TOWARDS A MODEL-BASED 3D MARKER-LESS HUMAN MOTION CAPTURE

QUAH CHEE KWANG

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2008

Acknowledgements

There are many people without whom this work would not have been possible. I would like to thank my research advisors from NTU and INRIA, Prof Seah Hock Soon and Prof Andre Gagalowicz respectively. I am very grateful to Prof Gagalowicz for his guidance and allowing me to benefit from his knowledge and experience. I am very thankful to Prof Seah Hock Soon for his many advices, nurturing and support throughout the duration of my graduate study.

I would also like to thank the NTU graduate research administration for supporting this study. I am also very thankful to INRIA for the research support.

Thanks also to my fellow lab-colleagues: Wu ZhongKe, Qian KeMao, Stephanus Tanjung, Andyardja Weliamto, Li Li and Xiao Xian for their useful idea exchanges.

Thanks also to Richard Roussel, Philippe Gerard, colleagues and friends of MIRAGE-INRIA who gave numerous suggestions and software exchanges.

I would also like to thank members of CAMTECH of NTU, especially Gerrit Voss for the useful knowledge exchanges and Jochem Quick who provides the system support.

Thanks also to Ta Huynh Duy Nguyen and Lee Shang Ping of IERC of NTU for the numerous idea exchanges and system supports.

Finally I would like to express my deepest gratitude to my family members, especially to my wife Amy, for their encouragements and supports.

Contents

List of Figures

List of Tables

Abstract

This research proposes a novel framework for capturing 3D human motion from video images using a model-based approach. Existing commercial motion capture methods that place markers onto the human will cause hindrance to the human performers. Our approach does not require any markers. Although many systems reported in the existing research literature do not use markers, there are still many problems related to marker-less human motion capture.

Our contributions consisted of the two main phases: (1) to construct a 3D human puppet model that is very similar to the subject, and (2) to follow the motion of the subject using this 3D model. The human model and movements have to be accurate so that we can obtain quantitative data for applications such as bio-mechanical analysis. A substantial amount of work has been emphasized on building the 3D human model, as the accuracy and reliability of the motion tracking depend very much on it.

The reconstruction of 3D human model is facilitated by a generic geometrical human model consisting of the external skin and its internal skeleton. The output is an accurate external skin of the subject with its estimated internal skeleton. This approach uses several cameras and does not require prior camera calibration. First, the camera calibration and 3D reconstruction take place simultaneously to produce the intermediate 3D model once the characteristic points between the generic model and the real one are registered. Then, we automatically matched the silhouette curves of the intermediate model and the real subject to yield a better 3D human model. Our setup requires no prior calibration and moderate

human interaction, its operation is simple, cheap and efficient as compared to the existing 3D laser body scanners and computer imaging methods.

Our human motion tracking algorithm starts by the automatic learning of the colour/texture onto the puppet model from its initial pre-positioned posture. Then, our computation will synthesize the 3D puppet movements such that it minimizes the image differences between the synthesized movements and real athlete's motion. This is realized by using a simulated annealing algorithm to search iterative for the optimal posture represented by the joint kinematics with the various degrees of freedom. The joint kinematics then drive the skin of the model puppet to produce the synthesized image, which will be used to compare with the real image. The colour texturing onto the puppet is updated and learned once every few frames so that the synthesis is not influenced by the changing articulated posture and illumination variations. The image rendering for the motion synthesis is the most computationally intensive module and it is sped up by using a graphics processor unit (GPU). With our results, we demonstrated that we are able to track the motion of the arms, which are usually highly articulated and quantitatively small in the images. The advantages of our method are: (1) it does not require image segmentation, (2) it copes with occlusion, and (3) it is able to operate in highly cluttered environments.

Chapter 1.

Introduction

Documented work for motion capture and analysis started as early as the 19th century, when Eadweard Muybridge [168] began photographing horses to analyze their movement. At that time a French physiologist Etienne-Jules Marey also made his chronophotography[1] work in studying the human actions filmed in videos. The importance of motion capture (mocap in short), is motivated by its applications over a wide spectrum of areas. Tracking and following the movement of human joints over an image sequence, and recovering the 3D body posture and kinematics are especially useful for the study of the human locomotion and bio-mechanical applications [30], such as gait analysis, injury prevention, rehabilitation, sports performance enhancements, etc.

Applications of motion capture in other domains include 3D animation, augmented reality, telepresence [46], human factor design, free-viewpoint video and virtualized reality [23], [70], post-production [47], etc. The kind of setups, methods and technologies that are used for motion capture are determined by their respective operational needs.

There are many technologies developed to capture the human motion. They range from magnetic, mechanical and optical systems, for which the subject needs to wear sensors and markers on the body, to the non-intrusive one which is based purely on using video

[1] Chronophotography is an application of the study of movement (science), and photography (art).

cameras. The operating environments can vary from the well controlled indoor environment to the highly cluttered outdoor scenes.

While the technologies for sensing, photography and imaging have changed and evolved over the centuries, improvement in the accuracy and reliability in acquisition of human movement is still a problem. The human movements are highly complex since they have many degree-of-freedoms (DOFs) and the skin is also deformable, thus they still pose many challenges to generally process them in general.

1.1. Objectives, Motivations, Challenges & Scope of Research

In this research, we focused on capturing 3D human motion that can be applied to bio-mechanical analysis. The extraction of information from the raw sensor-data needs to be accurate so that they can be used for analysis purpose e.g. comparing gait efficiencies of different athletes.

The operating environment can eventually be in a cluttered scene. The non-intrusive factor is also very important so that it can be applicable to scenarios in sports competitions. This implies that existing commercial motion capture methods that require the subject to wear sensors are not suitable, hence constraining us to just video cameras. We may also have to face the situations when minimal number of cameras can be used due to constriction and limited resources. This prompted us to explore the methods that are associated with the computer vision/computer graphics collaboration to accomplish our task.

Computer vision largely deals with the analysis of pictures in order to emulate the human perception e.g. understanding, recognition and tracking of objects in the visual

scenario by inferring 3D information from one or more images. However, after many years of research, the machine vision is still far from emulating its human counterpart. Machine vision systems generally face severe problem due to their lack of understanding of the 3D scene, illumination variation, occlusions and in addition, the human has complex kinematics and deformable skin e.g. stretching of skin and bulging. These make image segmentation and feature extraction, which are common entry points to their system, not being able to yield consistent and reliable information for subsequent analysis purpose. Some researchers even term these as the semantic gap [37].

On the other hand, computer graphics start from 3D scene models and through the use of synthesis techniques produces the pictures. However, the synthesized images lack realism. In general, computer graphics do not know how to fully simulate a real scene in an automatic way.

In this dissertation, our algorithms are built on the concept of collaboration between computer vision and computer graphics to tackle our problems. This concept had been proposed as early as in 1990 [44], and is gaining popularity [82], [143]. This approach had been applied to applications such as modeling complex indoor scenes [45] and rigid object tracking [48]. Computer graphics will provide the knowledge-base for the 3D modeling, animation and synthesizing, whereas, computer vision will supply the learning and analysis capabilities.

In recent years, research had shown that the quality of the human tracking results relies heavily on the similarity between the 3D puppet model and the real subject. This motivated us to start our research with 3D reconstruction of the human subject. Our customized 3D

human model needs to be accurate and has an external skin and its skeleton. The difficulties in building this 3D human model is reviewed in Chapter 2.

In order to acquire the correct kinematics of 3D motion, the 3D model puppet has to be animated correctly to generate the visual information that is compatible with the human motion that appears in the real images at each time step. The main difficulties are: illumination variation, occlusion, large search space due to complex human motion and deformable human skin. Moreover, the subject may be wearing clothes that flicker, which further adds to the difficulties.

In this dissertation, we assume that the subject is within the field-of-view of the filming cameras. A subject wearing skin-tight clothes is required. It is also assumed that the skin deformation can be neglected, although we will discuss it in this thesis. In this dissertation, rigid skin transformation is used. The computational speed is another factor when considering the large search space. However, the motion tracking algorithm can be executed in an offline process, since this is permissible in the prototyping phase, as long as the computational time is keep within a reasonable duration. We consider the tracking of the facial expressions and finger motions as different problems that are outside the scope of this research.

1.2. Summary of Contributions

The major contributions of this dissertation are as follow:

1) A novel method is proposed to accurately construct a 3D human model from several un-calibrated wide-baseline images by using characteristic points and limb silhouettes to deform a 3D generic model. Our setup requires no prior calibration and moderate human

interaction, thus the operation is simple, cheap and efficient. The final output is a customized 3D puppet model of the subject consisting of an external skin and its skeleton. The results show that our algorithm can be adapted to human subjects with different sizes and shapes.

2) In the building of the 3D model, we realized a new method for camera pose estimation and reconstruction of a 3D human model at the same time by using characteristic points on a generic model and their respective correspondent in the wide baseline 2D images. No prior camera calibration is needed. The subject himself will be the calibration object, with the surface feature points of the generic model adapted to him. The sparsely distributed reconstructed feature points are fed into the radial basis function to deform the generic model. This stage yields the intermediate 3D model. We show that this method is able to cope with shape, size and feature correspondent uncertainties.

3) A method that refines the construction of a 3D human model from the matching of silhouette curves is proposed. This is realized by automatically registering the differences between the silhouette curve from the real images and the intermediate 3D model (from 2 above). This module results in an improvement over the human reconstruction approach that uses only feature points.

4) A new 3D model-based method for capturing the 3D human motion using simulated annealing with GPU acceleration is presented. In our set-up, we do not assume that prior motion database is available for producing generative and discriminative model through training examples. Our algorithm is able to automatically track the motion of human arms, which have an additional degree-of-freedoms from the rigid body. Generalization to the case of complete human body movement could be easily extended. We are able

follow the trajectory of the arms which usually appear quantitatively small in the images. We animate the 3D puppet model, built earlier, with its colour texture to synthesize the information for matching with the real images.

1.3. Organization of the Dissertation

The rest of this thesis is structured as follows:

- We start in Chapter 2 by presenting the background, review the related work and the respective pros and cons of their methods.
- In Chapter 3, we give an overview of the overall framework and workflow for model-based human motion capture. The setups and fundamental concepts employed for each section of the framework are presented.
- Chapter 4 presents a method for camera pose estimation and constructing a 3D human model at the same time by using characteristic points. Output from this stage is referred as the intermediate human model. We investigate the accuracy of the results by making comparison with reference to prior calibrated data.
- Chapter 5 explains in detail the method that we employed for the extraction of silhouettes from both the intermediate 3D puppet and the real images.
- Then in Chapter 6, we detailed the curve matching algorithm that registers the silhouettes of the 3D puppet with their counterpart in the real images. The differences in the matching are used to compute the deformations for the external skin and its internal skeleton to their proper locations, which results in a customized 3D human puppet model of the subject.

- In Chapter 7, we make use of the 3D human puppet model that we have constructed to track the human movement. A model-based analysis-by-synthesis method is presented with qualitative and quantitative results.
- Finally, in Chapter 8 we conclude with a summary of our work, and then describe the possible future extensions.

Chapter 2.

Background and Reviews

This chapter surveys the state of the art methods that are related to the model-based motion capture framework. We also give some background of the technologies that are relevant to our research. These include existing motion capture methods, 3D model reconstruction, and important issues such as skin deformation of human when undergoing motion and estimation of human skeleton and joints. The reviews on related works are concentrated on the general aspect such as their fundamental functionality structures and main concepts. We will also discuss the problematic issues related to image segmentation and feature extraction.

2.1. Commercial Motion Capture

Existing motion capture (mocap) devices fall into 3 main categories: magnetic, mechanical and optical systems. The main providers for magnetic systems are Ascension [164] and Polhemus [172]. The latter provides mechanical systems as well. The main manufacturers of optical mocap equipment are Motion Analysis [169], Peak Performance [171], Qualisys [173] and Vicon [175].

Magnetic motion capture systems utilize sensors placed on the body to measure the low-frequency magnetic field generated by a transmitter source. The sensors and source are cabled to an electronic control unit that compiles their reported location within the

measurement field. The electronic control unit is networked with a host computer that represents these positions and rotations in the 3D space.

Mechanical motion capture systems require the performer to wear a mechanical armature fitted to their body. The fitting is usually done with elastic straps and belts, which hold plastic plates against the body of the performer. The sensors in a mechanical armature are usually variable resistance potentiometers or digital shaft encoders. These devices encode the rotation of a shaft as a varying voltage (potentiometer) or directly as digital value.

Existing commercial optical motion capture systems utilize reflective or pulse-LED (infra-red) markers attached to joints of the athletes' body. Multiple infra-red cameras are used to track the markers to obtain the movement of the subject. Currently, this kind of system is mainly used for biomechanical analysis.

All motion capture systems have one common disadvantage, requiring the subject to wear some kind of devices or markers on the body. This will be a hindrance to the true movement of the performer. In addition, this kind of acquisition setup is only suitable for a very well controlled (mainly indoor) environment, which is, unrealistic and impossible during a sport tournament for example. Furthermore, post-processing of motion capture data is tedious and time consuming e.g. re-establishing correspondents due to sensors confusion in optical sensors.

2.2. Computer Vision-based Motion Capture

Computer vision-based approaches for motion capturing, with their potential to operate in natural environments, had started to catch the attention of industrial mocap manufacturer

[170], [175]. Large numbers of works on tracking and analysis of human motion using computer vision techniques had been proposed over the years. A couple of reviews on computer vision-based motion capturing methods had been done, [2], [50], in the years 1994 and 1999. In a more recent review in 2006, [97] extended to their earlier reviews [96] of year 2001. Many vision-based mocap techniques had been proposed to tackle different application requirements. In systems that require accuracy and reliability, some kinds of human model are usually used to facilitate the tracking [83], [141]. Whereas in applications for visual surveillance and activities monitoring [89], which require speed and robustness, their systems usually heavily involve low-level vision techniques such as image segmentation, thresholding, statistical and probabilistic formulations.

2.2.1. Typical Vision-based Mocap Framework

The inputs to all vision-based systems are images acquired from electro-optic sensors e.g. video cameras, infra-red cameras, etc. Electro-optic devices are popular largely due to their non-intrusive and passive nature. Moreover, they are easy to set up and their prices are cheap. To deal with occlusions and kinematics singularities [123], multi-cameras are usually used, although single camera setup may be used due to constriction and limitation of resources.

In the computer vision-based mocap survey by [96], tracking is defined as establishing coherent relation of subject and/or landmarks between frames. Pose estimation of subject is the process of identifying how the subject is configured in the scene e.g. posture of a human described by its kinematics at an instantaneous time. For 3D model-based motion capture approaches, the tracking and pose estimation are usually closely coupled with each other. The notion of motion tracking is used differently and loosely defined throughout the

literature of visual analysis of human. In this dissertation, since in our framework tracking and pose estimation are tightly coupled, and our aim is to quantify the human posture at each instantaneous time and follow it over a time sequence, the overall process refers to 3D human motion capture or 3D human motion acquisition.

Nearly every vision-based mocap follow the steps: (1) segmentation of subject from the rest of the image, (2) these segmented images are transformed into some kinds of higher level representation to suit a particular tracking algorithm, and (3) how the subject should be tracked from one frame to the next.

In these kinds of frameworks, many proposed algorithms had relied heavily on the image segmentation, which is a very crucial part of the system, and modeling of its result. In addition, some assumptions regarding the background scene had been used e.g. constant and low cluttered background. The information that is presented for segmentation can be either spatial or temporal image data. The main methods for segmentation are: edge detection/filtering, image subtraction [75], [140], color segmentation [95], blob segmentation [107], optical or motion flow [18], labelling via graph-cut [16], etc. Also in [102], they introduced 3D context awareness to extract the silhouette from multi-view images. The common higher-level representations that are derived from the segmented images will be in the form of: feature points, silhouette, bounding box, blobs, motion flow fields, texture and edges. It is also a common practice to combine various representations to be used for tracking.

The use of silhouette is one popular approach [12], [25], [92] [128], where 3D volume data of human is built from multiple 2D silhouettes to yield the human posture that is tracked at each instantaneous time sample. This kind of approach is commonly known as

the shape-from-silhouette (also called visual hull) method. It relies on calibrated still cameras, and that the subject can be segmented from the background by assuming constant and low cluttered background. The visual hull method is also used for 3D reconstruction of models and is discussed in more details later in Section 2.4.2 of this chapter.

Various motion tracking algorithms, frameworks and steps were proposed based on the visual hull approach. In [25], they extended their temporal shape-from-silhouette technique [26] to track the articulated rigid segments through the alignment of multiple-views silhouette poses across time. Then in [73], the authors proposed a *stochastic meta descent* minimization algorithm to fit a human model made up of super-ellipsoid parts to the volumetric data. In [101], the authors used a deformable mesh model with 'repulsive force' to guide the inter-frame 3D volumetric deformation and at the same time detecting collision. Then in [92], they estimate the human movement parameters by using the extended Kalman filter after utilizing the Bayesian network to estimate the body part sizes from 3D voxel data. Also in [23], they adapt an articulated 3D human model to the multi-view silhouette via Powell's minimization.

Natural 2D image feature points had been used for rigid object tracking [136]. However, in the work of [60] that put distinctive optical markers on the subject they found out that without a prior model, 50 percent of the markers fail to find the correct correspondence after less than 20 frames. It must be noted that even with a single correspondence outlier could cause the mocap system to drift toward catastrophic failure.

