.7. The first line shows the top view of the fitting results in upper body twist with and without "P-C" relation. The red wire frame shows the body model and the gray volume shows the reconstructed data. The second line shows the front view of the fitting results in kinematic skeleton model.

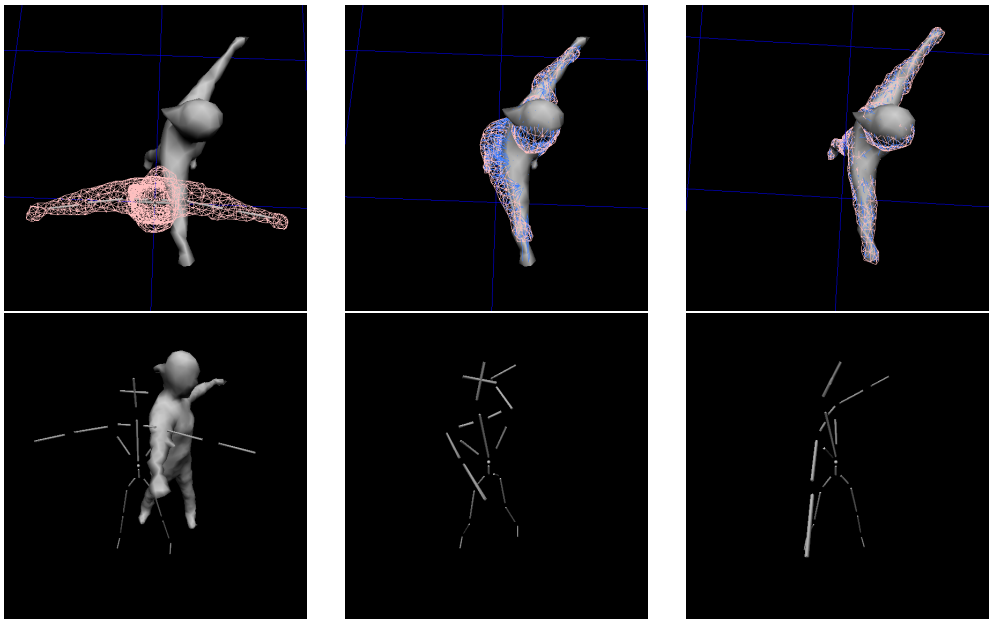


Figure 4.7: Result using parent-child relation

Chapter 5

Experiment

In the previous chapters we saw how to reconstruct a human body shape, how to generate a human body model, and how the tracking works.

The features of our system are:

- Applied voxel coloring algorithm for 3D human body reconstruction.
- Human body structure with attached skin of deformable mesh.
- Model hierarchy ordered posture estimation for motion tracking.

As we expect that the advantages of those methods can bring a precision motion tracking result, an examination experiment is described in this chapter, presenting the results of the tracking system both in simulation and real word environments

5.1 Deformable Mesh Model vs. Segment Model

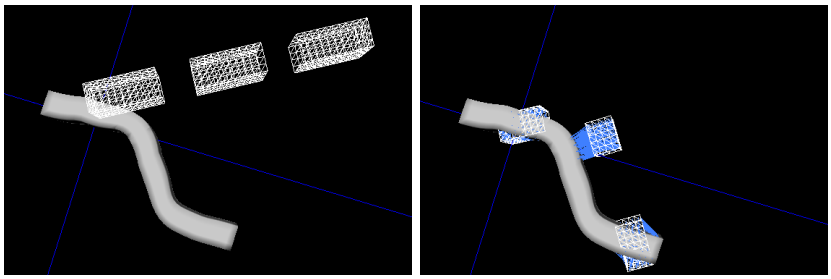
This section gives a comparison to show the merits of a deformable mesh model compared with segment model.

Segment model is the model constructed with segments; each segment has a transformation matrix to the origin just as the kinematic skeleton model. A joint locates between every two segments, and segments are linked together to generate the whole model. Usually, the segment model is generated using primitive shapes like cylinders or sphere. In our study, we generate the segment model by cutting the deformable model at its initial pose, to make sure that each segment in segment model translates and rotates the same as the corresponding link in kinematic skeleton model.

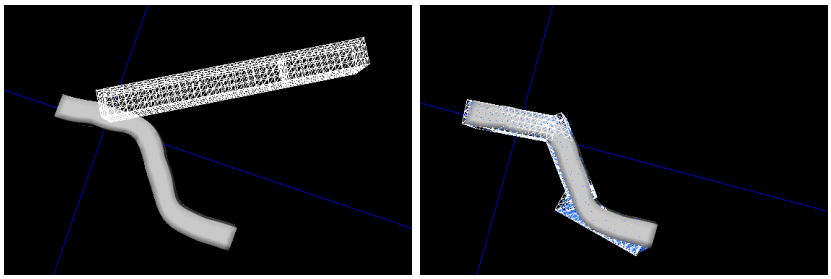
First, we evaluate it using a simple chain model with 2 joints (6 DOF), and the 3D voxel data to observed is generated from the model with Gaussian noise.

We have a check of the segment models with different lengths to generate the most suitable segment model (Fig. 5.1). We set the model position at (40, 30, 30) with rotation (30, 0, 0) and observed data is at (0, 0, 0) with rotation (0, 0, 0).

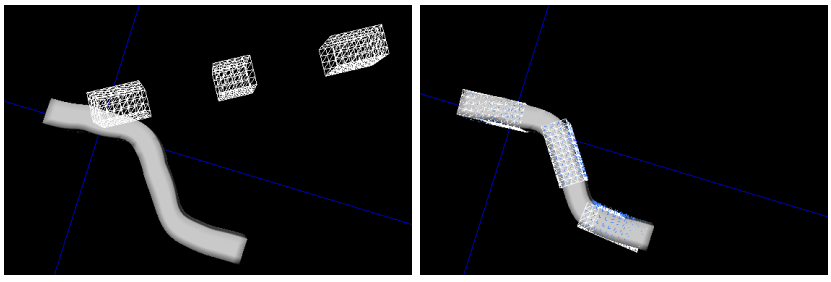
From Fig. 5.1, we can see that when the segment length is almost the same as the link length in the deformable mesh, it shows better estimation result than other segment models.



(1)short segment model



(2)long segment model



(2)suitable segment model

Figure 5.1: Fitting results of segment model with different lengths

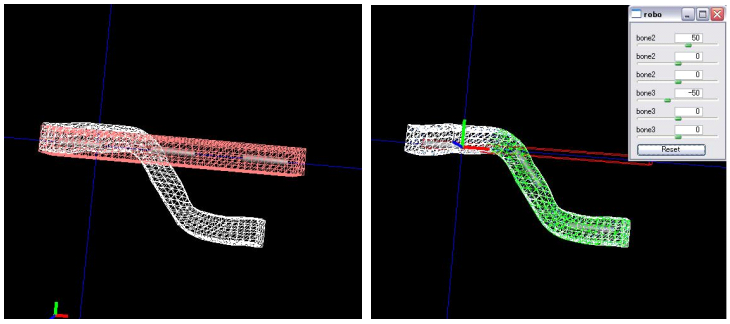
Then we evaluate the estimation results of the deformable mesh model and the segment model in the joint angle errors and distances errors between the model and the observed data.
error_ang is defined as

$$\text{err\_ang} = |\text{e\_ang} - \text{t\_ang}| \tag{5.1}$$

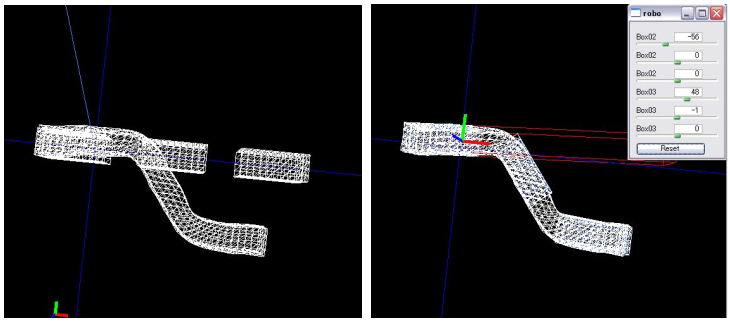
where e_ang is the estimated angle and t_ang is the set true angle of the model.
error_dis is defined as

$$\text{err\_dis} = \frac{\sum_{n=1}^{N} |\text{e\_pos} - \text{t\_pos}|}{N} \tag{5.2}$$

where e_pos represents the model meshes, t_pos represents the corresponding data meshes, and *N* is the number of the model meshes.