Representation such as blobs [41], [156], and bounding boxes [78] have been introduced to stabilize tracking. These kinds of representations are obtained by segmentation and by labeling the image regions using similarity characteristics e.g.

subspace coefficients via DCT [104], similar color [93], optical or visual motion flow [17]. However, segmentation algorithms perform very poorly in cluttered and noisy scenes as they are ill-posed problems. Without a proper 3D articulated model, tracking error is accumulated over time and this leads to drift and then eventually to divergence.

To facilitate tracking and pose estimation, a prior model or example data is usually incorporated. The pose of the prior model is matched with the one represented in the real images. The prior model may be used in the form of: (1) 3D animatable geometric model [55], [69], (2) motion and dynamic model [157], [91] (3) learning or training and representation of example poses [42], [126], using approaches such as the principal component analysis (PCA), Bayesian network [109] or discriminative learning [76], (4) parametric deformable models e.g. active contours [9], or (5) combination of the various modeling methods [106]. To reduce the search space, prediction such as the Kalman filter and CONDENSATION [66] algorithm might be utilized. Prediction is a useful mean to reduce the search space during the matching of poses. Numerical optimization methods such as the gradient decent, Gauss-Newton are commonly used to minimize the error between the prediction and observation functions. Last but not least, stochastic search using the particle filter had also been suggested [34].

In the work of [125], a motion model of walk was used for tracking. In [83], the subject wore a tight texture pattern used for tracking (based on texture learning), pose estimation and Kalman filter prediction. In [1], they track the human motion by combining classifiers tuned from example-based human motion. In the work of [106] they combine view-based and model-based methods. In [79], they parameterize hidden Markov temporal models with 2D spatial views of body parts. Then in [150], the authors proposed hierarchical

interpretation of action represented by *grammars*. In [35], they use the iterative closest point to fit and register the segmented depth data from stereo images with a 3D model. In work of [77], they model the human as articulated chain of 3D cones, and tracking is done by minimizing the Chamfer distances between the synthesized and real contour. Chamfer distance is also a similarity measurement in [49], where they model the subjects with super-quadrics used for human pose estimation that employed graph-based searching strategy. Among these methods, parameterization of the articulated motion kinematics constraints may be used to confine the solution [19], [85], [138].

2.2.2. Use of 3D Model

In applications that require accurate and reliable measurements, good 3D models are always required. Experiences from prior works [52], [54] and [68] had indicated that the process of tracking is very sensitive to the shape parameters used. Therefore, it is inappropriate to use, for example, a generic "averaging human" model for accurate and precise tracking of human with a different shape and size. However, it is noted that good and comprehensive 3D models are difficult and take time to build. Therefore, in the uncontrolled surveillance applications, good 3D models may not be available. All existing methods that use 3D model apply the analysis-by-synthesis methodology one way or another.

The 3D human model that has been used consisted of the main body parts and structure needed to animate the articulations i.e. torso, arms, legs and head. The 3D model is normally driven by a series of kinematics chain relating the different body parts, propagating from the parent nodes to their children nodes for forward kinematics or the

reverse for solving inverse kinematics. The 3D model is also useful for preventing unreasonable synthesis e.g. knee bending must be between 0^o to 100^o.

In the work of [55], they adapt the H-Anim human model to the specific subject and used it for tracking by combining the iterative closest point algorithm with Levenberg-Marquart optimization. In [141] they construct the super-quadrics model of the subject with the aid of 3D scanner, and then apply the model to perform tracking from hybrids of segmented information. In the work of [110], they use the H-Anim model for human tracking as well. Also, in the work of [147], a 3D graphical human model is used. Other approaches like [56] attempt to match the stick-figure model with mathematical skeletons (see later) extracted from images. In [111], the 3D human model and motion are adapted simultaneously through a parameterized state vectors during the process of tracking, however this means that the geometric model and the motion will go through a transient state before they converge, thus the model or motion can drop to local minimum.

2.2.3. Applications and Evaluation of Performance

The algorithms, technologies and equipment setups that had been used for motion capture are driven by the following main application areas: surveillance, interaction/control, and analysis. The types of motion capture applications determine these main performance requirements, in terms of robustness, speed and accuracy. For example, in human motion analysis applications accuracy is very important and a certain amount of robustness for tracking is required so that it can cope with cluttered environments. On the other hand, for the interactive/control applications e.g. interface to computer games and augmented reality, the processing speed is the most essential. For surveillance application, robustness is the most crucial so that the system can deal with all possible scenarios.

There are a few ways to evaluate the performance of the technology for acquiring human motion, which is subjected to the context of application:

Quantitative tests rely on comparing the ground truth data with the estimated ones to evaluate the accuracy and precision. This kind of test is important if the technology is meant for analysis applications. Several ways can be used to estimate the ground truth data: (1) manual segment and corresponded data, (2) benchmarking to established commercial devices. The former is time consuming and tedious, whereas the latter may be expensive to obtain.

Qualitative tests rely on visual inspection and is the most widely used. The most common approach is by back-projecting the estimated 3D motion to overlay onto the real video images and compute the error using e.g. sum-of-square difference (SSD), normalized correlation, etc. This way of assessment is normally sufficient for surveillance and interactive/control applications.

2.3. Issue of Human Skinning

Human undergoing motion will produce skin deformation, usually termed as skinning. The skinning issue has been studied in computer graphics; however it had not been properly reviewed in relation to any existing vision-based motion capture. Skinning is necessary for the geometric properties and color texture on the human skin to be warped correctly when undergoing motion. Some examples for the human skin deformation are: (1) bending of elbow causes the skins to stretch, (2) wrinkle of skin, (3) 180 degrees twisting of wrist causes the skin near the wrist twist to about 180 degrees, whereas the skin near the elbow is twisted about 5 degrees. Other kind of deformations can arise from bulging of muscle,

bending of body, etc. All these deformation factors are anatomically related; therefore the anatomical structure of 3D model must be able to mimic the subject closely.

One of the simplest and fastest ways to animate skinning is to consider a rigid skin model that is attached onto the articulation of the respective bones and ignore the deformations near the joints. However, rigid skinning does not look realistic near the joints. The deformation of human skin is highly complex, by itself, it has attracted many research interests over the years, and mimicking the real human skin still remains a challenge. Skinning has been an actively researched area in computer graphics. In this section, we will review some of the common skinning methods, which can be categorized into (1) example/training based, (2) parametric based, and (3) anatomical/physical based methods. Every of these techniques was proposed to fulfill the demands of its respective application such as real-time requirement and realism when the skin deforms.

2.3.1. Skinning from Examples

This kind of method requires training data. Examples of this kind of method reported in the existing literature are [3], [72], [84] and [137]. Their key objective is to generate smooth visual realistic movement via key postures. Their procedure normally involved:

1. Obtaining training data from scanning or interactively sculpted key postures.
2. Using these training data to minimize a set of weighted blending function parameters using e.g. least-square minimization and radial basis function.

The resultant blending function parameters will be an interpolated or extrapolated of movement with respect to the subject's kinematics. The deformable objects for these methods are usually represented using triangle meshes. The synthesizing and rendering of

example-based skinning is fairly manageable with the modern computers and commonly used in computer gaming. These methods in general do not preserve the volume of the object undergoing deformation.

This kind of method had its disadvantages:

1. Needing a skill artist to sculpt all the key postures so that we can cover the whole range of movements.
2. The process of posture sculpting is very time consuming.
3. All the vertices at the joint will collapse (e.g. in the situation when the human forearm twist towards 180°, fig. 2.1) if the range of movement is not well represented. Also, the *sunken* elbow problem is undesirable.
4. Difficulties in retargeting the same blending function onto another subject of different shape and size.

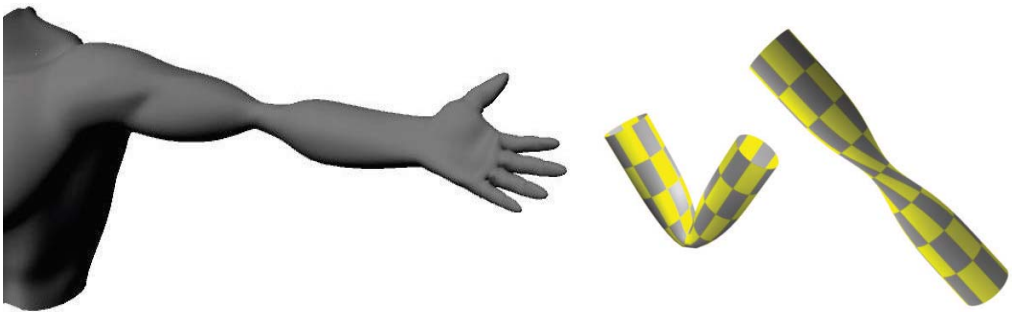


Figure 2.1. Skin collapses when the range of movement is not properly represented e.g. the twisting motion

There are also research done to acquire the skin deformation. In [131], they attempt to estimate the skin deformation from silhouette, and in [32], they capture the skin deformation in a controlled environment with the subject wearing a structured garment.

Also, in [108], they use the optical marker system and some human interaction for data preparation to capture the human skinning.

2.3.2. Parametric Skinning

Parametric surface such as the Bezier, spline or cross-section surface, representing smooth surfaces with relative small number of control points, may model the surface skin e.g. [20], [135]. In general, manipulating its control points deforms the parametric surface. This method does not require training via example data or the complexity of modeling the physical properties of the human tissue. In the work of [65], the authors introduced the control of sweep surfaces from 3D ellipses that model the human body while keeping constant the volume of the object undergoing deformation. However, its main limitation is the difficulty to manipulate objects with branch joints such as the human shoulders.

2.3.3. Physical and Anatomical-based Skinning

This type of approach attempts to take into account the physical properties (not biologically related) of the human tissue. In [112], they used metaballs to model the body tissues. Other methods include volumetric object to model the muscles, which in turn deformed the skin [5], [132], [153]. Although it is always assumed that the physical-based approach would be the most accurate, however, the musculature structure varies between different individuals. Moreover, to extract the human anatomical properties and modelling them are extremely difficult. The drawbacks of these methods are (1) the requirement of a skillful operator to construct these complex objects binding the skeleton to the skin, and (2) very intensive computations needed to synthesize and manipulate the musculature objects.

2.4. 3D Human Reconstruction

There have been numerous works done using vision-based system to construct the 3D human model. They usually construct the surface skin of the model, and their techniques fall into 2 categories: (1) 3D scanner-based system, and (2) passive multi-camera system.

2.4.1. 3D Scanner-based and Active Systems

The scanner-based and active systems are available from many commercial companies. There are many products that provide full 3D body scanning using laser scanners e.g. [166], [167], [174]. The work of [31], [159] that used structured light and projector are classified under this family of approach. Typically, they use the triangulation principle, and with laser light or pattern projection method and a CCD camera. Body scanners usually capture the shape of the entire human body in about 15 to 20 seconds. The 3D surface mesh of the subject could be obtained at the resolution of about 1 to 2mm. A summary by [124] gives a brief overview of some 3D body scanners in table 2.1.

Table 2.1 Some of the commercial body scanning products and their features

Companies	Cyberware [166]	$(TC)^2$ image twin [174]	Vitronic [176]	Hamamatsu [167]	Wicks & Wilson [177]
Products	WB4, WBX	2T4s	Vitus	64 B Scanner	TriForm BS
Time (sec.)	~17	~12	<20	<16	<12
Accuracy (mm)	~1	~1	~1	~1.5	~2
Technology	Laser line	Structured light	Laser line	Structured light	Structured light
Point density (mm)	3×3	2.8×2.5		~5×5	

All of these commercial tools are rather expensive; they cost few thousands US$ e.g. a whole body scanning system by Cyberware is priced at about US$300,000. We also noticed that this kind of acquisition method requires the subject to stay still and rigid for the whole duration of scanning, about 15 seconds for full body coverage, which is quite

constrictive and not very practical in certain application. We have to take note that some laser scanned data may require the user to register a few key landmarks manually before automatic processing is performed to stitch up the multiple range images. In addition, scanner-based methods only give the surface of the model and do not contain any information on the skeleton.

2.4.2. Passive Multi-view Systems

The multi-view approach is much cheaper; also video cameras are more easily available and their set-ups are much simpler. Since video images are 2D, the pinhole camera model is always assumed. All multi-view 3D reconstruction methods need the camera poses and orientation to be known. These parameters may be obtained from the calibration of cameras realized beforehand or executed on the run by using structure-from-motion self-calibration methods [86], [114] and [115]. The most common approach usually begins with a similar fashion as with the motion capturing algorithm i.e. image processing or segmentation is carried out to extract natural characteristic features from the images: corners, feature points, edges/lines, regions, silhouettes, etc. Then the features that appear in the respective images are tracked to establish point matches for camera calibration on the run, and 3D reconstruction when calibration is available.

Shape-from-silhouette method is a very common approach to reconstruct human models using 2D silhouettes from multi-views, due to the simplicity for silhouette segmentation and representation convenience. In [152], the human model is built using silhouette-based volumetric reconstruction from multiple calibrated cameras, whereas in [139], they construct the subject by mapping 2D silhouette and 3D surface features from multiple cameras to deform a generic humanoid model. Then in [68], they propose a

method to construct the specific parts of the human subject e.g. arms, legs, from orthogonal camera views via shape motion segmentation by having the human subject undergoing pre-defined movement. Also in [88], they reconstruct 3D object that varies with time by putting together the visual hull approach and a 3D dynamic model similar to the active contour. All these shape-from-silhouette approaches require (1) the subject to be segmented from their image backgrounds, (2) cameras to be calibrated a prior.

The theory of visual hull for general 3D object reconstruction had been reported in [81], for smooth curved objects, [13], [15] for un-calibrated cameras or unknown viewpoints. Also in [40], they perform 3D reconstruction by combining shape from silhouette and hypothesis voxel removal to obtain the 3D surface of the object. The theoretical limitation of visual hull had been presented in [80]. We must take note that the resultant 3D model will be 'blocky' if there is insufficient view coverage. Figure 2.2 shows the cone of visual hull using a 6-cameras set-up in our human model reconstruction. Figure 2.3 displays the undesirable blocky effects especially for the human arms.

Another recent approach for 3D human reconstruction [124] extended the approach of [114] using feature extraction and tracking, matching, self-calibration and dense point correspondences, to recover the point cloud representing the subject. Their method requires the subject to remain still for about 40 seconds during the video capture in order to obtain the coverage of the whole body. Their results showed some limitations when completing the regular surface model from the point cloud e.g. requiring the semi-automatic closure of holes to clean up the model. Closely related to this method is to first track the feature tokens in video streaming, and then apply the *factorization* method [27], [113], [148], or recursive extended Kalman filter estimation [6], [158] to recover the motion and shape of

rigid objects. Also in many early works e.g. [43], [58], they attempted to recover structure and motion from perspective views parameterized by the projective geometry assuming that the correspondents could be pinpointed. After the shape and motion had been computed, with these methods, bundle adjustment [145] may be utilized to refine the object shapes and camera poses, and in [134] aided by a 3D geometric model. However, these methods are very constrictive for the subject, and more importantly (1) feature points alone are too sparse to make up the human, and (2) reliability of feature points tracking suffers severely from foreshortening and occlusion limitations.

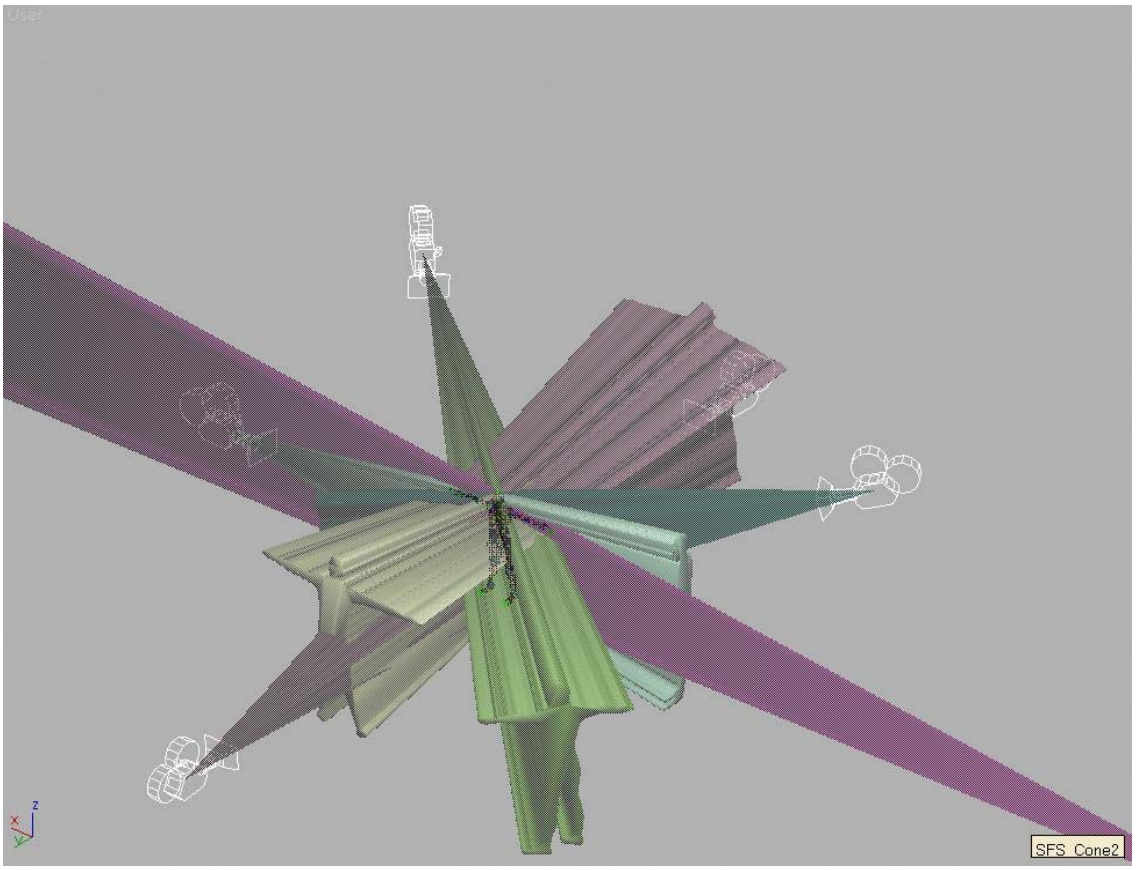


Figure 2.2. Cone of the visual hull method using 6 cameras

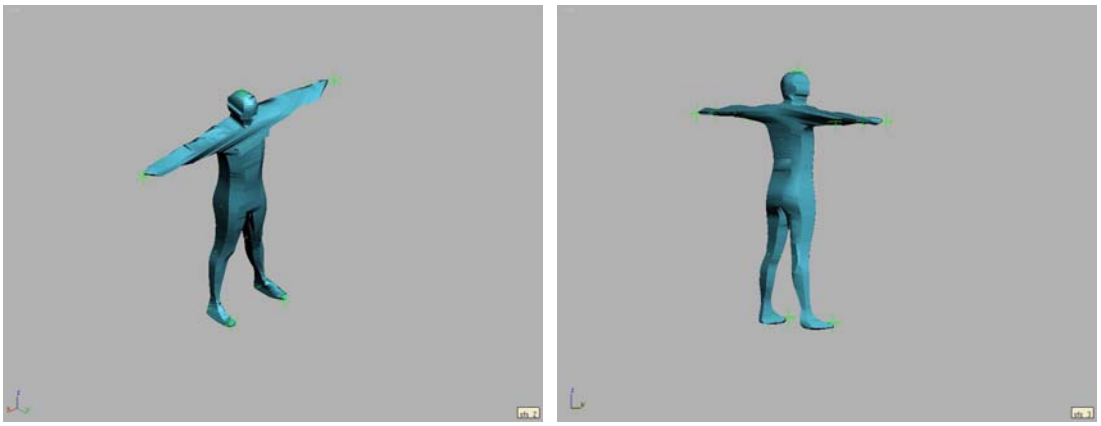


Figure 2.3. Shape-from-silhouette using 6 cameras. Some parts of the body are very blocky and unreal.

There are other approaches that tackle the 3D reconstruction of human by first building an example database from range scanned 3D human models e.g. whole-body range scans [4], and human faces [10]. This database is parameterized into a 3D morphable characterization so that 3D model can be reconstructed from a new subject's image query. The example database and the parameters characterizing the morphable model need to be well represented so that interpolation rather than extrapolation takes place while processing the new query images.

Finally, there are a few remarks pertaining to constructing the 3D model:

1) It is impossible to recover the scale of a 3D object, unless some kind of metric measurement of the 3D scene is known e.g. baseline between views.

2) We cannot recover the absolute position and orientation of the cameras and the 3D structure. Only their relative values can be computed, unless if there are some form of global positional references.

3) The reconstructed model must have regular surface i.e. should not contained non-manifold problems e.g. holes and open edges. This is to ensure that the animation, deformation and rendering of the model can take place correctly.

2.5. Estimation of Skeletons

In order to synthesize the correct animation and acquire the true kinematics, the skeleton of the 3D puppet must be similar to the subject. In general, scanner-based and multi-camera systems do not reconstruct the human skeleton. In medical applications, the human skeleton is obtained via X-ray, however its drawbacks are: (1) X-ray is not easily available, (2) registering and integrating the X-ray data of skeleton with the 3D surface model could require tedious post-processing.

There are works reported for estimating the skeleton joints location using commercial mocap systems e.g. [22], [61], [103] and [105]. They are usually done by using commercial motion capture devices to estimate the joint locations with the subject undergoing predefined motions. There are 3 important factors to take note while using these methods: (1) tedious post-processing may be needed to clean up the data, (2) the joints that are estimated come from the extrapolation of data since the sensors are placed on the surface of the subject since they are not at the joints, and (3) it is not the real anatomic skeleton or bone, which may be useful for animation.

Instead of estimating skeleton joints by using commercial mocap systems, some researchers also attempted to reconstruct the instantaneous posture form using its skeletons and joints over a sequence of frames [7], [142], [162]. These methods assume that the

point correspondences are given, and usually obtained via tedious manual point-clicking over the sequence of images.

Another common approach to estimate skeletons and joint locations is to use medial axis extraction algorithms based on mathematical morphology. Many techniques to compute 2D and 3D skeletons of object have been proposed over the years [53], [87]. In [127] skeletons were obtained from 2D images of the segmented subject, and in [90], [155] the authors extract the skeleton of the 3D model constructed by its visual hull. However, researchers have acknowledged that such approaches are very sensitive to noise and the uncertainties due to noise are not easily tractable. Other works attempt to improve the skeleton estimation by labeling body parts to kinematics chain of super-quadrics representing voxel data [144]. Also in [98], they derive a common kinematic structure through representing the shape of the subject in each time frame by using augmented Multiresolution Reeb Graph [62]. However, we have to take note that the skeletons obtained mathematically are not the same as anatomic skeletons and do not take into consideration the skin deformations, hence numerical skeletons will not give the true anatomic joint locations e.g. the human's elbow joint is not at the centre of the surface skin.

2.6. Discussions and Summary

In this chapter, we have presented a general survey of the existing literature for 3D visual tracking and modeling of humans. We have also briefly addressed the issue of skin deformation, which by itself requires focused attention. The vision-based approaches that had been used for human motion tracking were very much diversified. Right from the entry point, they utilize different kinds of segmentation techniques to extract various

representations for the tracker, ranging from feature or contour points to optical flow, and each of these have its own problems such as occlusion, lack of texture and foreshortening. Hence, there is no fool-proof method for image segmentation that can cater for all the environments. The tracking algorithms can range from active contours, example-based, numerical optimization for correlation matching. All these methods were proposed to meet a particular application or operational need. Hence to derive an algorithm that is general needs a lot more research and development work e.g. efficient construction of large set of model databases, tests and verifications. The main conclusion of the surveyed methods was they are still too fragile for practical usage. This motivates our approach whereby good 3D geometrical human puppet models of the individual subjects are reconstructed and applied to motion tracking. In addition, we will attempt to make use of the natural texture and illumination information instead of relying heavily on the problematic feature segmentation and representation, for doing the correlation matching.

Chapter 3.

Framework

This chapter presents the overview of our motion capture framework and summarizes the methodologies for acquiring 3D human motion. The pipeline for the motion capture consisted of a series of processes. In Section 3.1, we give the process overview. In Section 3.2 and Section 3.3 we explain the 3D generic human puppet model and the main concept to adapt it to fit with the specific subject. Finally, after the 3D puppet is pre-positioned (Section 3.4), motion acquisition will take place and is described in Section 3.5.

3.1. Process Overview

Figure 3.1 shows the block diagram of our human motion capturing framework. The typical flow of our process is:

1) Construction of the customized 3D human models of the subjects and store them into the database.

2) Pre-positioning of human model to register with the subject seen in the starting image of the video sequence.

3) Automatic tracking of the 3D human motion by analyzing the differences between the real images and the synthesized animated human motion with the use of feedback.

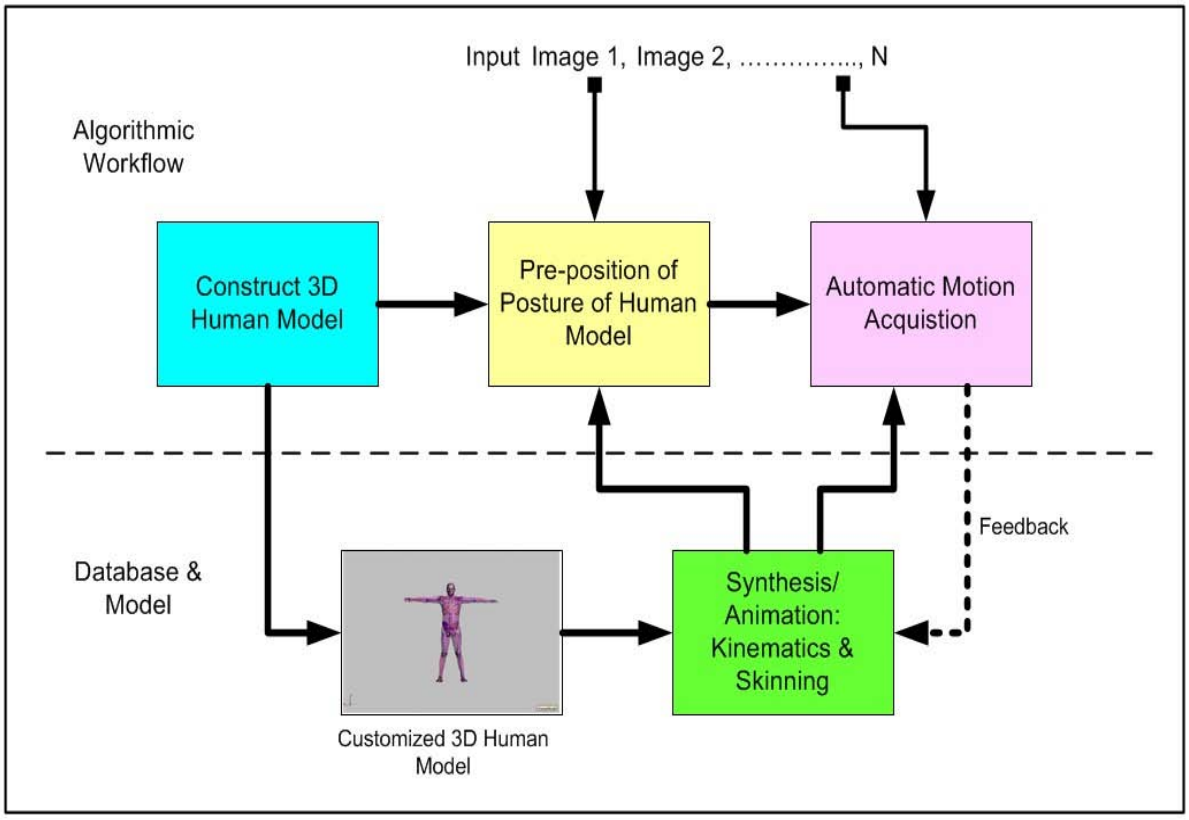


Figure 3.1. Block diagram of our human motion capturing framework

In this dissertation, we focused on (1) the reconstruction of a 3D human puppet model, and (2) tracking the human motions in the video images by synthesizing the posture of the customized animated 3D puppet using rigid skin transformation. For the tracking of human motions, we concentrated on acquiring the movement of the arms, which is quantitatively small and articulated. This will give the flavor of the global human body tracking that we intend to implement in the future. The pre-positioning is done by using the 3D interactive software e.g. 3DS Max [163]. The components (1) and (2), and their integration allowed us to realize the framework of Figure 3.1. The evaluation of the success can then lead us to consider the feasibilities in more complicated scenarios e.g. multiple people full-body tracking in highly cluttered environments. A substantial amount of work has been targeted at the reconstruction of the 3D human model since prior research works had indicated and

stressed the importance of using an accurate 3D puppet model for the acquisition of human motion.

3.2. Generic 3D Human Model

Our human model reconstruction will make use of a generic 3D human model that is made up of the external surface mesh and its skeleton (Figure 3.2). This 3D generic model that we used is the ACT human model that is available in 3DS Max. The external skin is a regular 3D surface mesh made up of about 15000 vertices and 40000 triangles. Its skeleton is an anatomical human bone structure and the respective joint nodes are linked up by the kinematics chain.

Figure 3.3 shows the overlaying of the generic model onto the different subjects in the real images from separate camera views. We can notice that they are not the same in term of both the shape and size, thus using the generic model for tracking will not yield the correct result.

A higher level representation can be obtained by partitioning the human model into various body parts i.e. head, torso, left leg, right leg, etc. Each body part is made up of a 3D mesh. Figure 3.4 shows the different body parts labelled with various colours. This higher level representation will be useful during the registration of the local human limb information (Chapter 6) and 3D human pose tracking (Chapter 7).

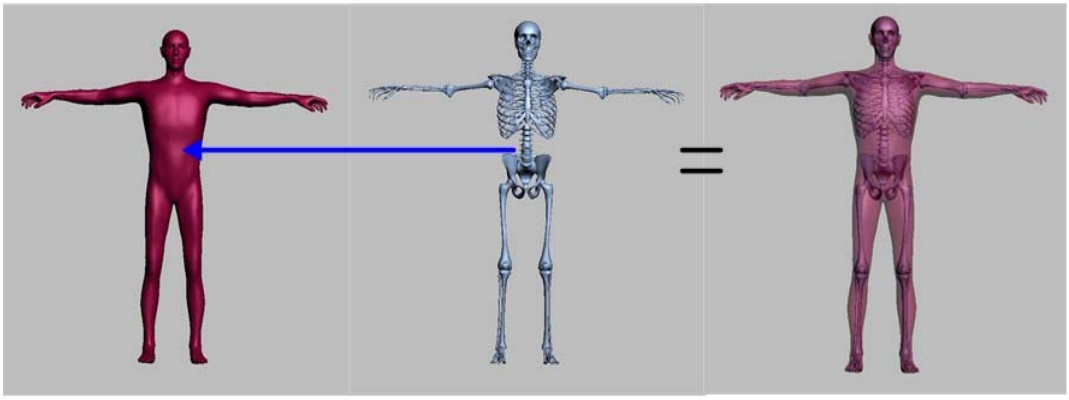


Figure 3.2. (a) Generic surface model, (b) generic skeleton, and(c) overall generic model

Figure 3.3. Examples of overlaying the generic model onto the subjects in the real images

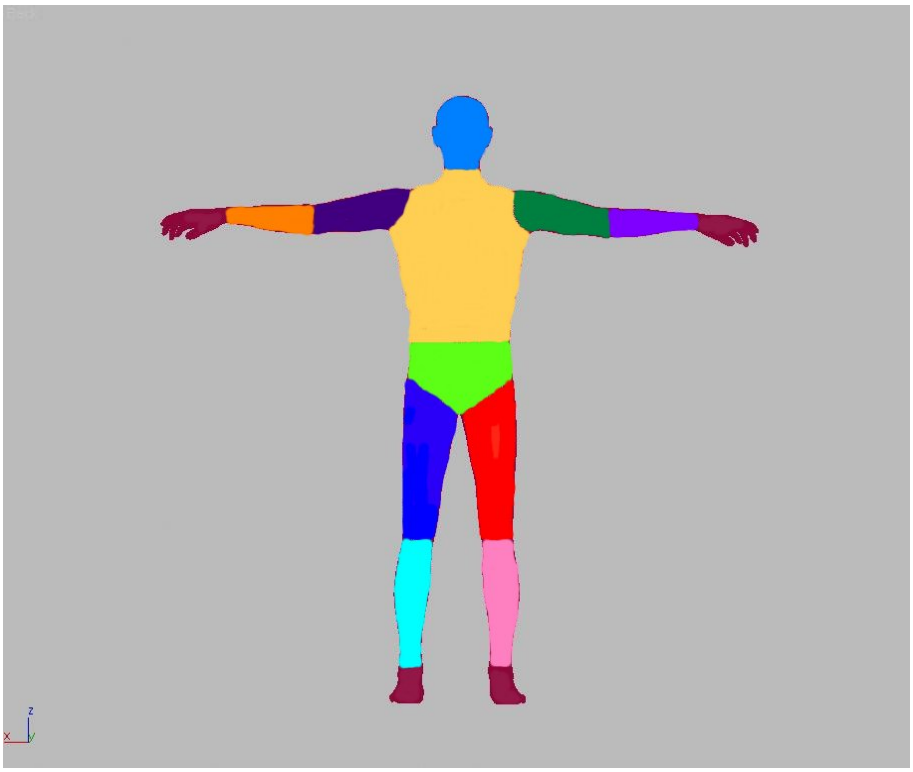


Figure 3.4. Various body parts labelled with different colour for higher level representation

3.3. Reconstruction of 3D Human Model

The block diagram of our model construction system is shown in Figure 3.5. Our task is to construct precisely the external skin of the model given (1) a 3D generic model and (2) a limited set of images of the subject acquired from different views. We used images from 6 different views providing an almost full body coverage of the subject. In this setup, we have 3 sets of unknowns parameters that we desire to solve: (1) camera poses, (2) shape and (3) size of the subject. Given these unknowns that are inter-related, our challenge is to resolve all of them at the same time from a simple setup.

Our method does not need any special calibration tools. The anatomic measurement of the player is used to deform this generic model to produce a specific model. Here, our strategy is to use the human body itself as a calibration tool. The generic 3D model guides

the camera calibration, which, in turn, allows 3D point reconstruction to estimate the camera poses and produces a customized 3D model. The generic 3D model is important because it also allows us to preserve the regular topology of the surface after its deformation, which is necessary during skinning and graphical rendering. Another factor to consider is that we do not want our subject to stay still or rigid for the whole duration of image acquisition (e.g. stay still for 10 seconds). Here, acquisition is instantaneous. Characteristic points are first used to yield an intermediate model, and next, silhouette limb curves are used for refinement to obtain the final 3D human model with its skeleton.