(1)deformable mesh model



(2)segment model

Figure 5.2: Fitting results of Deformable mesh model and Segment model: simulated chain model

The fitting results of deformable mesh and segment model are shown as Fig. 5.2.

Both the models and observed data is set at the same position, and the joint angle of the observed data is set with $(50, 0, 0, 50, 0, 0)$ degree and data is generated with $\sigma = 0.1$.

Fig. 5.3 shows that with different Gaussian noise, deformable mesh model shows robust fitting results both of the error in distance and joint angle.

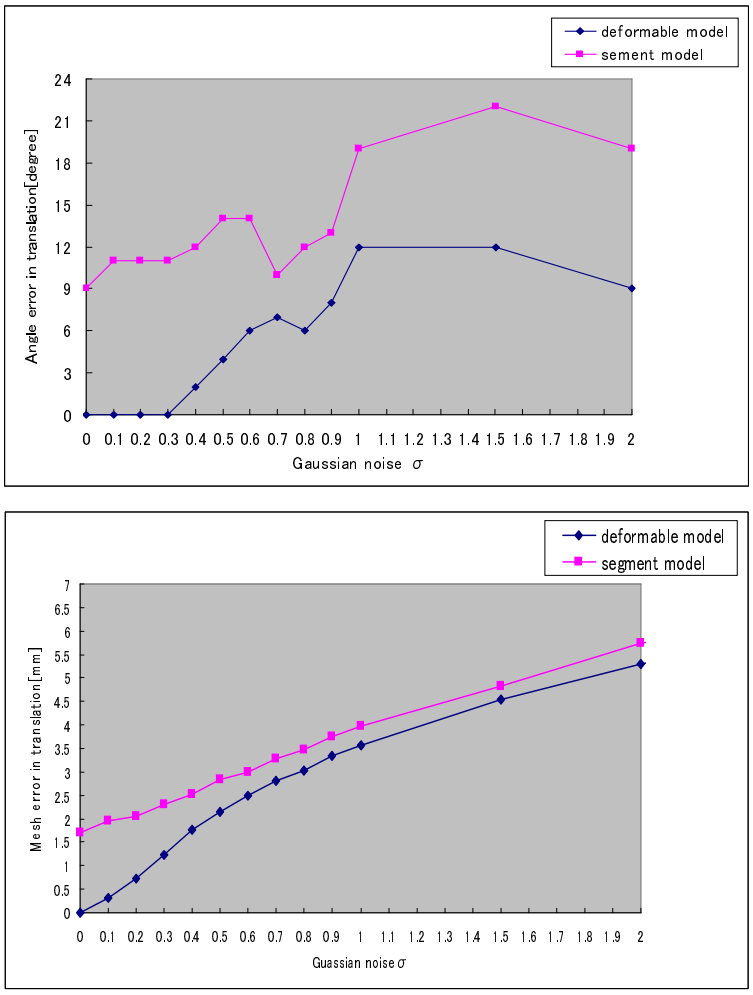


Figure 5.3: Comparasion between Deformable mesh model and Segment model

The same robust fitting result of the human body model is shown in Fig. 5.4. The segment model of the human body is generated by cutting the deformable human body model. It has the same kinematic structure as the skeletal model of the articulated human body model. Notice that because of the lack of meshes around the hips, the twist of the leg in the segment model shows a poor fitting result.

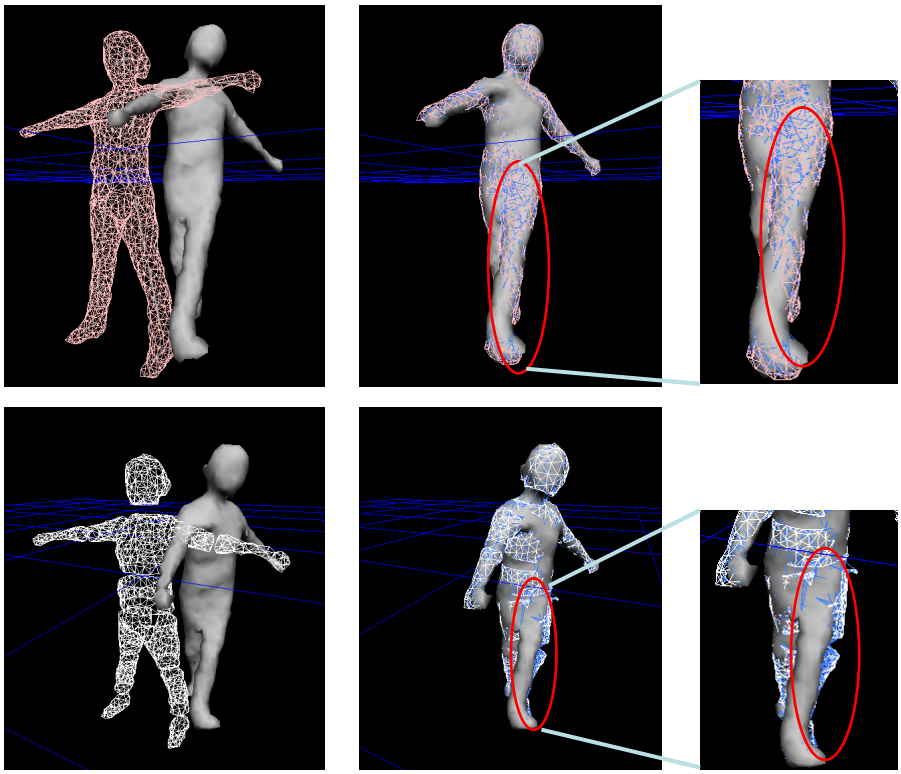


Figure 5.4: Fitting results of Deformable mesh model and Segment model: real constrcuted data of the human body

5.2 Tracking Evaluation

5.2.1 Simulation

The simulation experiments have been performed by using our designed 3D human body model as described in Chapter 3.

First, we show the convergence performance of our algorithm by fitting the body model to the body data generated from the same body model directly. We call such data ”standard data”.

Then, we reconstruct the human body shape of the model with virtual cameras and the reconstructed shape data is shown.

Finally, we evaluate the tracking performance between the standard data and the reconstructed data.

Convergence Performance

As described in Section 4.4, the estimation of the positon and joint angle of the given data is processed iteratively. Convergence performance is evaluated by the calculation of estimation error under different iterations shown in Fig. 5.5.