The inputs to the algorithm are:

1) 2D images from the wide baseline views (ideally we should have good view coverage of the subject). This acquisition will be done in a single time instance (without the need of the subject to stay still or rigid for few seconds) by gen-locking the cameras. A maximum of 6 images had been used.
2) Generic 3D human model (surface skin, skeleton and joint nodes; see Figure 3.2).
3) 2D/3D feature point matches in the image views and 3D generic human model points. The point correspondences are performed interactively with the system only at the beginning of the procedure (Figure 3.9).
4) Silhouette contours of the subject seen in the different views. This can either be done interactively via a user or get the subject to wear special colored clothes for automatic contour extraction.

The outputs of the algorithm are:

1) Calibrated camera poses of the different views.

2) Customized 3D model that will overlay nicely onto the images of the subject's silhouette limb in all the views.

This task can be realized on an off-line basis. It comprises four main stages: (1) choice of the 3D precise location of the characteristic points chosen by the user on the 3D generic model using the choice proposed qualitatively by the software on an IMAGE of this generic model; (2) camera calibration and reconstruction of model characteristic points, as described in Chapter 4, (3) refinement of model via silhouette limbs deformation, and (4) skeleton estimation, as described in Chapters 5 and 6.

Representation of the customized 3D human model consists of:

1) The surface skin made up of triangle mesh.

2) Bones made up of triangle mesh as well. Alternative, for simplicity, they can be 3D curves/lines formed by the medial axes of these bones. The bones of different body parts are also linked through a forward kinematics chain, the root of which starts from the pelvis. During 3D animation, the bones will drive the surface skin by manipulating its kinematics, which deform the surface accordingly (please refer back to Section 2.3 for the issue on skin deformation).

3) Given also the calibrated cameras parameters and real images of the subject, the texture coordinate of each vertex on the 3D model can be computed by back-projecting its 3D position through the camera parameters onto the 2D image coordinate i.e. (u, v) coordinate. This texture coordinate is used for shading and rendering of the 3D model during the motion animation.

This scheme of representing our 3D model using triangle mesh and texture were chosen since they are easy representation for image synthesis and are well supported in many existing graphics hardware.

Our approach consists first in showing Figure 3.6 to the user. This figure presents images of the four main views of the generic model we chose. In this figure, the user may see green circles and numbers which specify the 32 characteristic points that the technique is going to manipulate. They were chosen from a bigger set of points but we chose the points that we believe to be the easiest to locate PRECISELY on images of real humans wearing clothes (that is the problem!). So, the user may immediately see what are the characteristic points we want him to play with. This choice is qualitative (semantic, if we may say so!). On the generic model, 32 anatomical characteristic points are proposed by the system (Figure 3.6). Then the user has to choose HIMSELF the 3D locations he wants to use ON the generic model by simply clicking on each of the 32 characteristic points of the image(s) on the left and THEN click on his choice of 3D vertex of the 3D generic model wireframe (see Figure 3.7).

Initialization of the 3D human reconstruction begins by establishing the 3D characteristic points and their 2D correspondents in the respective images by using an interactive point-matching tool that we have developed. Figure 3.7 shows an example of selection of points (red crosses) on the 2D image of the subject guided by the 3D anatomical features on the 3D generic model. The 32 points of the 3D generic model are shown from two different viewpoints in Figure 3.8. Figure 3.9 and Figure 3.10 show the 2D feature points on the subject's images (Please view Appendix D for more examples of features points on the subject's images).

Human input is to ensure that the correspondents are all correct, so that the calibration will be stable. Although feature extraction for wide baseline images had been reported [8], [24], automatic feature extraction methods still face problem under unpredictable lighting condition, lack of texture, foreshortening and occlusion in our situation when the views have wide-baseline for full body coverage. Moreover good image features that are based on texture invariants are different from anatomical landmarks such as the finger tips, knees, etc. Also, visual anatomical landmarks are important clues related to the underlying skeleton. Therefore, image features will not be used since they may not be of meaningful anatomical landmarks.

Although human input is required, it is limited to 32 feature points in several 2D images ONLY at the beginning of the process, and neither cleaning up nor further tweaking is needed during the program execution later on. Thus, the operation of our setup, which requires no prior camera calibration and moderate human intervention, are simple, cheap and efficient. In [129], they had also carried out a similar approach. Many systems that were developed to provide good and reliable 3D reconstruction usually begin with some kind of user interventions e.g. [36], [100] and [139]. A fully automatic system can be easily appended to our framework in the future if reliable feature extraction and point correspondent techniques are available.

Through our study, we also know that feature points alone are insufficient to build the 3D model. Therefore, silhouette information will be utilized to improve the shortcomings in just feature points. This is done through matching the projected silhouette of the 3D model with reference to the real image silhouette of the subject (stage 2: silhouette limbs reconstruction in Figure 3.5).

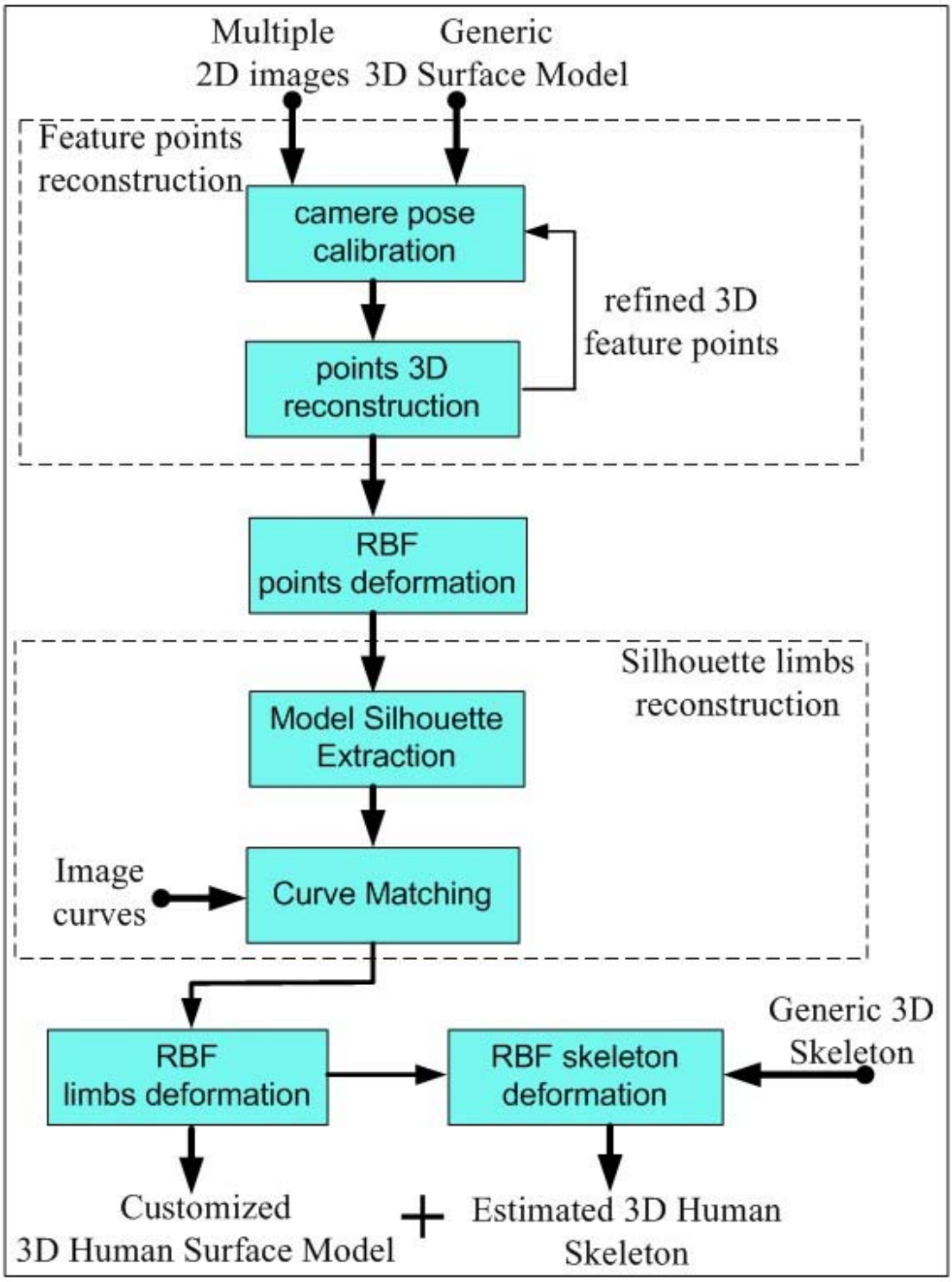


Figure 3.5. Block diagram of our 3D human construction algorithm

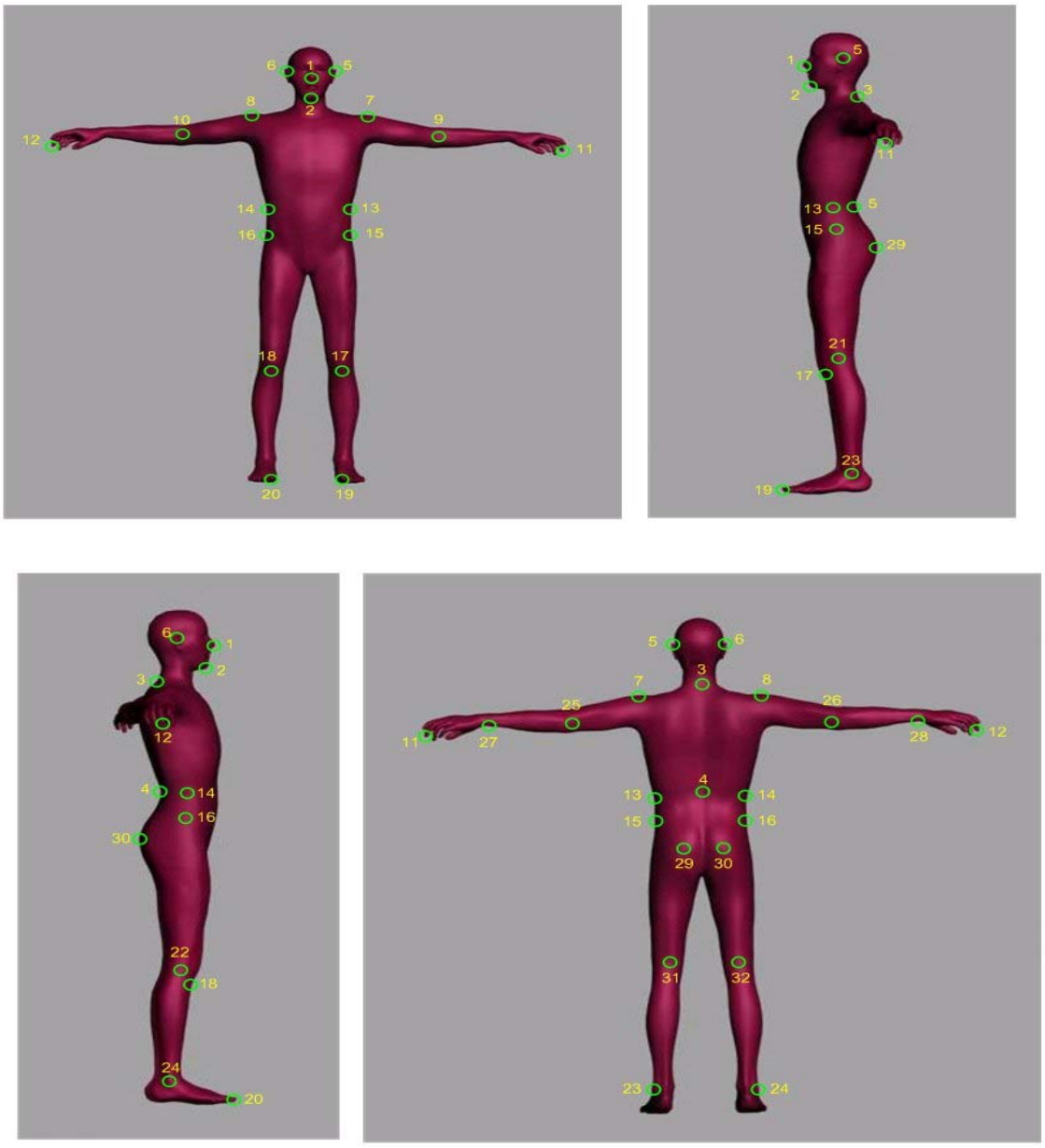


Figure 3.6. The 32 anatomical related characteristic points chosen are visualized (in green circles) on the generic model

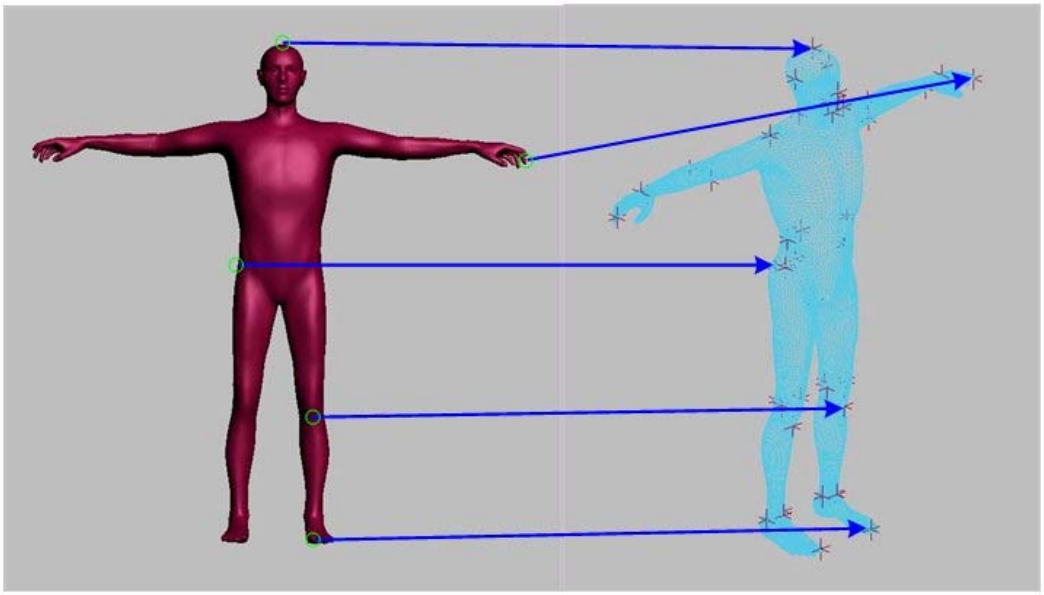


Figure 3.7. Selected 3D anatomical related characteristic points of the generic model (on the left) are chosen INTERACTIVELY on the 3D generic model on the right

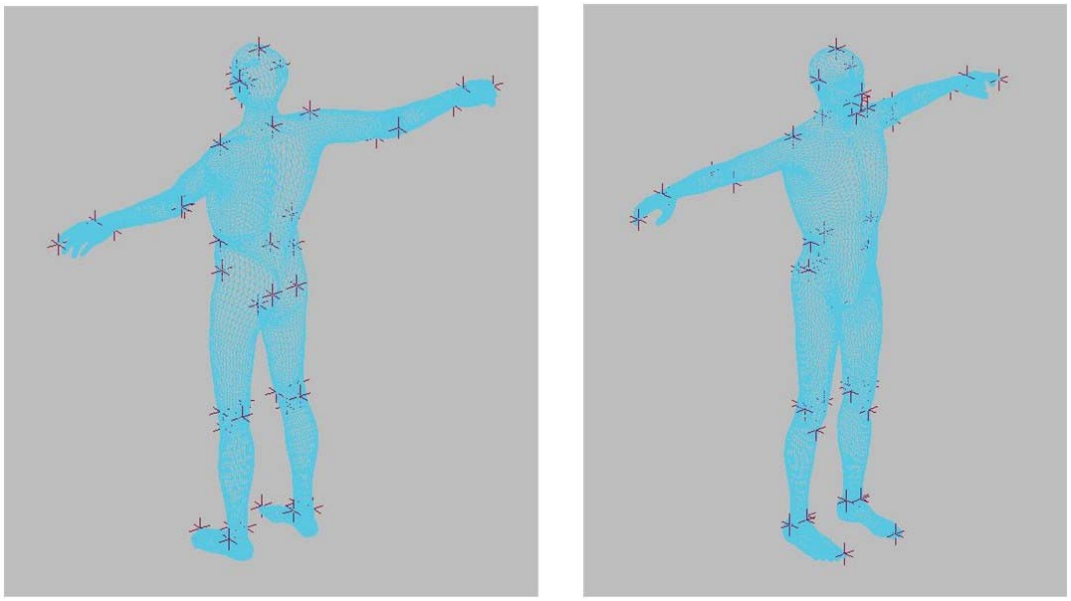


Figure 3.8. 3D characteristic points on the generic model seen from different views

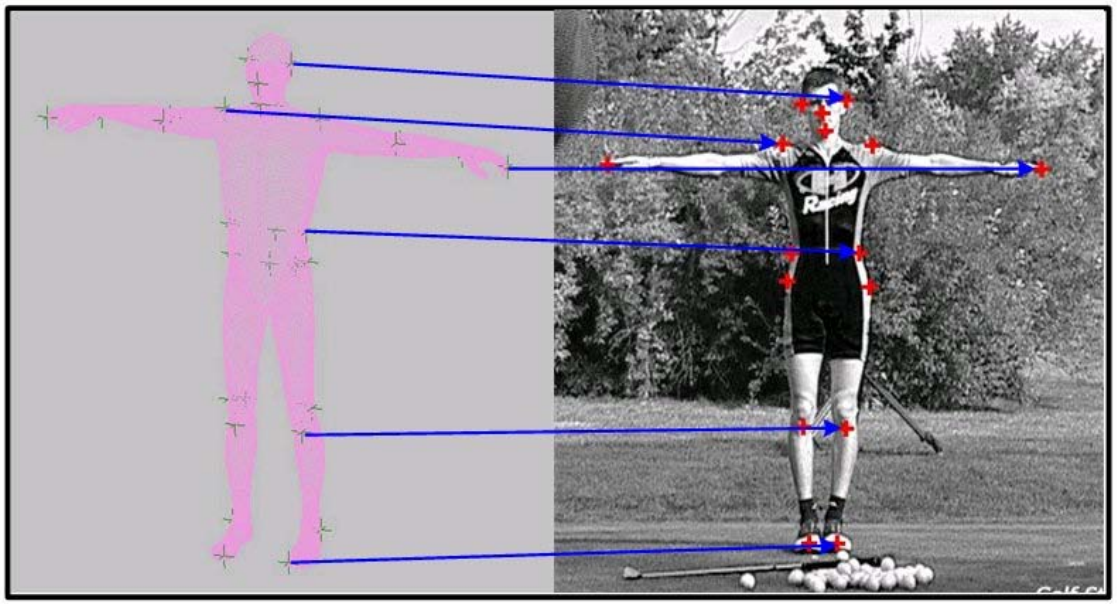


Figure 3.9. Example of characteristic points selection on an image of a real golf player (on the right). They appear as RED crosses when they are clicked. The user first clicks on a green cross of the generic model (left) and then chooses its correspondent on the image

Figure 3.10. Examples of feature points of the subjects' images from different views

3.4. Pre-positioning

Before the automatic tracking of human motion can begin, the posture of the 3D model puppet is registered and aligned with the subject in the first image of the video sequence in

all the respective camera views. In this dissertation, we achieve the pre-positioning of the initial human posture interactively by using 3DS Max. Figure 3.11 shows the pre-positioned posture of a golf player's arms at the same time instance overlaid onto images from different views. An alternative way to do the pre-positioning is by automatic recognition of human posture, however based on existing literature more work have to be done and is outside the scope of our discussion.

Figure 3.11. Pre-positioned upper and lower arms of golfer overlay onto different views

3.5. Motion Tracking

A model-based analysis-by-synthesis framework is proposed for the tracking of articulated human motion. A simple flow of this process is shown in Figure 3.12. The detailed description of this methodology is explained in Chapter 7. The method that we used attempts to deal with difficulties caused by images with cluttered and moving background, and partial occlusion.

The inputs to the algorithm are:

1) Instantaneous video streaming from the multiple cameras.

2) 3D animatable human puppet model of the subject and its color texture information.

The outputs of the algorithm are:

1) Instantaneous 3D posture of the subject, represented by its 3D joint kinematics.

2) The 3D mesh of the subject's instantaneous body posture overlaid onto the respective input images undergoing study.

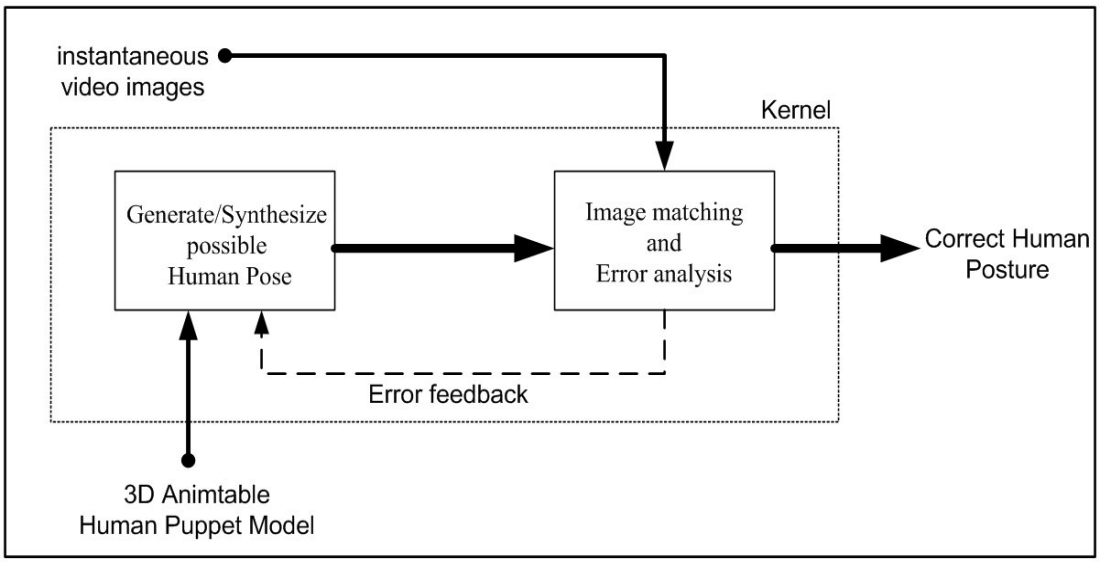


Figure 3.12. Simplified overview of our human motion tracking algorithm

Once the 3D animatable puppet model has been positioned and registered correctly with the subject in the images at first time instant, the color texture of these images will be mapped onto the puppet via back-projecting the geometric information onto the 2D images. The possible human pose will be generated and the 2D images of the synthesized pose will be rendered. Then, the rendered images will be compared with the instantaneous video images, while the matching errors are analyzed and fedback to the synthesizing module. This analysis-by-synthesis loop iterates until it converges when the errors are minimized. The algorithm will proceed to the images from the next time instant. The kernel of the

analysis-by-synthesis algorithm can be realized by using a numerical minimization method such as the simulated annealing.

Chapter 4.

Camera Calibration and Reconstruction from Feature Points

This chapter describes the first stage of our model adaptation system, which is made up of the first three blocks in the system shown in Figure 3.5. We will carry out camera calibration and 3D model reconstruction at the same time by using un-calibrated images filmed from wide baseline. Using a set of 2D characteristic points from the subject's images in cooperation with their respective correspondents from the 3D generic model, we iterate a process comprising the camera calibration and 3D point reconstruction until convergence is achieved. We will obtain a set of sparse deformed 3D model points and calibrated camera poses. By using the sparse deformed model points, we complete our initial customized 3D model by interpolating the deformations to entire generic model using radial basis functions (RBF). The results show that our method can operate in different scenarios such as human subject of varied shapes and sizes, and possible input uncertainties.

4.1. Camera Calibration and Pose Estimation

Pose estimation and camera calibration can be thought of as processes to determine the geometric mapping between 2D image pixels and 3D rays in the world space. Given an image I and the 3D geometry of an object E of the film scene, we want to find the

geometric relation between the camera and the object E as it appears in the image I. The parameters we seek for during these processes are (1) the extrinsic parameters (i.e. the rigid rotation and translation representing the camera's orientation and position in the world coordinate system), and (2) intrinsic parameters (consisting of the focal length, optical center, pixel ratios and distortions).

Over the years, many camera calibration techniques were already been developed. Every camera calibration algorithm assumed that the 3D object information and their corresponding features in the 2D images are already established. These features are usually lines or points. The main approaches are: closed-form method, numerical solution, or relating linear camera model to the perspective model by iterative solutions.

4.1.1. Closed-form method and numerical solution:

One of the early approaches of this class is the Tsai's method [146]. Since then many related calibration techniques had been proposed [57], [59], [151] and [161]. From the 3D/2D point or line correspondents, a linear least square formulation is carried out before the intrinsic and extrinsic parameters are decomposed to obtain a closed-form solution [161]. However, if the closed-form solution is insufficient, then it can be used as an initial estimate followed by a non-linear optimization method to refine the solution. The accuracy and stability of the non-linear optimization is highly dependent of the uncertainties of the 3D/2D correspondents.

4.1.2. POSIT for Camera Calibration

The geometrical relation between a weak perspective pinhole camera and a perspective one may be formulated. DeMenthon [33] proposed the POSIT (pose iteration) algorithm, which

improves the camera pose iteratively starting from a weak perspective camera model until convergence is reached. Later, R. Horaud in [63], proposed a refinement by beginning from the para-perspective camera model and introduced a way to take into account the orthogonal constraint associated with the rotation matrix. POSIT, iteratively minimizes the distance between estimated projections of features points of the object and their real projections in the image. As POSIT is iterative, it requires a stopping criterion which is directly related to the quality of the calibration. The possible stopping criteria are: (1) when the maximum or mean of the difference (called re-projection error) between the real points in the image and the projections of the 3D corresponding feature points is lower than a threshold, (2) the rate of change of re-projection error is lower than a certain threshold or (3) after a maximum number of iterations have elapsed.

The original POSIT algorithm was performed with known values of intrinsic parameters. We can find the intrinsic parameters prior to using POSIT by using a camera calibration algorithm like [14] as preprocessing. Another alternative that we propose is to add an additional layer above POSIT in order to seek for the intrinsic parameters. This is done by regarding POSIT as a function of the intrinsic parameters. For the true set of intrinsic parameters, the calibration provided by POSIT will be optimal, whereas, for any other set of intrinsic parameters, the pose estimation will be of worse quality. We will minimize the re-projection error of the pose estimation in the space of the intrinsic parameters. This minimization can be done by using the “downhill simplex” algorithm [116]. The simplex minimization will converge to local minima, and in practice we can further constrain the search space of the downhill simplex. Multiple initialization points can also be tried once in every few millimeters interval and the computations can be

completed in a few seconds using modern day processors. We can also limit the focal length to be greater than 0 and not more than 1000mm (telescopic camera). Experiments have shown that optimization of intrinsic parameters did not bring very satisfactory results.

The POSIT technique is adapted for calibrating our cameras, since it turned out to be very stable during our computations. In our camera calibration setup, the calibration tool is the subject itself. We do not need any special calibration tool.

4.1.3. POSIT Formulation

A classic pinhole camera model is shown in Figure 4.1. The feature points of a 3D reference object are $\mathbf{M}_i$ where i = 1, 2, …N, and N is the total number of feature points. These points is projected onto an imaging plane **G**, thus producing $\mathbf{m}_i$. We have the following sets of known values:

1. The 3-vector feature points of $\mathbf{M}_i$ in the object reference system, (U_i, V_i, W_i). For simplicity, we can use this as the world coordinate system.

2. The projected 2-vector $\mathbf{m}_i$ of the feature points on the imaging plane, (x_i, y_i).

We can compute $\mathbf{M}_o$, the centroid of the $\mathbf{M}_i$ points in the object reference system, (U_o, V_o, W_o). From these known values, we seek for rotational and translational transformations of the camera with respect to the reference object. Also $\mathbf{M}_i$ and $\mathbf{M}_o$, which are unknowns in the camera coordinate, have the coordinates (X_i, Y_i, Z_i) and (X_o, Y_o, Z_o) respectively. The rotational transformation, also called the rotational matrix can be written as:

$$\mathbf{R} = \begin{pmatrix} \hat{\mathbf{i}}^{\mathrm{T}} \\ \hat{\mathbf{j}}^{\mathrm{T}} \\ \hat{\mathbf{k}}^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{pmatrix} \tag{4.1}$$

Then, refer again to Figure 4.1, the translational transformation, **T**, is the vector $\overrightarrow{\mathbf{OM}_o}$, which is equal to $\frac{Z_o}{f}\overrightarrow{\mathbf{Om}_o}$, by assuming that the focal length f is known. Next recall the fact that $\hat{\mathbf{k}} = \hat{\mathbf{i}} \times \hat{\mathbf{j}}$. Thus, the pose of the camera with respect to the object, and vice versa, is fully defined once we find $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$ and Z_o.

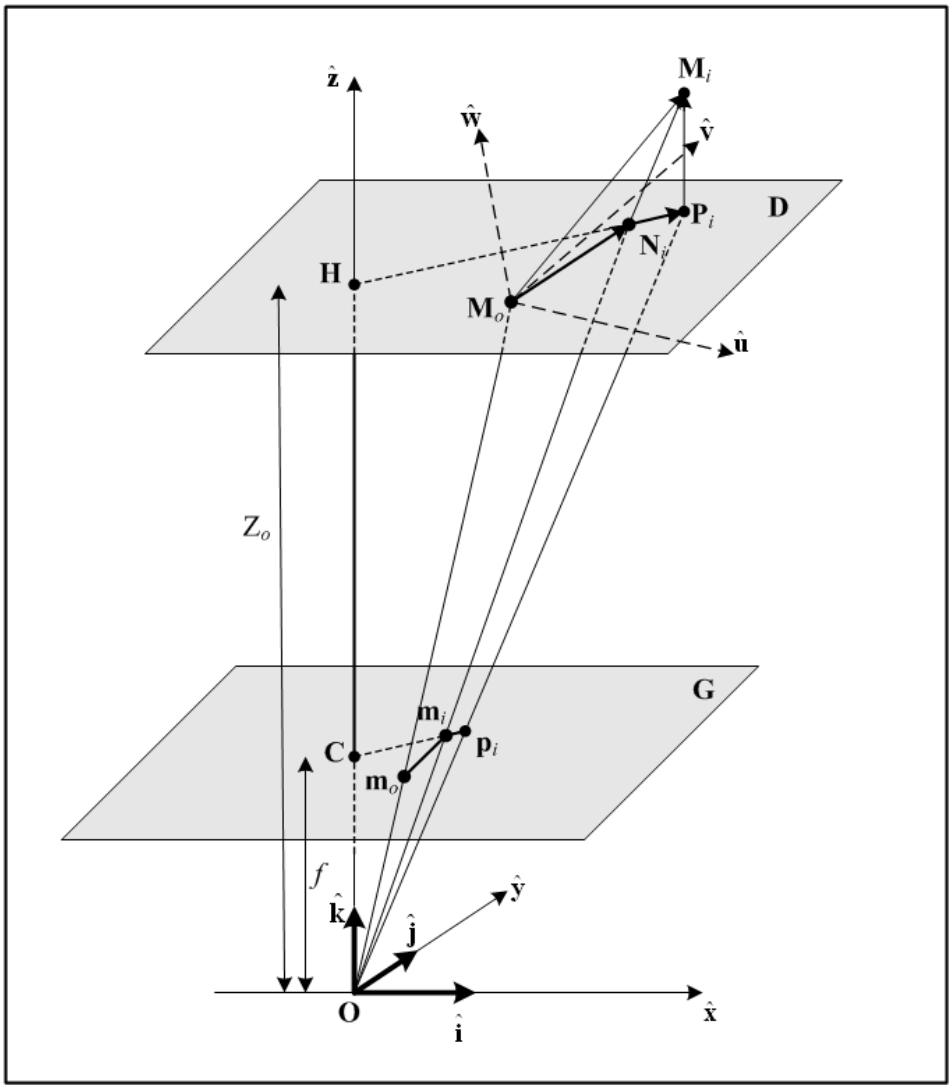


Figure 4.1. Perspective projection of $\mathbf{m}_i$ and scaled orthographic projection of $\mathbf{p}_i$ for an object $\mathbf{M}_i$ and a reference point $\mathbf{M}_o$.

POSIT Geometry

Scaled orthographic projection (SOP) is an approximation to the "true" perspective projection. In SOP, the image of a point $\mathbf{M}_i$ of an object is a point $\mathbf{p}_i$ on the image plane, which has the coordinates:

$$x_i' = fX_i/Z_o, \quad y_i' = fY_i/Z_o \tag{4.2}$$

Whereas for perspective projection an image point $\mathbf{m}_i$ would be obtained instead of $\mathbf{p}_i$ with coordinates:

$$x_i = fX_i/Z_i, \quad y_i = fY_i/Z_i \tag{4.3}$$

Then, the image coordinates of the SOP projection of $\mathbf{p}_i$, where $s=\frac{f}{Z_o}$, can be written as:

$$x_i' = fX_o/Z_o + f(X_i - X_o)/Z_o = x_o + s(X_i - X_o),$$

$$y_i' = y_o + s(Y_i - Y_o) \tag{4.4}$$

Now let us consider the equations that characterize the "true" perspective projection and relate the unknowns $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$ and Z_o to the known coordinates of the 3-vector $\overrightarrow{\mathbf{M}_o\mathbf{M}_i}$ in the object coordinate system, and to the known image points $\mathbf{m}_i$ and $\mathbf{m}_o$.

$$\overrightarrow{\mathbf{M}_o\mathbf{M}_i} \cdot \frac{f}{Z_o}\hat{\mathbf{i}} = x_i(1+\varepsilon_i) - x_o \tag{4.5}$$

$$\overrightarrow{\mathbf{M}_o\mathbf{M}_i} \cdot \frac{f}{Z_o}\hat{\mathbf{j}} = y_i(1+\varepsilon_i) - y_o \tag{4.6}$$

in which ε_i is defined as (please see [33] for proof):

$$\varepsilon_i = \frac{1}{Z_o} \overrightarrow{\mathbf{M}_o\mathbf{M}_i} \cdot \hat{\mathbf{k}} \tag{4.7}$$

Next equations (4.5) and (4.6) can also be written as:

$$\overrightarrow{\mathbf{M}_o\mathbf{M}_i} \cdot \boldsymbol{I} = \zeta_i, \quad \overrightarrow{\mathbf{M}_o\mathbf{M}_i} \cdot \boldsymbol{J} = \eta_i \tag{4.8}$$

where: $\mathbf{I}=\frac{f}{Z_o}\hat{\mathbf{i}}$, $\mathbf{J}=\frac{f}{Z_o}\hat{\mathbf{j}}$, $\zeta_i = x_i(1+\varepsilon_i) - x_o$, $\eta_i = y_i(1+\varepsilon_i) - y_o$

and the term ε_i have known values computed through the previous iteration with an initial values of all 0.