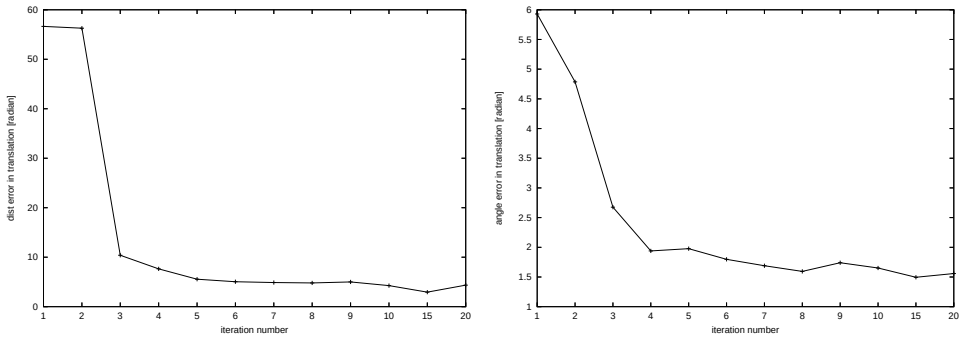


Figure 5.5: Estimation error of interations

As shown above, increasing the iterations can lead to better estimation results. Considering saving the computation time, and the results change little when the iterations is up to 10, so we set the iteration max number to 10 in this system.

3D shape reconstuction

The images taken from virtual cameras are arranged as shown in Fig. 5.6. All the cameras are on the top of the subjects.

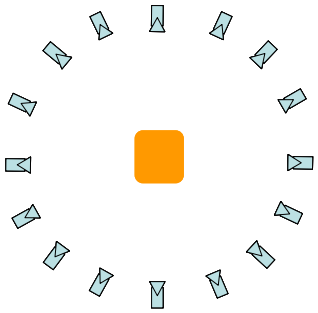
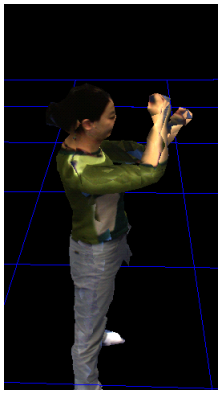


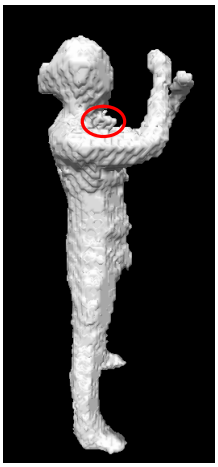
Figure 5.6: virtual camera configuration

We show the reconstructed human body shape using visual hull and voxel coloring algorithm as following:

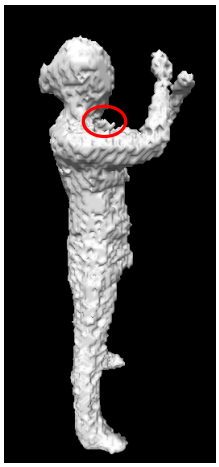
As shown in Fig. 5.7, there is difference though little can be seen for the human body shape in a normal pose and position.



one image
from virtual camera



visual hull
reconstruction

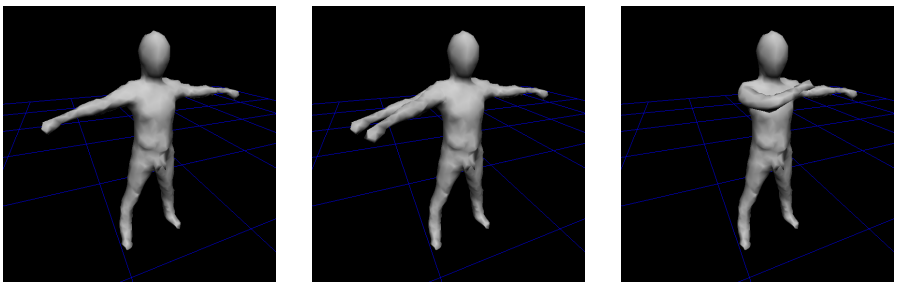


voxel coloring
reconstruction

Figure 5.7: Reconstruction result 1

Tracking

The rough initial pose can be estimated by a random sample approach or an example based learning approach. In this study, we assume that the rough initial pose of the first time frame is given. When the pose of the first frame is known, the pose in the subsequent frames can be estimated by applying the proposed fitting algorithm by setting the previous frame's pose as the initial pose of the current frame.



(a) original pose (c) last frame

Figure 5.8: The generated data sequence

As shown in Fig. 5.8, the observed data sets are generated by 15 degrees difference every frame in the joint angle of the arms. The joint angle of the initial frame is set as $(0, 0, 0)$, shown as (a) in Fig. 5.8, and the last frame is shown as (c) in Fig. 5.8.

Besides the recognition error in joint angles and surface patch distances, we compare the translation error and rotation error between standard data and reconstructed data as shown in Fig. 5.9.

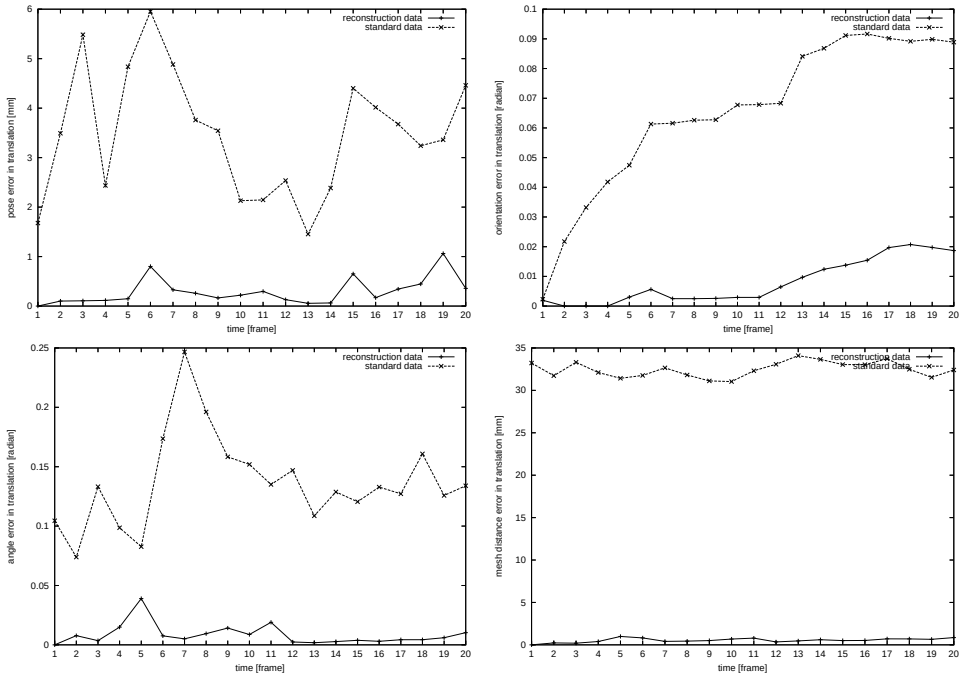


Figure 5.9: Estimation error

5.2.2 Real Environment

Our experiment results in the real environment are shown in the following sections.

Human Body Shape Reconstruction

Eight clustering PCs are set up to control the 8 synchronized cameras (SONY DXC-9000, 3CCD color video camera). The images are taken 30 frames/sec and arranged around the ceiling as Fig. 5.10.

To generate the 3D shape of the human body, we need camera calibration (Appendix A) to determine the relationship between what appears on the image plane and where it is located in the 3D world. The calibration cube in use is shown in Fig. 5.11.