When given all the *N* feature points, equation (4.8) can be expressed in the form:

$$\boldsymbol{AI}=\boldsymbol{x}', \boldsymbol{AJ}=\boldsymbol{y}' \tag{4.9}$$

Hence computing ***B***, the pseudo-inverse of ***A***, we get:

$$\boldsymbol{I}=\boldsymbol{Bx}' \text{ and } \boldsymbol{J}=\boldsymbol{By}' \tag{4.10}$$

POSIT Pseudo-code

The steps for POSIT algorithm for *N* feature points can be executed as follow:

1. Form the matrix ***A*** as in equation (4.9)
2. Compute the matrix ***B***, which is the pseudo-inverse of ***A***.
3. Initialize ε_i^n at iteration n=0. The superscript n denotes the iteration count.
4. Beginning of loop, starting from n=1: Compute $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$.
   - Compute $x' = x_i(1+\varepsilon_i^{n-1}) - x_o$ and $y' = y_i(1+\varepsilon_i^{n-1}) - y_o \ \forall\, i$ from ε_i^{n-1} to yield $\boldsymbol{x}'$ and $\boldsymbol{y}'$.
   - Obtain ***I*** and ***J*** via equation (4.10).

- Compute the scale *s* as the average between the norms of ***I*** and ***J***.
- Normalize ***I*** and ***J*** into unit vectors $\hat{\mathbf{i}}$ and $\hat{\mathbf{j}}$.

5. Compute new ε_i^n:
   - Compute $\hat{\mathbf{k}} = \hat{\mathbf{i}} \times \hat{\mathbf{j}}$
   - Compute $Z_o = f/s$
   - Compute ε_i^n by using equation (4.7)
6. If $|\ \varepsilon_i^n - \varepsilon_i^{n-1}\ | >$ threshold, $n=n+1$; Go to step 2.
7. Compute the translational vector $\mathbf{T} = \overrightarrow{\mathbf{OM}_o} = \overrightarrow{\mathbf{Om}_o}/s$
8. Set the rotational matrix to be orthonormal: $\hat{\mathbf{k}}' = \hat{\mathbf{k}}/|\hat{\mathbf{k}}|$, $\hat{\mathbf{j}}' = \hat{\mathbf{j}}/|\hat{\mathbf{j}}|$.

4.2. Characteristic Points Reconstruction

From the camera pose calibration we get a set of camera parameters, which maps each 2D image pixels onto 3D rays in world space. We also have, at entry of the system, (1) the generic 3D model in triangular meshes and (2) a set of 2D characteristic points from the images corresponding to their respective 3D points on the generic model.

We will deform the 3D characteristic points towards the new positions through 3D point reconstruction using the calibrated camera poses and the set of 2D characteristic point correspondents from the images. Given *M*, the total number of images acquired, each of the 32 feature points (indexed by *i: i* = 1, 2, ….32) could be seen in the images K_i times ($K_i \leq M$). We will not reconstruct the point if a feature point is only seen in 1 image i.e. when $K_i<2$.

For the feature points that are visible in 2 or more views, we can compute their 3D position via the intersection of projected rays (multi-stereo approach). We have taken into account that the 3D rays do not intersect when the calibration is not perfect (which is our case). To construct these 3D points, we generalized the stereo triangulation reconstruction algorithm for multiple views. It is a linear algebra solution which seeks for a point in 3D space such that it minimizes the sum-of-squares of distances to the projected rays from all the possible views (Figure 4.2). More precisely we take each pair of rays and take the middle of the segment which minimizes the distance between the two rays and compute the centroïd of all these middle points to get the result.

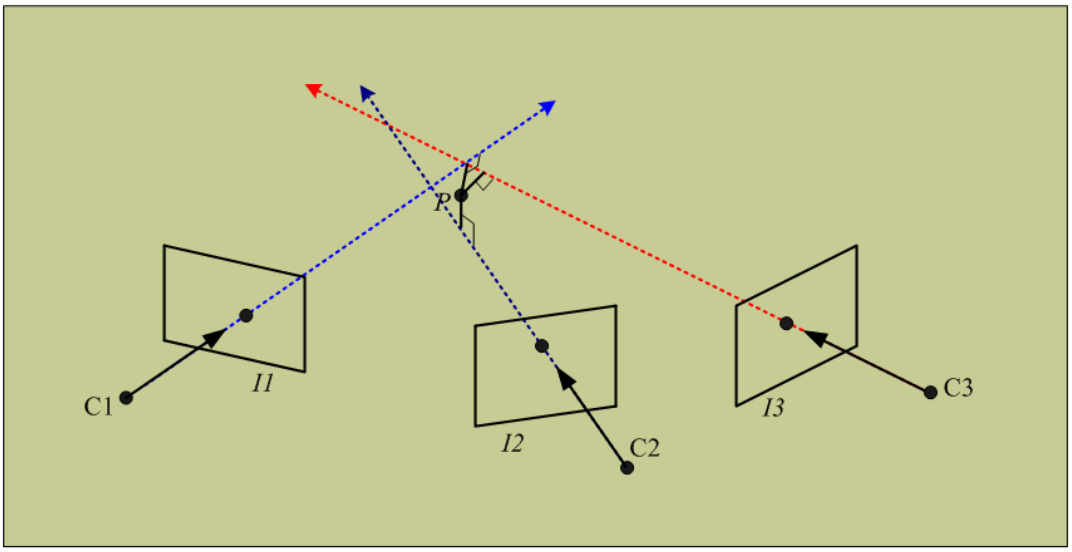


Figure 4.1. Triangulation of projected rays from the images and 3D reconstruction when the rays do not intersect (*P* is the reconstructed point)

4.2.1. 3D Triangulation from Multiple Ray Intersection

Given a 2D image point in pixel coordinate, we can convert it into the metric unit by knowing the camera imaging sensor size. Alternatively, if we know the field-of-view and the focal length, this value can also be calculated.

For a feature point **m** in the imaging sensor plane, it can be written as a 3-vector:

$$\mathbf{m} = \begin{pmatrix} u \\ v \\ f \end{pmatrix} \tag{4.11}$$

where f is the focal length, and (u, v) are the sensor coordinate (Figure 4.3) in metric unit e.g. millimeter.

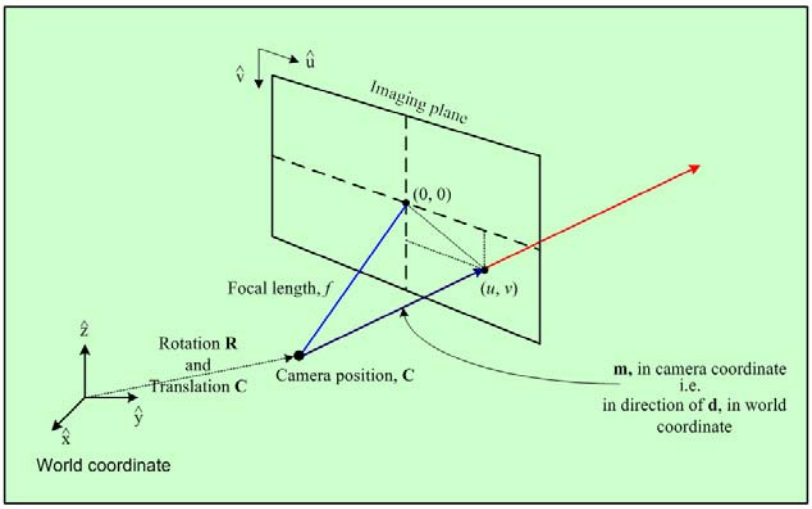


Figure 4.2. Projecting imaging ray in world coordinate

For 3D triangulation of 2D imaging points, assuming that a minimum of 2 calibrated camera views are available, we can formulate the direction of ray coming out from each imaging centre $\mathbf{d}_{i,j}$ in the world coordinate as:

$$\mathbf{d}_{i,j} = \begin{pmatrix} a_{i,j} \\ b_{i,j} \\ c_{i,j} \end{pmatrix} = \frac{1}{\|\mathbf{m}_{i,j}\|} \bullet \mathbf{R}_j \bullet \begin{pmatrix} u_{i,j} \\ v_{i,j} \\ f_{i,j} \end{pmatrix} \tag{4.12}$$

where i is the index of each 3D point when it is visible and projected to the respective camera **C** indexed by j. **R** is the rotational matrix of the each camera 3-axis vectors with respect to the world coordinate system, and it is arranged in column-wise to signify transformation from the camera coordinates to the world coordinates. The (a, b, c) are the coefficients of the unit vector **d**, $\forall\, i, j$: $\mathbf{d}_i$ visible in $\mathbf{C}_j$.

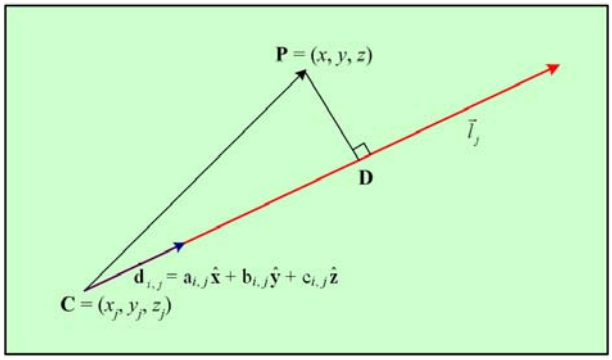


Figure 4.3. A point **P** that we want to construct by minimizing its distance to its projected ray

Consider Figure 4.4, which shows a point **P** = (x, y, z) that we want to reconstruct and a ray $\vec{l}$ that starts at the camera centre $\mathbf{C}_j = (x_j, y_j, z_j)$ and has a unit direction vector $\hat{\mathbf{d}}_{i,j} = a_{i,j}\hat{\mathbf{x}} + b_{i,j}\hat{\mathbf{y}} + c_{i,j}\hat{\mathbf{z}}$. For simplicity of illustration, we ignore the notation i to keep our explanation to 3D reconstruction for a single point. For a particular camera, the vector $\overrightarrow{\mathbf{CD}}$ is the projection of CP onto the ray $\vec{l}_j$, and we can write:

$$\overrightarrow{\mathbf{CP}} = (x-x_j)\hat{\mathbf{x}} + (y-y_j)\hat{\mathbf{y}} + (z-z_j)\hat{\mathbf{z}} \tag{4.13}$$

And its magnitude is:

$$\left\|\overrightarrow{\mathbf{CP}}\right\| = \sqrt{(x - x_j)^2 + (y - y_j)^2 + (z - z_j)^2} \tag{4.14}$$

Thus, $\overrightarrow{\mathbf{CD}}$, the projection of $\overrightarrow{\mathbf{CP}}$ onto $\hat{\mathbf{d}}_i$ is:

$$\overrightarrow{\mathbf{CD}} = (\overrightarrow{\mathbf{CP}} \bullet \hat{\mathbf{d}}_j)\hat{\mathbf{d}}_j \tag{4.15}$$

$$= [a_j(x-x_j) + b_j(y-y_j) + c_j(z-z_j)]\ \hat{\mathbf{d}}_j \tag{4.16}$$

Since $\hat{\mathbf{d}}_i$ is of unit length, the magnitude of $\overrightarrow{\mathbf{CD}}$ is then:

$$\left\|\overrightarrow{\mathbf{CD}}\right\| = a_j(x-x_j) + b_j(y-y_j) + c_j(z-z_j) \tag{4.17}$$

From the points PCD, which is a right angle triangle, we can invoke the Pythagas theorem:

$$\left\|\overrightarrow{\mathbf{DP}}\right\|^2 = \left\|\overrightarrow{\mathbf{CP}}\right\|^2 - \left\|\overrightarrow{\mathbf{CD}}\right\|^2 \tag{4.18}$$

$$\Rightarrow \qquad \left\|\overrightarrow{\mathbf{DP}}\right\|^2 = [(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2] - [a_j(x-x_j) + b_j(y-y_j) + c_j(z-z_j)]^2 \tag{4.19}$$

When we are given the 2D image correspondent in 2 or more calibrated views, for each set of correspondents seen in j cameras, we can minimize the sum-of-square error of $\left\|\overrightarrow{\mathbf{DP}}\right\|^2$ for rays projected from all the possible camera views:

$$E(x, y, z) = \sum_{\forall j} [(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2] - [a(x-x_j) + b(y-y_j) + c(z-z_j)]^2 \tag{4.20}$$

An analytical optimization solution is obtained by differentiating $E(x, y, z)$ with respect to x, y and z, and then setting the partial derivatives to zero. Thus we will obtain 3 equations:

$$\frac{\partial E(x,y,z)}{\partial x} = \sum_{\forall j} [x - x_j - a_j^2 x + a_j^2 x_j - a_j b_j y + a_j b_j y_j - a_j c_j z + a_j c_j z_j] = 0 \tag{4.21a}$$

$$\frac{\partial E(x,y,z)}{\partial y} = \sum_{\forall j} [y - y_j - a_j b_j x + a_j b_j x_j - b_j^2 y + b_j^2 y_j - b_j c_j z + b_j c_j z_j] = 0 \tag{4.21b}$$

$$\frac{\partial E(x,y,z)}{\partial z} = \sum_{\forall j}[z - z_j - a_j c_j x + a_j c_j x_j - b_j c_j y + b_j c_j y_j - c_j^2 x + c_j^2 x_j] = 0 \qquad (4.21c)$$

After rearranging the (x_j, y_j, z_j) terms on the right side of the equation, we get

$$\sum_{\forall j}[(1-a_i^2)x - a_i b_i y - a_i c_i z] = \sum_{\forall j}[(1-a_i^2)x_i - a_i b_i y_i - a_i c_i z_i] \qquad (4.22a)$$

$$\sum_{\forall j}[-a_i b_i x + (1-b_i^2)y - b_i c_i z] = \sum_{\forall j}[-a_i b_i x_i + (1-b_i^2)y_i - b_i c_i z_i] \qquad (4.22b)$$

$$\sum_{\forall j}[-a_i c_i x - b_i c_i y + (1-c_i^2)z] = \sum_{\forall j}[-a_i c_i x_i - b_i c_i y_i + (1-c_i^2)z_i] \qquad (4.22c)$$

Hence these 3 equations become the 3-by-3 matrix of:

$$\begin{bmatrix} \sum_j (1-a_j^2) & -\sum_j a_j b_j & -\sum_j a_j c_j \\ -\sum_j a_j b_j & \sum_j (1-b_j^2 & -\sum_j b_i c_i \\ -\sum_j a_j c_j & -\sum_j b_i c_i & \sum_j (1-c_j^2) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \sum_j [(1-a^2)x_j - a_j b_j y_j - a_j c_j z_j] \\ \sum_j [-a_j b_j x_j + (1-b_j^2)y_j - b_j c_j z_j] \\ \sum_j [-a_j c_j x_j - b_j c_j y_j + (1-c_j^2)] \end{bmatrix} \qquad (4.23)$$

This is a linear algebra expression of the form **AP**=**b**. Hence for each set of 2D image correspondents from the multiple views we can compute its 3D position by solving for:

$$\mathbf{P} = \mathbf{A}^{-1}\mathbf{b} \qquad (4.24)$$

4.2.2. Parallel or Anti-Parallel Ray

The equation (4.14) will not yield a unique solution when the determinant of the matrix **A** is zero. This happens when the projected rays are parallel or anti-parallel, i.e. the solution can be anywhere along the lines or simply do not exist. In practice, we can simply check for the determinant of the 3-by-3 matrix A and do not triangulate the point when the matrix is close to singularity.

4.3. Iterative Calibration/Reconstruction and Deformation Vector

When all the possible characteristic points are reconstructed, we modify the generic model on these points. Now, we have the new 3D model points and their projections onto the *M* images (which do not move). However, it is not difficult to notice that when calibrating using the subject itself, the 3D characteristic points of the generic model do not project correctly in the early iterations. We then turn over to camera calibration with POSIT to determine a better position of the (deformed) 3D model with respect to the camera. With the new camera poses, we reiterate the process of 3D reconstruction to obtain a new refined set of 3D model points, and we continue the loop of calibration / 3D reconstruction until the process converges. The convergence criterion is simply when the rate of change of camera poses between the iterations goes to zero. Although there is the element of shape dissimilarity between the real 3D model and the generic 3D model and also the high number of parameters that we need to compute, our results in Section 4.5 will demonstrate that our process converges properly. In the situation when the cameras are already calibrated beforehand, then the 3D reconstruction of points is just a single iteration.

At the end of this process, we have the original set of 3D points from the initial generic model G_i and a final set of reconstructed 3D points P_i. Using G_i and P_i we obtain a set of deformation vectors $\overrightarrow{G_iP_i}$ (see Figure 4.5 for example). The set of deformation vectors will be used for the whole 3D model (next section).