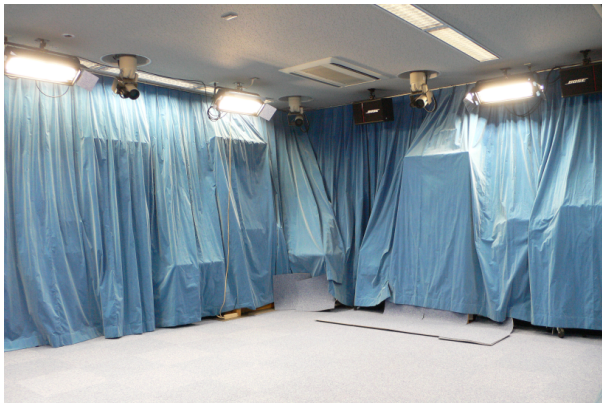


Figure 5.10: experiment space for image capture

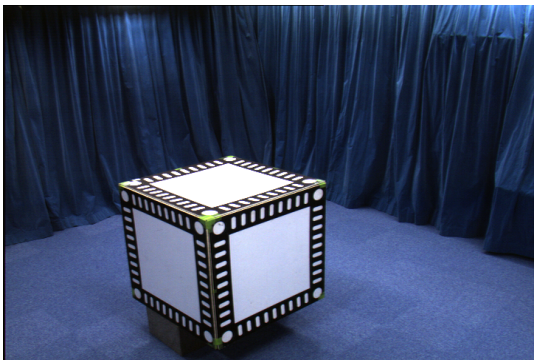


Figure 5.11: calibration box

Using the threshold $\tau = 0.97$, background segmentation results are shown in Fig. 5.12. And the reconstruction results are shown as Fig. 5.13.

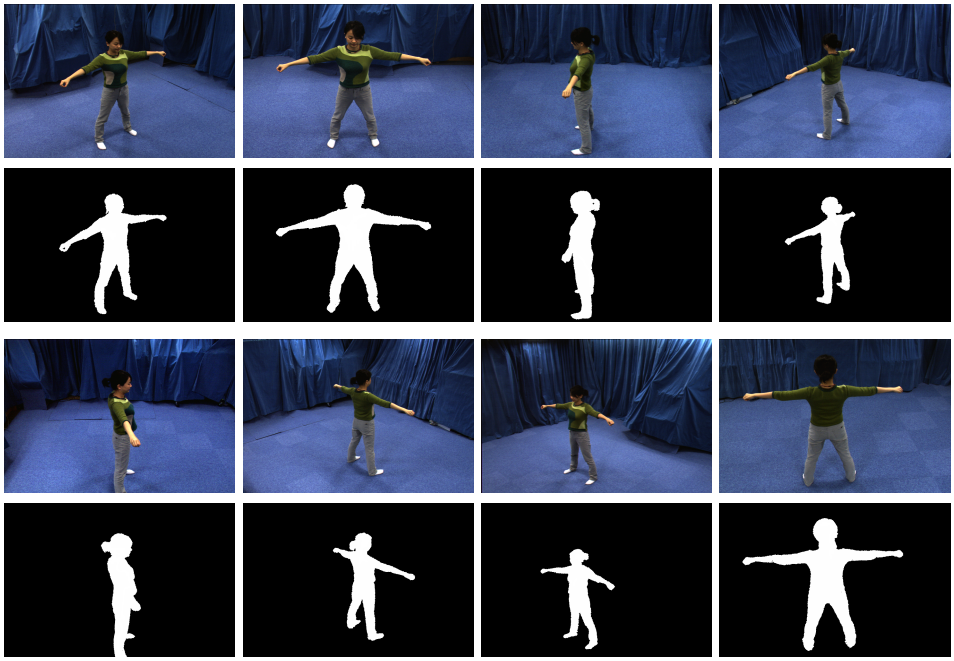


Figure 5.12: Segmentation results for 8 views

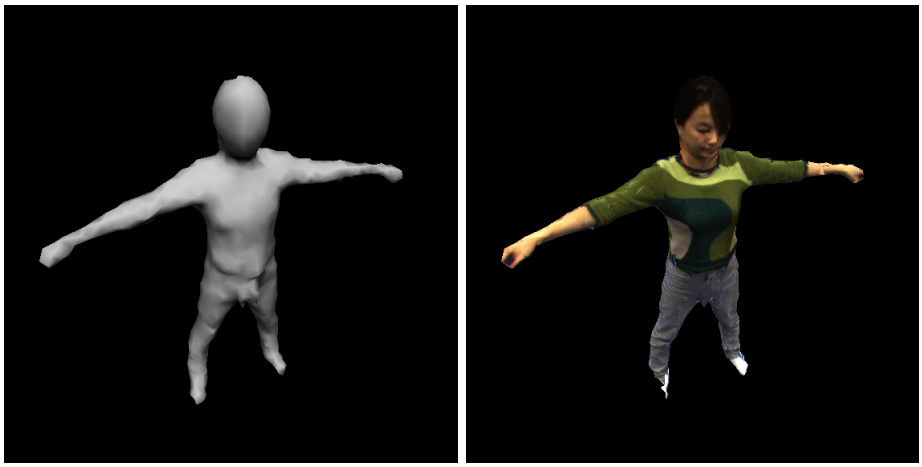


Figure 5.13: Reconstruction and texture mapping of the human body shape

Human body tracking

We show results for images of different body poses and motions.

- Fig. 5.14 and Fig. 5.15 are some results of tracking for the standing body while moving the upper/lower body.
  Note that we can track the pose of body parts coming close to each other. Kehl [15] shows that in their system, color information used to deal with such a pose leads to correct estimation. However, in our algorithm, there is no need to allocate color information to the body parts.

- Fig. 5.16 shows tracking results of upper-body twist, which is rarely mentioned in tracking systems because of its difficulty to estimate. Using the P-C relation link structure, it shows satisfying tracking results.

- Fig. 5.17 presents tracking results while the user is walking around whereas in Fig. 5.18, the user is jumping. Note the large motions and the presence of self-occlusions.

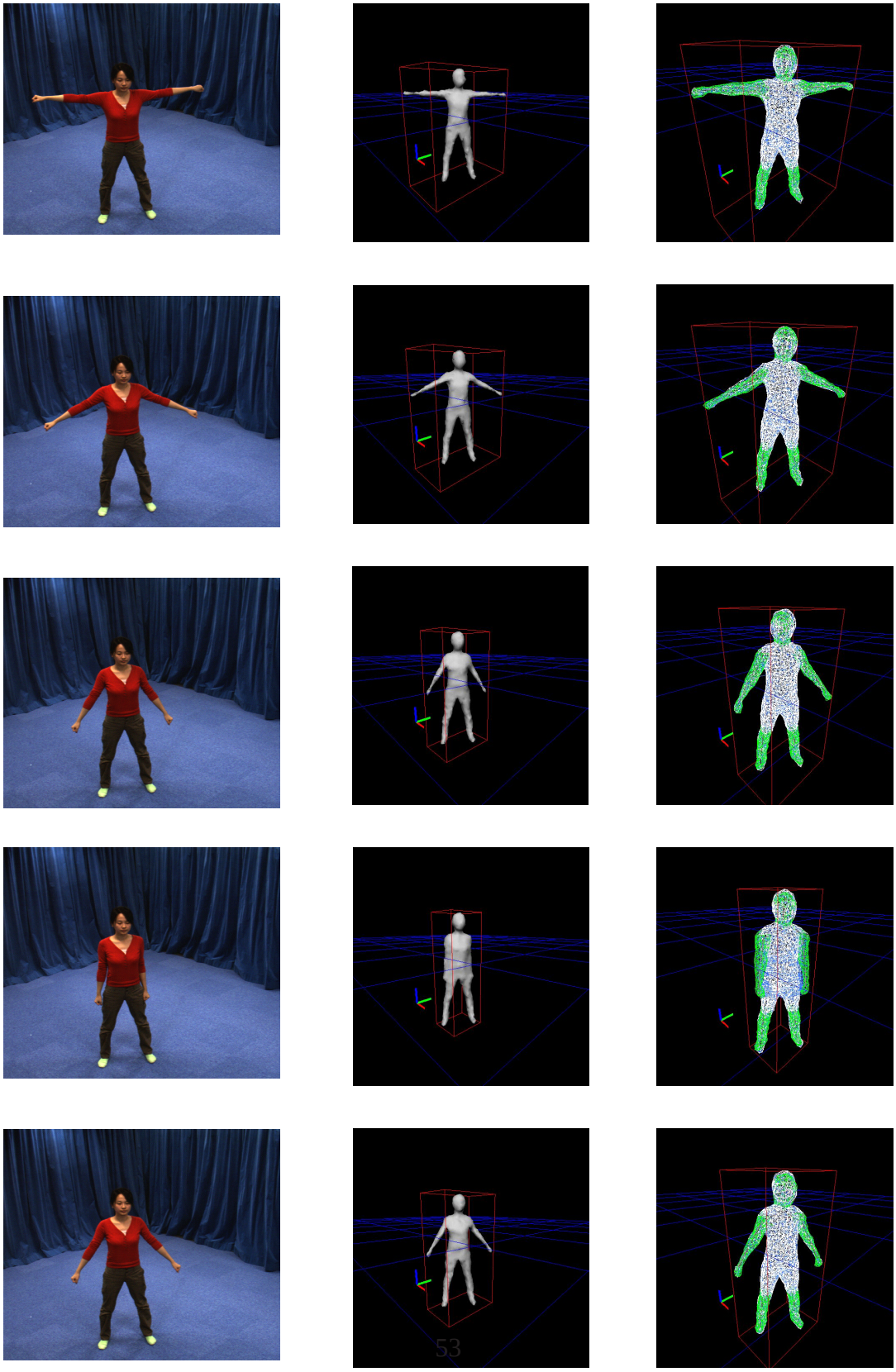


Figure 5.14: Tracking results of reconstruction data: upper body

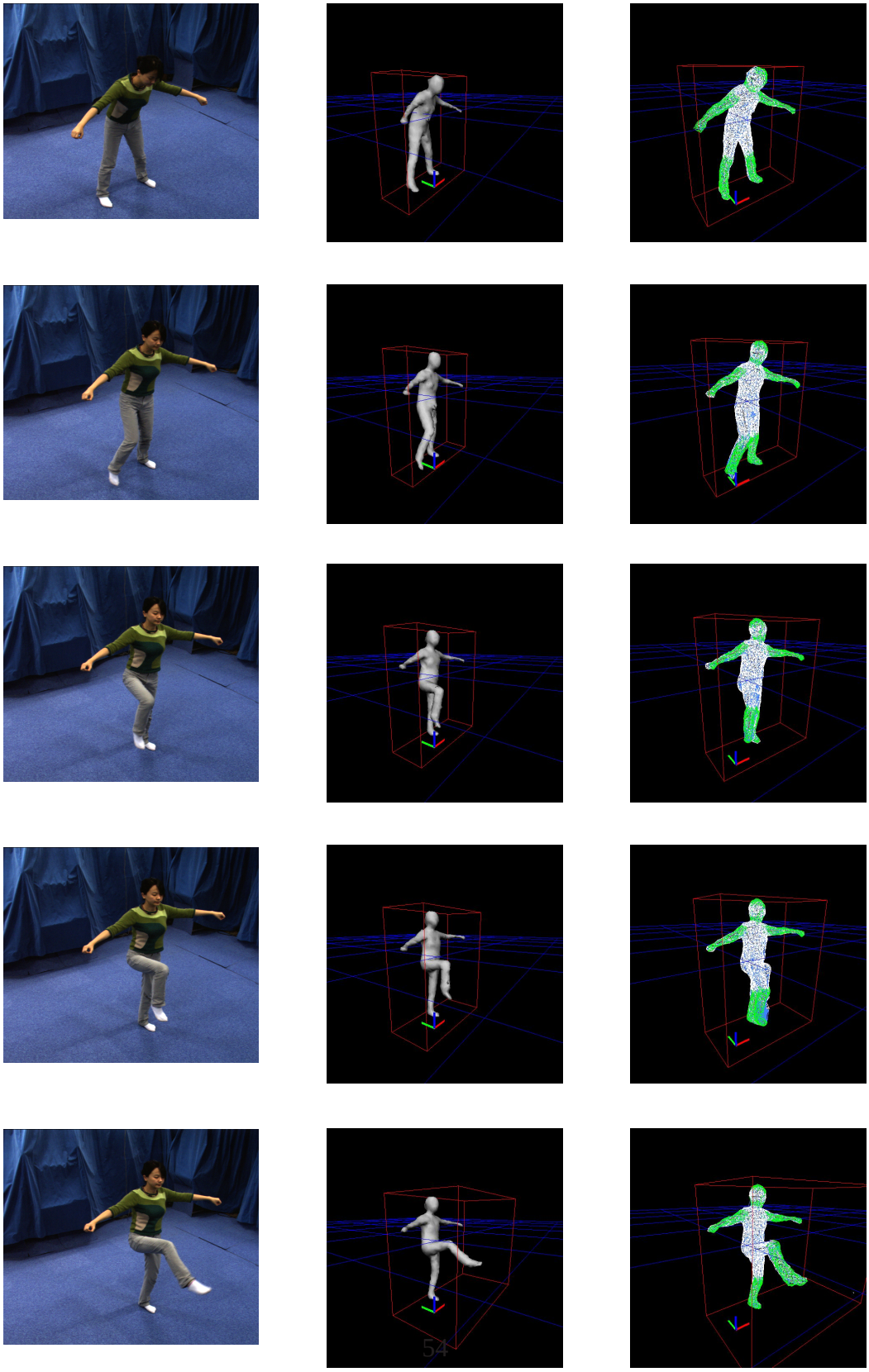


Figure 5.15: Tracking results of reconstruction data: Lower body

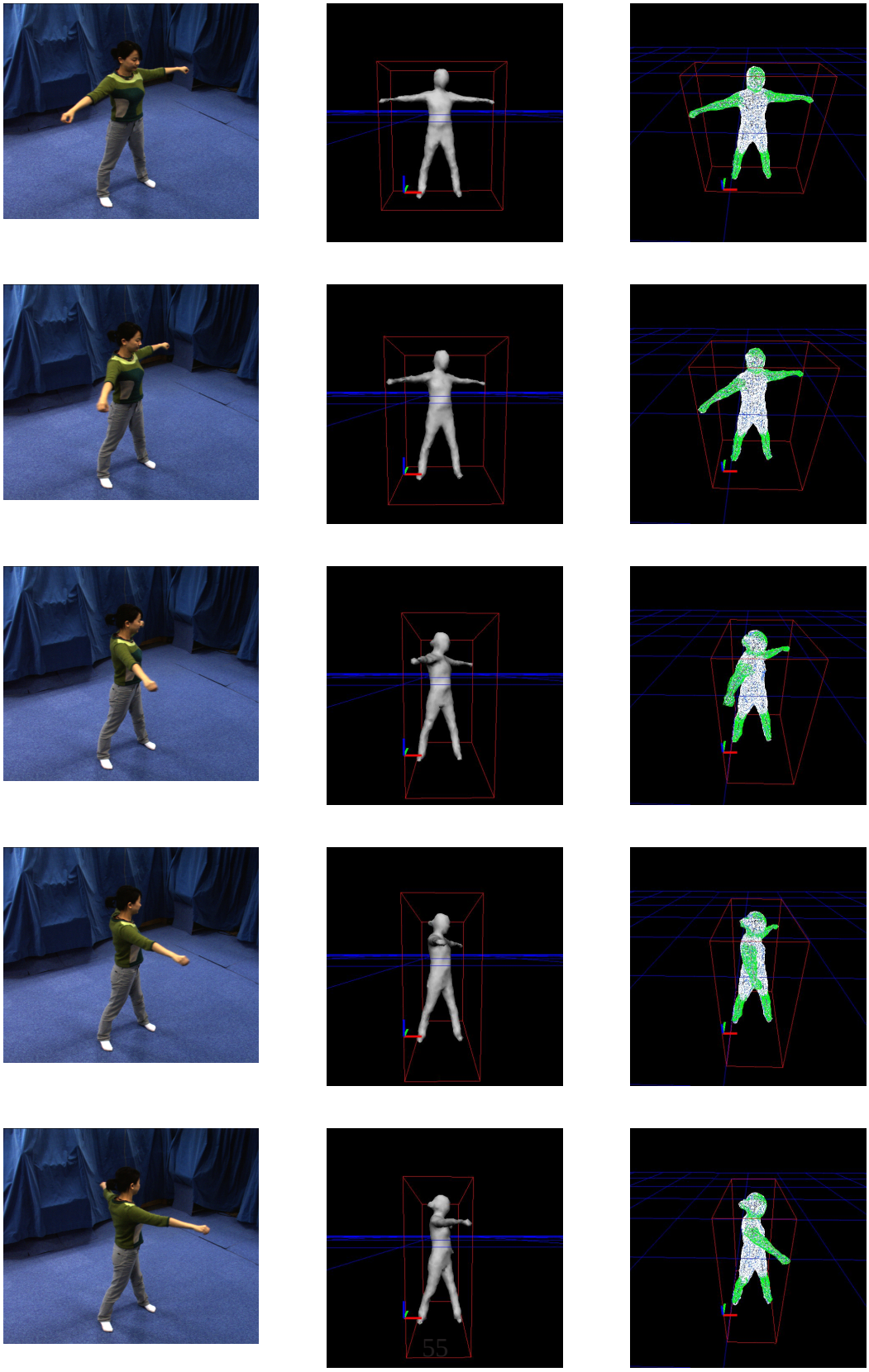


Figure 5.16: Tracking results of reconstruction data: upper body twisting

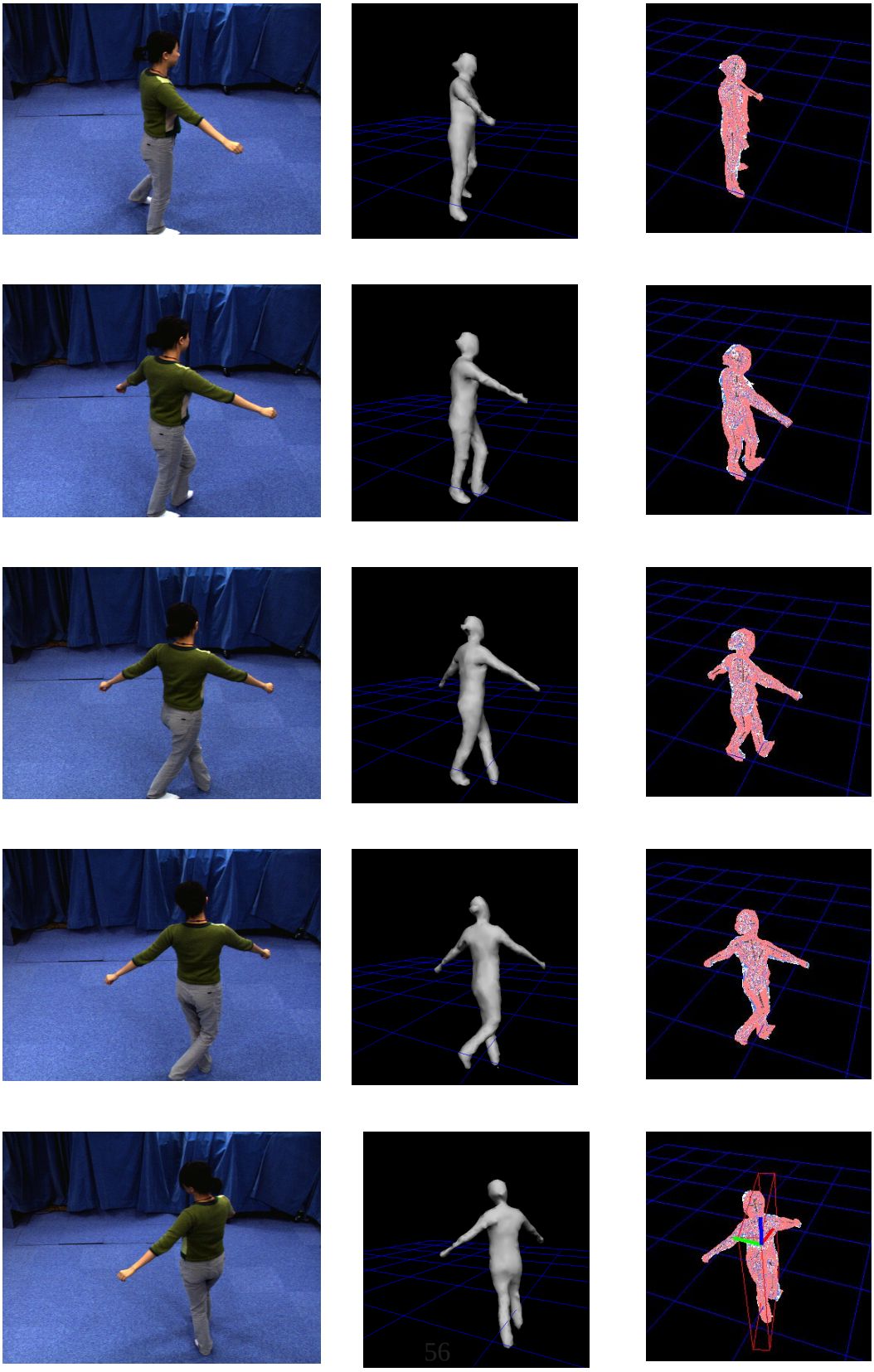


Figure 5.17: Tracking results of reconstruction data: walking

Figure 5.18: Tracking results of reconstruction data: jumping

Chapter 6

Conclusion

summary

We have presented a human body tracking system by using a deformable mesh model. There are two main steps in our research:

- 3D shape reconstruction
  During this step, we reconstruct the human body shape based on the shape-from-silhouette methods. Relatively precise reconstruction data can be obtained by adopting voxel coloring.

  The advantages of this approach are:

  - Robust estimation result
    Compared with tracking in 2D space, in which parts of the body are easily occluded depending on the relative position of the body to the camera and on the body pose, different views in 3D space could eliminate this problem. By using the photo-consistency, we can acquire more precise reconstructed data. Both of these lead to a robust tracking result.
  - Simpler analysis
    Tracking in 3D space is close to the feelings of human beings. Therefore, it is easy to locate the initial position and pose for the model in 3D space than in the 2D images. And the tracking results can be easily understood.