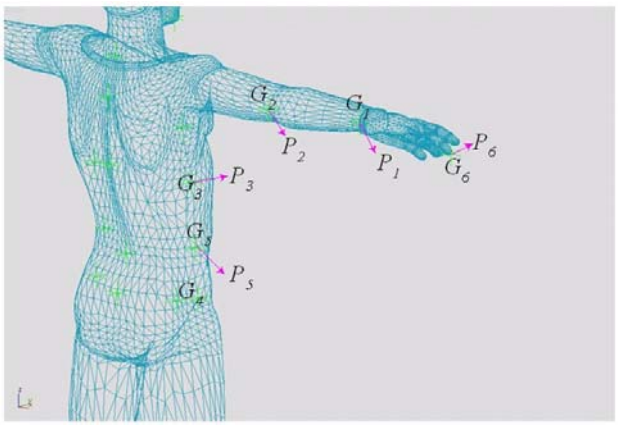


Figure 4.4. Example showing some deformation vectors resulted from 3D reconstruction

Readers may notice the analogy of our calibration/reconstruction algorithm to the standard bundle adjustment [145]. Bundle adjustment had been termed as a process of refining a visual reconstruction that describes the 3D scene geometry and the parameters of its acquisition optical devices. Bundle adjustment assumes 3D scene geometry and camera parameters are available for refinement. It is usually formulated into a non-linear optimization problem that represents its data in the form of a large sparse matrix. In contrast, our algorithm is formulated with the following characteristics:

1. Every feature point will have its respective reference on the surface of the generic model. These references serve as an important relationship for the deformation of 3D surface model and its underlying skeleton in the latter stages.
2. We do not assume that camera parameters are available for refinement, i.e. they have to be calibrated as part of this iterative algorithm (please see also setup 3 of Section 4.5.1.).
3. The initial 3D features that come from the generic model may be quite far away in distance from the real subject when both of these are superimposed onto each other (please see Figure 3.3). This depends on the size and shape of the subject.

4. We do not need to represent our data as a large sparse matrix.

From point 2 and point 3, our method can be viewed as more of a model adaptation algorithm, rather than just refining of variables as in standard bundle adjustment. In addition, our calibration and reconstruct processes are more strictly decoupled as compared to the classical bundle adjustment that was formulated into a nonlinear least-square optimization function.

4.4. Reconstruction by Interpolation of Deformations

Considering the reconstructed characteristic 3D model points, we notice that they are very sparsely distributed. These sparse points are not sufficient to represent the complete 3D model. Therefore, we make use of the sparse points in collaboration with the generic 3D model to complete the 3D model deformation via interpolation by radial basis function.

We have obtained a set of sparse deformation vectors in 3D from the preceding stage. Let $\overrightarrow{G_i P_i}$ be the i^{th} deformation vector, which sends G_i onto P_i (where G_i is the deformation center). Let us consider 3 functions, F_x, F_y, F_z: $\Re^3 \rightarrow \Re$, which, in each point in space, returns a deformation value respectively along axes X, Y and Z. We can formulate the radial basis functions from the 3 constraints F_x, F_y and F_z, so that the deformation defined for each center G_i is exactly $\overrightarrow{G_i P_i}$ and is interpolated elsewhere. In each point of space O, the deformation will be a linear combination of the deformations at the centers, according to the distance between O and each center. For N centers, the deformation along axis X will be:

$$F_x(o) = \sum_{i=1}^{N} W_{xi} \bullet \sigma(|o - o_i|) \tag{4.25}$$

where $\sigma(r)$ is the radial basis function, and A_{xi} is the weight with respect to the i^{th} deformation center, according to the x-axis. In order to obtain a smooth interpolation, which is not too local, we choose the simple norm:

$$\sigma(r) = r \tag{4.26}$$

This function is equivalent to the morphing function with can be considered as a special case of the RBF. It has been proven by Duchon [38] that the deformation obtained is then continuously differentiable.

We must define the function F_x, respectively F_y and F_z, so that $F_x(G_i)$ (respectively $F_y(G_i)$ and $F_z(G_i)$) gives the exact deformation from G_i to P_i along the X axis (respectively Y and Z axes), where $\overrightarrow{G_iP_i}$ is the i^{th} deformation vector. Thus, we obtain the set of following constraints:

$$\begin{aligned} F_x(G_i) &= \overrightarrow{G_iP_i}\,\mathbf{x} \\ F_y(G_i) &= \overrightarrow{G_iP_i}\,\mathbf{y} \\ F_z(G_i) &= \overrightarrow{G_iP_i}\,\mathbf{z} \end{aligned} \tag{4.27}$$

where $\overrightarrow{G_iP_i}\,\mathbf{x}$ (resp. $\overrightarrow{G_iP_i}\,\mathbf{y}$ and $\overrightarrow{G_iP_i}\,\mathbf{z}$) is the X coordinate component (resp. Y and Z coordinate) of $\overrightarrow{G_iP_i}$.

In the equation (4.15), we have N unknown variables for each direction X, Y, Z, i.e. 3 unknown weights $[W_{ix}\ \ W_{iy}\ \ W_{iy}]$ for each deformation center, and in (4.17), there are $N\times3$ constraints (N for each direction). This linear equations system can be written in the matrix form according to equations (4.15), (4.16) and (4.17):

$$\begin{bmatrix} \sigma(|o_1-o_1|) & \sigma(|o_1-o_2|) & \cdots\cdots & \sigma(|o_1-o_N|) \\ \sigma(|o_2-o_1|) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \sigma(|o_N-o_1|) & \cdots\cdots & \cdots\cdots & \sigma(|o_N-o_N|) \end{bmatrix} \begin{bmatrix} W_{x1} & W_{y1} & W_{z1} \\ W_{x2} & W_{y2} & W_{z2} \\ & \vdots & \\ W_{xN} & W_{yN} & W_{zN} \end{bmatrix} = \begin{bmatrix} \overrightarrow{GP}_{x1} & \overrightarrow{GP}_{y1} & \overrightarrow{GP}_{z1} \\ \overrightarrow{GP}_{x2} & \overrightarrow{GP}_{y2} & \overrightarrow{GP}_{z2} \\ & \vdots & \\ \overrightarrow{GP}_{xN} & \overrightarrow{GP}_{yN} & \overrightarrow{GP}_{zN} \end{bmatrix} \quad (4.28)$$

This system (4.18) is completely determined and unique, is in the form of:

$$\boldsymbol{\sigma W} = \boldsymbol{A} \quad (4.29)$$

Hence, this is solved with a least square linear algebra solution:

$$\boldsymbol{W} = \boldsymbol{\sigma}^{-1}\boldsymbol{A} \quad (4.30)$$

After having obtained the deformation weights $\boldsymbol{W}$, we can then use them to deform the rest of the model points using equation (4.15). This gives us a specific complete initial 3D external skin model of our subject.

The computation time of the matrix inverse rises exponentially with the size of matrix. We can notice that from equation (4.18) that the matrix '$\boldsymbol{\sigma}$' is symmetrical. Fast computation of the inverse of symmetrical matrix can be achieved by using the Bunch-Kaufmann factorization that is available from the CLAPACK [165]. It will highly accelerate the computation time especially when the size of the matrix is large.

4.5. Implementation, Results and Discussion

In this section, we show the results of executing our algorithm for two subjects of different shapes and sizes. These real images provided by the MIRAGES lab, INRIA, France are filmed for the Golf Stream project. All the image sizes are 720×576 pixels. One is the image set of the 'big man' who is a bit plumb and stands taller than 1.85 metres, while the other is the 'small man' who is much skinnier and about 1.7 metres in height. Moreover,

the ‘small man’ is standing with his feet close together as oppose to feet apart for the stanza posture. We show in our results that our algorithm is able to converge correctly for subjects of different sizes and shapes.

Our algorithm was implemented using C++ without any code or hardware optimization. The tests are conducted on an Intel Pentium IV processor.

4.5.1. Setups for checking results

There are 3 different setups that we will use to check and verify our results. Six images giving coverage of each subject were used for building its 3D model. We will assume that the interactively corresponded feature points are accurate and correct.

Setup 1:

Calibrated cameras – when all the cameras are calibrated, and the 2D characteristic points corresponded, then the 3D reconstruction is simply the triangulation of all these characteristic points. This setup is always stable and could be used as a reference.

Setup 2:

Begin by pre-positioning the 3D generic model to roughly align with all the cameras views, and then click on the 2D characteristic points. In other words, the camera poses were provided with an initial value. From the 2D corresponding features, in cooperation with the 3D feature points on the surface of the generic model, the execution starts with 3D reconstruction via triangulation of features from the coarse camera poses before iterating the camera calibration and 3D reconstruction.

Setup 3:

Similar to setup 2, from the 2D corresponding features, in cooperation with the 3D feature points on the surface of the generic model, the camera calibration and 3D reconstruction

loop iterates until convergence is reached. In contrast with setup 2, camera calibration is executed first before 3D reconstruction takes place, and the calibration/reconstruction algorithm iterates. Thus, in this setup, no initial camera poses were used. This setup has the most uncertainties.

The RBF deformations take place after the respective setups have converged.

4.5.2. Visual Analysis

Figure 4.6 shows the top-views of the scenes for both subjects from setup 3. We can see that the relative positions and orientations of the objects in the scenes are very similar. These results are also very similar to those using setups 1 and 2.

We can back-project the reconstructed 3D model onto the 2D images by using the calibrated camera geometries to visualize the differences. Figures 4.7 and 4.8 are the results of the 3D reconstruction of both the subjects. The results are very similar for all the three setups. We can observe that the global shape of the subjects, when overlaid onto the background images from the different views are very close to each other. However, the limbs of the subjects are not always properly fitted, due to the fact that sparse feature points do not contain enough information to make up the local surface of the human body (see figures 4.9 and 4.10).

4.5.3. Quantitative Analysis

Reprojection Error

The reconstructed 3D features are back-projected onto the 2D imaging plane via the calibrated camera geometries to compare the differences with the hand-clicked points in the images. Figure 4.11 shows the typical convergence using setup 2 (blue label) and setup

3 (red label). As a result of initial pre-positioning of the 3D puppet, setup 2 took fewer iterations to converge. The algorithm usually takes less than 20 iterations to converge, which is less than 100msec. The typical average re-projection error is about 1.4 pixels with a standard deviation of 0.7 pixel.

Comparison Between the Different Setups

3D model reconstruction from un-calibrated cameras can recover the 3D object geometry up to a relative size and orientation. We will compare the quantitative measurement of the shape and size of the models reconstructed from the three different setups. Since we already knew the indices of the model vertices, a simple way to compare their 3D corresponding relative positions can be done by aligning their relative translations and orientations.

Given 2 sets of 3D points representing 2 objects that we already knew their correspondent topological relationship, the distribution and location of the 3D points with respect to their principal axes should be similar (see Figure 4.12). In other words, we can transform the 3D points of each human models onto their respect principal axes (PCA transformation), and then sum up the magnitude of the positional differences.

For a set of 3D points **A**, which stacks all the 3D points as column vectors, its principal axes are the 3 eigenvectors of the 3×3 covariance matrix **C**, computed as:

$$\mathbf{C} = \frac{1}{N}\sum_{i=0}^{N}(\mathbf{A}-\overline{\mathbf{A}})(\mathbf{A}-\overline{\mathbf{A}})^{T} \tag{4.31}$$

where $\overline{\mathbf{A}}$ is the mean or centroid of the point-set **A** and N is the total number of 3D points. Then the eigenvectors $\mathbf{v}_i$ and its corresponding eigenvalues λ_i can be calculated from the formulation:

$$\mathbf{Cv} = \lambda\mathbf{v} \tag{4.32}$$

The PCA transformation is performed on all the 3D points given as

$$\mathbf{A'} = \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \mathbf{v}_3^T \end{bmatrix} \bullet (\mathbf{A} - \overline{\mathbf{A}}) \tag{4.33}$$

The square roots of their eigenvalues are the distributions of the data in the respective principal axes. In other words, the difference in scale between the two models is in fact the ratio between their eigenvalues i.e. if the ratios between the 3 eigenvalues are similar, then the 2 model shapes can be uniformly scaled. The detailed derivation of PCA can be found in references such as [64].

Comparing the Reconstructed 3D Features and Mesh

We transformed the 3D points reconstructed from the 3 different setups into their respective principal axes for comparison (using equation 4.23).

Table 4.1 shows the typical average distance errors and standard deviations for comparing the relative shapes and sizes of the features reconstructed from the 3 different setups. Since the depth of the cameras and the size of the model are adapted to each other, the size of the 3D model built using the different setups will be some simple scaling by a factor. Thus, all the 3D points are scaled by the average of the 3 ratios of their square-rooted eigenvalues i.e. from eigenvalues of setup 1 = $\{\lambda_1, \lambda_2, \lambda_3\}$, eigenvalues of setup 2 = $\{\kappa_1, \kappa_2, \kappa_3\}$, therefore the scale *s*, is calculated as:

$$s = \frac{1}{3}\sum_{i=1}^{3}\frac{\lambda_i}{\kappa_i} \tag{4.34}$$

Table 4.1 also shows the scaling factors for the different setups. The same method is used for comparing the results after RBF deformation (see table 4.1).

We had observed that the ratios for eigenvalues between each setup in all the cases are less than 1.8% error, which means they are fairly uniformly scaled, therefore the formulation in equation (4.24) holds. From the results, we can see that the differences are relatively small and will be hardly visually noticeable from the cameras, which were placed from a distance.

Table 4.1 Typical errors for comparing the reconstructed subject via different setups

	Positional error and standard deviation for:		Scale factor
	3D feature points	Model mesh vertices	
Setup 1 vs setup 2	0.93 cm, 0.38 cm	0.85 cm, 0.14 cm	1.0074
Setup 1 vs setup 3	0.91 cm, 0.37 cm	0.879 cm, 0.405 cm	1.0692
Setup 2 vs setup 3	0.19 cm, 0.034 cm	0.16 cm, 0.0756 cm	1.0614

Table 4.2. Typical distances of each camera measured to the centroid of the 3D model

	Camera 1	Camera 2	Camera 3	Camera 4	Camera 5	Camera 6
Distance from	17.92 m	7.92 m	5.36 m	7.89 m	5.59 m	6.16 m

Table 4.2 shows the typical distance between each camera measured to the centroid of the 3D model. The same method for checking the positional differences for feature points used in the previous sub-section is used for checking the relative position of the cameras estimated using the different setups. An example setup of relative directional differences between the cameras is shown in table 4.3 (setup 1). Each relative direction is the angle

measured between 2 optical axes i.e. pointing direction of the two cameras. Table 4.4 compares the mean differences between the relative positional and viewing direction of the cameras from the three setups.

Table 4.3. Example of relative directional differences between the cameras (e.g. setup 1)

Angle between	Camera 1	Camera 2	Camera 3	Camera 4	Camera 5	Camera 6
Camera 1	----	89.6°	176.9°	137.2°	89.7°	37.74°
Camera 2	89.6°	----	91.3°	133.2°	173.1°	124.08°
Camera 3	176.9°	91.3°	----	41.97°	89.04°	142.46°
Camera 4	137.2°	133.2°	41.97°	----	47.9°	101.31°
Camera 5	89.7°	173.1°	89.04°	47.9°	----	57.45°
Camera 6	37.74°	124.08°	142.46°	101.31°	57.45°	----

Table 4.4. Error comparison of the camera poses via different setups

	Positional difference and standard deviation	Rotational difference and standard deviation
Setup 1 vs setup 2	0.279 m, 0.11m	2.508°, 2.25°
Setup 1 vs setup 3	0.293m, 0.10m	2.581°, 2.09°
Setup 2 vs setup 3	0.034m, 0.021m	0.231°, 0.29°

From the comparison between the results of the cameras position and viewing direction, we can see that the amount of error is relatively low, considering the inter-relationship between the camera poses and 3D model that we have to solve. Hence, at this stage, it is suffice that the camera calibration and 3D reconstruction are stable and consistent.

Uncertainties from 2D Image Correspondents

This sub-section presents an empirical study assuming the interactive selections of feature points on the 2D images are not perfect (which is really the case!). We will analyze the outcome of the camera calibration and 3D reconstruction given these uncertainties. The imperfection is simulated by adding random white pixels noise to the 2D feature points.

Figure 4.13 presents the plots for the convergence of the algorithm for the re-projected pixel error plot versus the number of iterations when the algorithm is perturbed by various amount of pixel noise. Table 4.5 shows the errors in the 3D model reconstruction due to noises in setup 3 as we make comparison with the 'noiseless' setup 1. Then on table 4.6 we show the positional and rotational differences due to the noise.