- Human body tracking using the deformable mesh model
  We propose the two-layered model, which acts correctly in kinematics and transforms naturally around the joint.

The advantages of this approach are:

- Easy to control
  A deformable mesh surface model linked to the kinematics skeletal model is described of articulated human body. The constraints ensure physically valid body configurations are inherent to the model, resulting in a non-redundant set of parameters.
- Robust in tracking
  Compared with the segment model, of which the meshes around the joint can not be used in tracking for its unnatural transformation or lack of the meshes around the joint, deformable mesh model leads to a better tracking result for its meshes around joints.

Future Work

For the future work, the following developments can be considered:

- Initialization of the human body model
  In our current system, how to determine the initial pose of the human body model has not been mentioned. However, tracking will fail if the model and the data are of completely different poses. Example-based learning is a solution to this problem [25]. We consider implementing this in the future.

- Improvement of the tracking results
  The captured data of the human body in this system is wearing close-fitting clothes. However, considering the case of Japanese traditional dances, clothes like the kimono present a problem to be solved. A kimono usually has wide sleeves, and the lower-body cannot be apart. The reconstruction shape is considerably different from the model, and the correct correspondence point cannot be allocated.

  We consider combining the results of marker-less motion capture system with the result of this system to refine the tracking result and to improve the accuracy of our method.

- Model generalization
  The tracking system in this study is only tested on one person. To generalize this system to different human body, to adjust the joint locations and scale the model meshes automatically is considered as a future work.

- Dance imitation by the humanoid robot
  We also consider using the motion obtained from tracking results of this system as the input to realize humanoid robot dances.

Appendix A

Camera Parameter

The 3D geometric objects are located in the world coordinate system and the camera is also located in the same world, viewing the objects. Seen from the camera, the coordinates of objects are expressed in the camera coordinate system. They are illustrated in Figure A.1

The camera is located at C and this point is named "focal point". Z_c represents the viewing direction. The transformation between world and camera coordinates can be described with the set of rotation and translation, $\langle \mathbf{R}, \mathbf{t} \rangle$. Since they represent the camera position and orientation, they are called "camera extrinsic parameters". Let a 3D point in the world coordinate be $\mathbf{x}_w = (x_w, y_w, z_w)$. Then, the coordinate of the point in the camera coordinate system, $\mathbf{x}_c = (x_c, y_c, z_c)$, is expressed as follows:

$$\begin{pmatrix} \mathbf{x}_c \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathbf{T}} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_w \\ 1 \end{pmatrix} \tag{A.1}$$

The 2D image can be obtained by projecting the camera-centered view onto the image plane (in Figure A.1). The point c at which the viewing direction and the image plane intersect, is named the "principal point", and the distance between that point c and the optical point C is called the "focal length". Let a projected point on the image plane be $\mathbf{U}$, then the projection equation can be written as follows:

$$\mathbf{u} = \mathbf{P} \begin{pmatrix} \mathbf{x}_c \\ 1 \end{pmatrix} \tag{A.2}$$

$$= \mathbf{P} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathbf{T}} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_w \\ 1 \end{pmatrix} \tag{A.3}$$

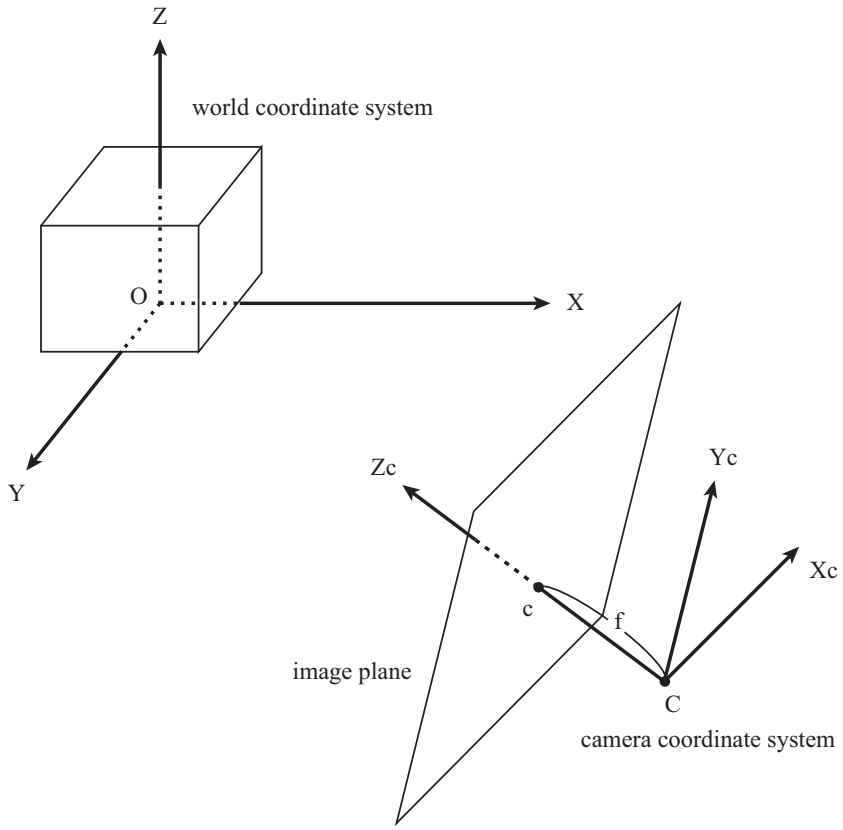


Figure A.1: The world coordinate system and the camera coordinate system

$$\text{where } \mathbf{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \ \mathbf{U} = \begin{pmatrix} \frac{u}{w} \\ \frac{v}{w} \end{pmatrix} \tag{A.4}$$

P is a 3×4 projection matrix and contains various parameters. They are called “camera intrinsic parameters”, and details are shown below.

$$\mathbf{P} = \begin{pmatrix} k_u & -k_u \cot\theta & u_0 \\ 0 & k_v / \sin\theta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{A.5}$$

They consist of the focal length, principal point, aspect ratio, and skew.

And only considering the transformation between a 3D scene point $\boldsymbol{X} = (X, Y, Z)^{\mathrm{T}}$,

into a 2D image point $\boldsymbol{x} = (u, v, 1)^{\mathrm{T}}$, the above equation can be represented as:

$$s\boldsymbol{x} = \boldsymbol{P}\boldsymbol{X} \tag{A.6}$$

$$\boldsymbol{P} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \tag{A.7}$$

In practice, besides the camera intrinsic and extrinsic parameters, lens distortions also affect the obtained 2D image. Lens distortions can be estimated by various camera calibration methods and they should be removed before any image processing.

Using the intersection points on the cube, the correponding points of the 2D image and the 3D are located. To solve the 12 unknown parameters, 6 points are enough. Least squares method

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -X_1X_{c1} & -Y_1X_{c1} & -Z_1X_{c1} \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -X_1X_{c1} & -Y_1X_{c1} & -Z_1X_{c1} \\ & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -X_nX_{cn} & -Y_nX_{cn} & -Z_nX_{cn} \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -X_nX_{cn} & -Y_nX_{cn} & -Z_nX_{cn} \end{pmatrix} \begin{pmatrix} c_{11} \\ c_{12} \\ \vdots \\ c_{32} \\ c_{33} \end{pmatrix} = \begin{pmatrix} X_{c1} \\ Y_{c1} \\ \vdots \\ X_{cn} \\ Y_{cn} \end{pmatrix} \tag{A.8}$$

$$\boldsymbol{A}\boldsymbol{c} = \boldsymbol{r} \tag{A.9}$$

with least square method,

$$\boldsymbol{P} = (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r} \tag{A.10}$$

Matrix P is calculated.

Appendix B

Quaternion Representation

In this Appendix, we explain the quaternion for the representation of the object rotation in detail.

In general, it is convenient to represent the rotation matrix and the translation vector as $\langle \mathbf{R}, \mathbf{t} \rangle$.

However, representing a rotation as the matrix form, $\mathbf{R}$, causes a great difficulty in the computation of the optimal rotation. While a rotation in 3D space has only three degrees of freedom, a rotation matrix has nine degrees. This restricts the values of $\mathbf{R}$ in a non-linear way as follows:

$$\mathbf{R}\mathbf{R}^T = I \tag{B.1}$$

$$|\mathbf{R}| = 1 \tag{B.2}$$

$\mathbf{R}$ must always satisfy these constraints to represent a rotation and this makes difficult to take advantage of the linear matrix form of rotation.

The generally accepted alternative for the representation of rotation is the use of quaternion. A quaternion is a 4-vector, consisting of a 3-vector $(u, v, w)^T$ and a scalar s, that is, $\mathbf{q} = (u, v, w, s)^T$ and it can represent an arbitrary rotation in the 3D space. It has several useful characteristics.

- The constraint of rotation is easily maintained by standard vector normalization.
- The inverse rotation is obtained by simply negating first 3 components of the quaternion vector.
- It can avoid the gimbal lock problem. Roughly speaking, the continuous change of the elements always lead to the smooth change of rotation, and vice versa.

- The intermediate rotation between two quaternions can be calculated linearly.
- With the quaternion representation, the rotation between two sets of corresponding 3D points can be solved in closed form.

Thus, the following vector is used to express the positon and the rotation of the model:

$$\mathbf{p} = (\mathbf{q}^T \ \mathbf{t}^T)^T \tag{B.3}$$

where **p** is a 7-vector, **q** is a quaternion representingthe rotation of the rigid model, and **t** is a 3-vector representing the model's position. If necessary, the form of rotation matrix is also used and the rotation matrix corresponding to **q** is denoted by **R**(**q**).

References

[1] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2, pp. 239–256, Feb. 1992.

[2] G. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette for articulated objects and its use for human body kinematics estimation and motion capture. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition(CVPR'03)*, Vol. 1.

[3] K.M. Cheung, T. Kanade, J. Bouguet, and M. Holler. A real time system for robust 3d voxel reconstruction of human motions. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition(CVPR'00)*, Vol. 2.

[4] Q. Delamarre and O. Faugeras. 3D articulated models and multiview tracking with physical forces. *International Journal of Computer Vision and Image Understanding, Special Issue on Modelling People*, Vol. 81, pp. 328–357, Mar. 2001.

[5] J. Deutscher, A. Davison, and I. Reid. Automatic partitioning of high dimensional search spaces associated with articulated body motion capture. In *IEEE Int. Conference on Computer Vision and Pattern Recognition*, 2001.

[6] Edsger. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, Vol. 1, pp. 269–271, 1959.

[7] D. M. Gavrila and L. S. Davis. 3d model-based tracking of humans in action: a multi-view approach. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 73–80, Jun. 1996.

[8] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Proc. of Computer Graphics (SIGGRAPH '93)*, pp. 19–26, NY, USA, 1993. ACM Press.

[9] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proc. Asian Conf. on Computer Vision*, Jan. 2000.

[10] E. Hunter, P. Kelly, and R. Jain. Estimation of articulated motionusing kinematically constrained mixture densities, 1997.

[11] K. Ikeuchi, Y. Takase, R. Kurazume, T.Ooishi, R. Sagawa, and K.Nishino. Modeling cultural heritage through observation. In *International Symposium on Artificial Intelligence, Robotics and Human Centered Technology for Nuclear Applications*, pp. 26–32, 2003.

[12] J.Denaviet and R.S.Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *ASME Journal of Applied Mechanics*, pp. 215–221, 1955.

[13] I. Kakadiaris and D. Metaxas. 3d human body model acquisition from multiple views. In *Proc. of the 5th International Conf. on Computer Vision*, pp. 618–623, Jun. 1995.

[14] I. Kakadiaris and D. Metaxas. Model based estimation of 3d human motion with occlusion based on active multi-viewpoint selection. In *IEEE Transaction on Computer Vision and Pattern Recognition (CVPR'96)*, pp. 81–87, Jun. 1996.

[15] R. Kehl, M. Bray, and L. V. Gool. Full body tracking from multiple views using stochastic sampling. In *Proc. IEEE Inter. Conf. on Computer Vision and Pattern Recognition*, pp. 129–136, Jun. 2005.

[16] K. N. Kutulakos and S. M. Seitz. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision, Marr Prize Special Issue*, Vol. 38, No. 3, pp. 199–218, July. 2000.

[17] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, Vol. 21, No. 4, pp. 163–169, Jul. 1987.

[18] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, Vol. 194, No. 2, pp. 283–287, Oct. 1976.

[19] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi. Generating whole body motions for a biped humandid robot from captured human dances. In *IEEE 2003 International Conference on Robotics and Automation*, Sep. 2003.

[20] S. Nakaoka, A. Nakazawa, K. Yokoi, and K. Ikeuchi. Leg motion primitives for a dancing humanoid robot. In *IEEE 2004 International Conference on Robotics and Automation*, Apr. 2004.

[21] K. Ogawara, K. Hashimoto, J. Takamatsu, and K. Ikeuchi. Grasp recognition using a 3D articulated model and infrared images. In *Proc. of The 2003 IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS2003)*, Jan. 2003.

[22] K. Ogawara, J. Takamatsu, H. Kimura, and K. Ikeuchi. Extraction of fine motion through multiple observations of human demonstration by dp matching and combined template matching. In *10th IEEE Int. Workshop on Robot and Human Communication(ROMAN)*, pp. 8–13, 2001.

[23] Michael Potmesil. Generating octree models of 3d objects from their silhouettes in a sequence of images. *Comput. Vision Graph. Image Process.*, Vol. 40, No. 1, pp. 1–29, 1987.

[24] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc. of 5th International Conf. on Computer Vision*, pp. 612–617, Cambridge, UK, Jun. 1995.

[25] R. Rosales and S. Sclaroff. Combining generative and discriminative models in a framework for articulated pose estimation. *International Journal of Computer Vision*, 2005.

[26] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, Vol. 35, No. 2, pp. 151–173, July. 1999.

[27] Mark R. Stevens, W. Bruce Culbertson, and Thomas Malzbender. A histogram-based color consistency test for voxel coloring. In *ICPR (4)*, pp. 118–121, 2002.

[28] VICON. motion capture system. www.vicon.com.

[29] Mark D. Wheeler and K. Ikeuchi. Sensor modeling, probabilistic hypothesis generation, and robust localization for object recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 17, No. 3, pp. 252–265, 1995.

[30] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 780–785, 1997.

[31] 剣持雪子, 小谷一孔, 井宮淳. 点の連結性を考慮したマーチング・キューブ法. 電子情報通信学会パターン認識・メディア理解研究会, PRMU, pp. 197–204, Jan. 1999.

[32] Y.Yang, A. Yuille, and J. Lu. Local, global, and multilevel stereo matching. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 274–279, Jun. 1993.