Table 4.5. Error in the 3D model due to noises in setup 3 when comparing with setup 1

	Reprojection error (pixels) and standard deviation	Scale factor	Positional error and standard deviation for:	
			3D feature points	Model mesh vertices
No noise	1.465, 0.86	1.0692	0.91 cm, 0.37 cm	0.85 cm, 0.33 cm
1 pixel noise	1.465, 0.87	1.0705	0.956 cm, 0.4199 cm	1.072 cm, 0.462 cm
2 pixels noise	1.504, 0.89	1.073	1.099 cm, 0.519 cm	1.318 cm, 0.545 cm
4 pixels noise	1.697, 0.96	1.0763	1.431 cm, 0.7214 cm	1.876 cm, 0.75 cm
6 pixels noise	2.000, 1.12	1.0791	11.232cm, 10.516 cm	2.473 cm, 0.981 cm
8 pixels noise	2.379, 1.33	1.081	11.35 cm, 10.53 cm	3.071 cm, 1.22 cm

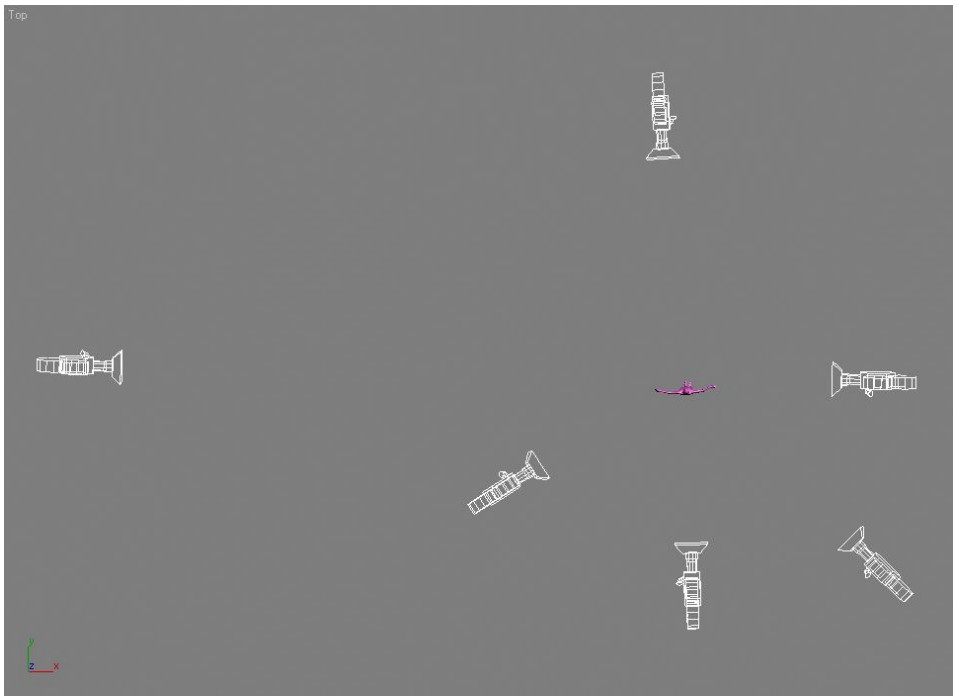
Table 4.6. Error in the camera poses due to noises in setup 3 when comparing with setup 1

	Positional difference and standard deviation	Rotational difference and standard deviation
No noise	0.293 m, 0.10 m	2.581°, 2.25°
1 pixel noise	0.321m, 0.119 m	2.91°, 2.48°
2 pixels noise	0.34 m, 0.146 m	3.23°, 3.02°
4 pixels noise	0.378 m, 0.22 m	4.01°, 4.03°
6 pixels noise	0.43 m, 0.259 m	4.92°, 5.05°
8 pixels noise	0.485 m, 0.304 m	5.78°, 6.01°

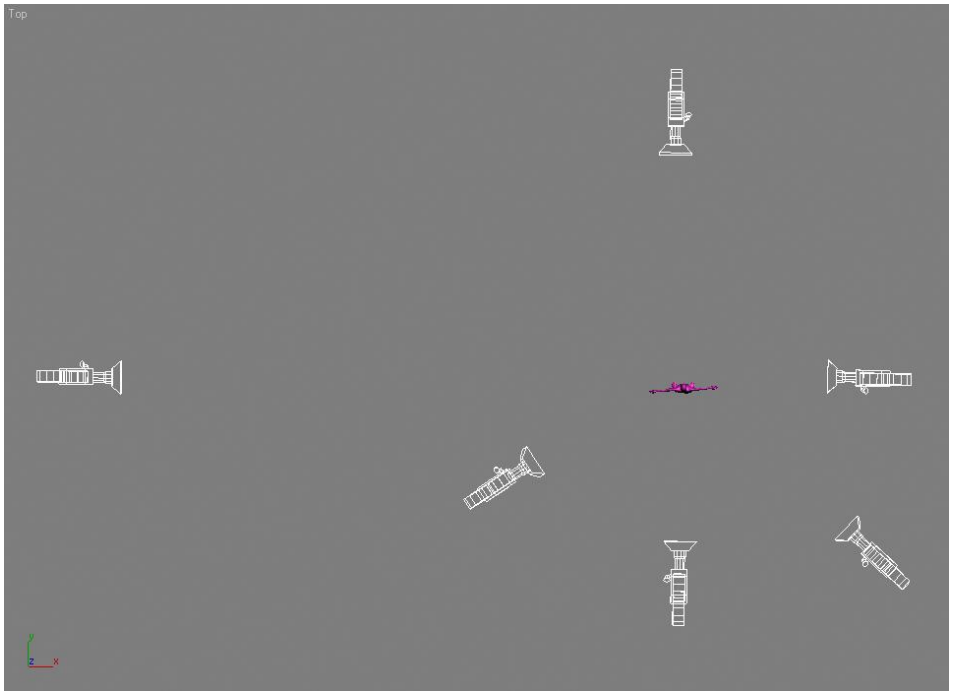
4.6. Concluding Summary

In this chapter, we showed that with un-calibrated images from wide baseline, our method, which comprises iterative camera calibration / 3D reconstruction and RBF deformation, is able to produce customized 3D puppet models of human subjects of different sizes. Given the number of unknown parameters to be solved are inter-related, and containing many possible local minima, there might be a slight difference between the relative camera poses. This could be due to the variation of the subjects' sizes. Nevertheless, the results are fairly consistent. The uncertainties caused by pixel noises in the 2D feature points were also investigated. We conclude that our algorithm can cope with errors of up to 4 pixels.

It is clear that 3D reconstruction using feature points are inadequate to build a whole model due to the sparseness. Nevertheless, at this stage we have a 3D model that is closer to the subject, called the intermediate 3D model. Further improvement to the intermediate model could be made through using the silhouette limbs, which will be explained in details in the following two chapters.



(a) Top-view of ‘small-man’



(b) Top-view of ‘big-man’

Figure 4.5. Top-view scenes of the final results of calibration/reconstruction obtained for two subjects of different size and shape

Figure 4.6. 3D Model of the 'big-man' superimposed onto the images from 6 different views

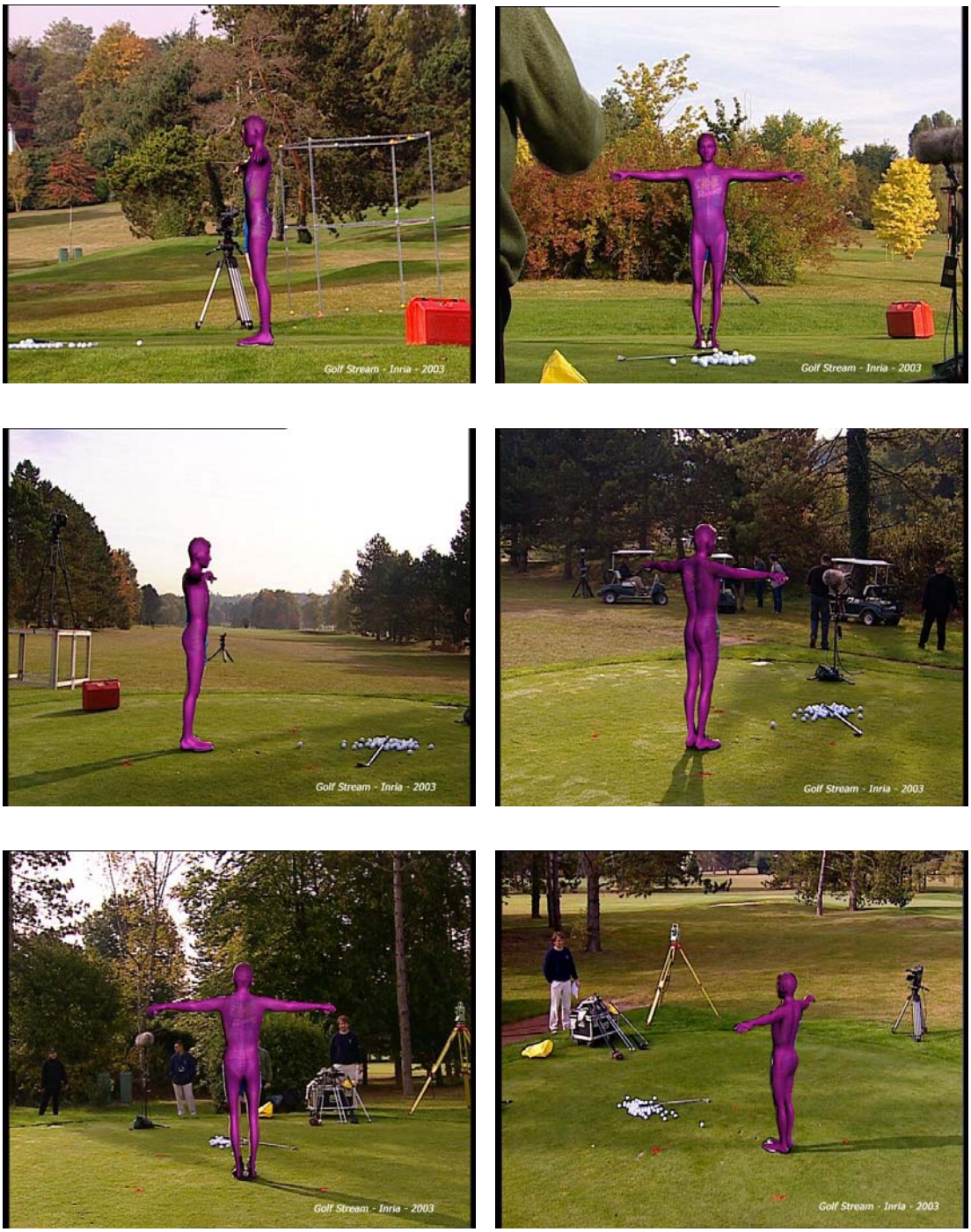
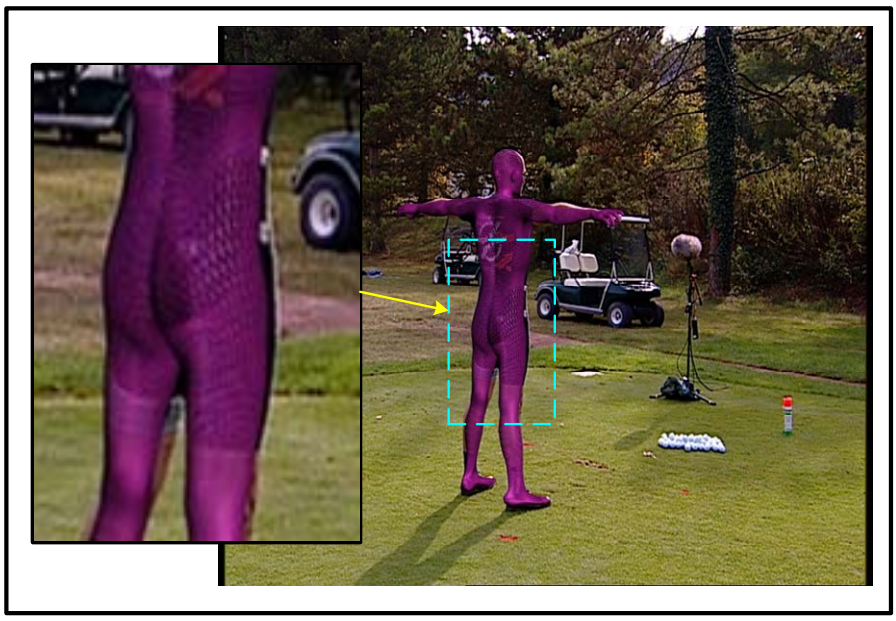
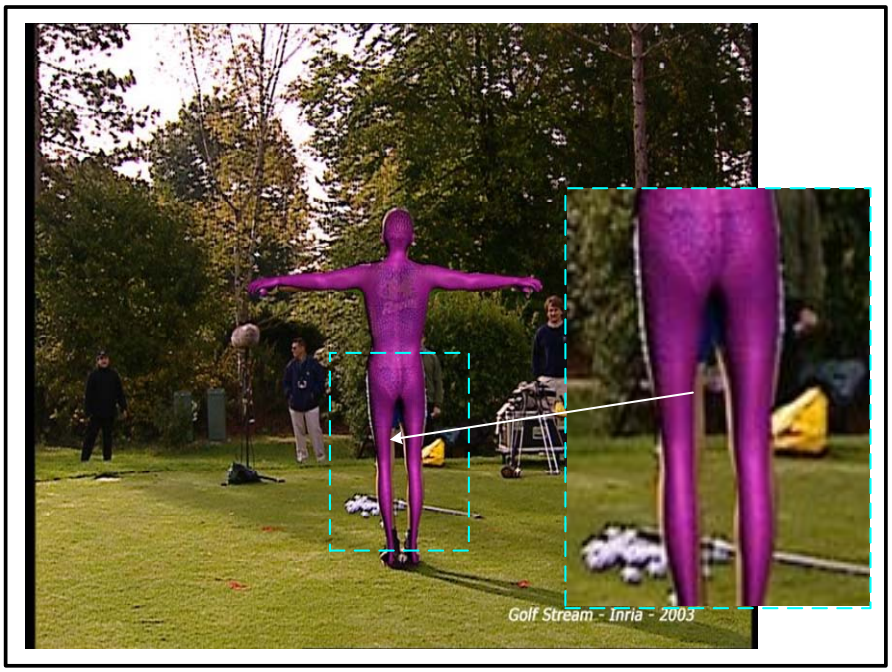


Figure 4.7. 3D Model of the ‘small-man’ superimposed onto the images from 6 different views



(a)



(b)

Figure 4.8. Visual results indicating that the 3D reconstruction from feature points is not enough, even though the global shapes and sizes are fine

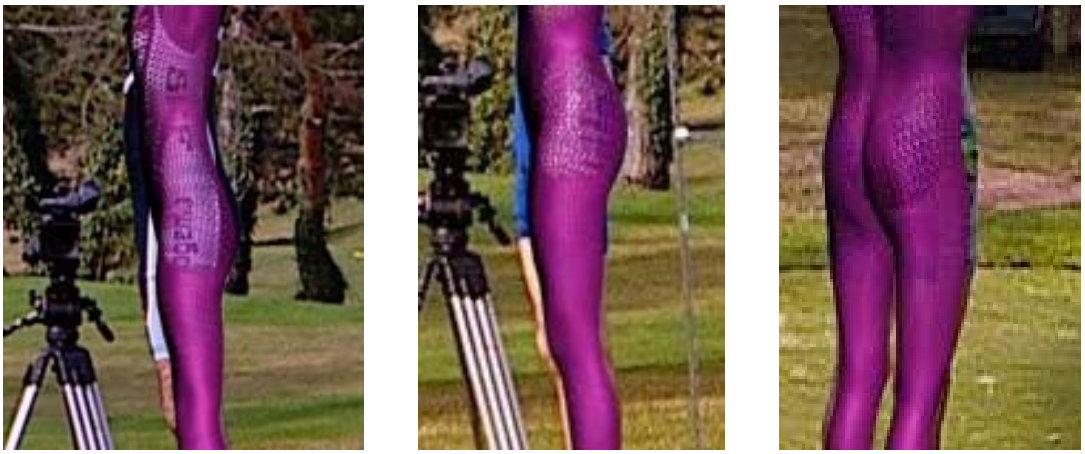


Figure 4.9. More examples showing 3D puppet not properly fitted via reconstruction from feature points

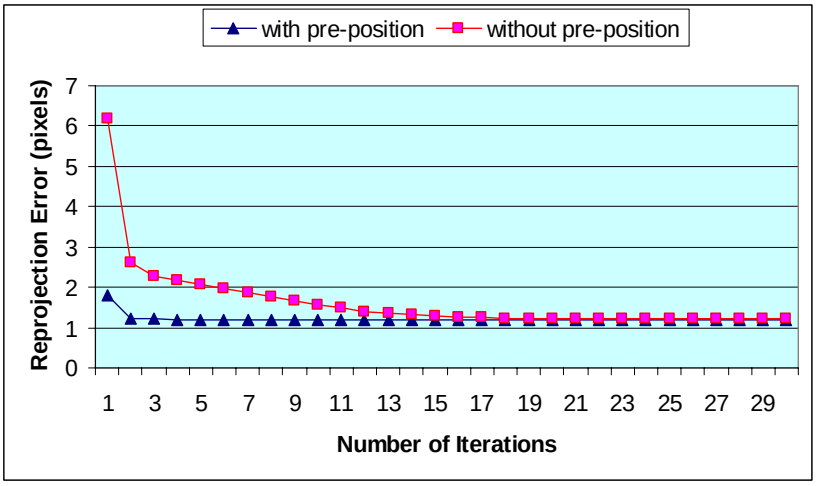


Figure 4.10. Typical convergence of the calibration/reconstruction iterations

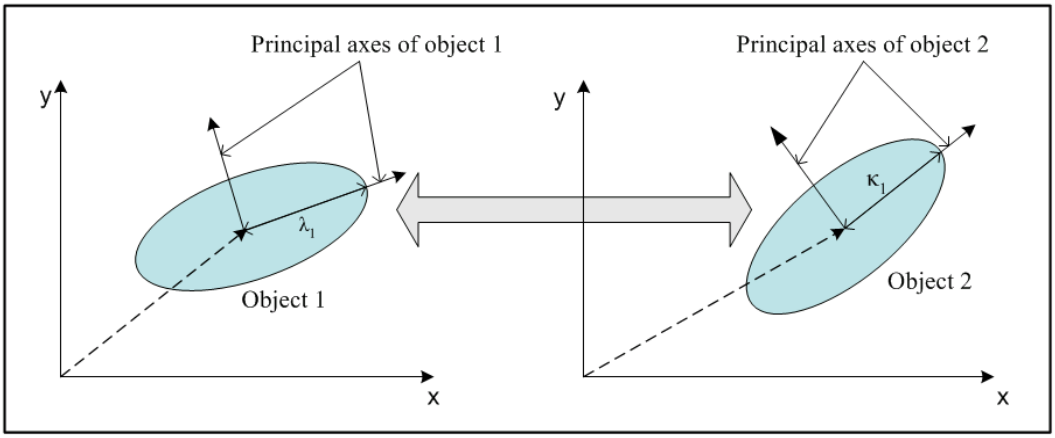


Figure 4.11. Comparing 2 shapes via principal component transformation (2D illustration)

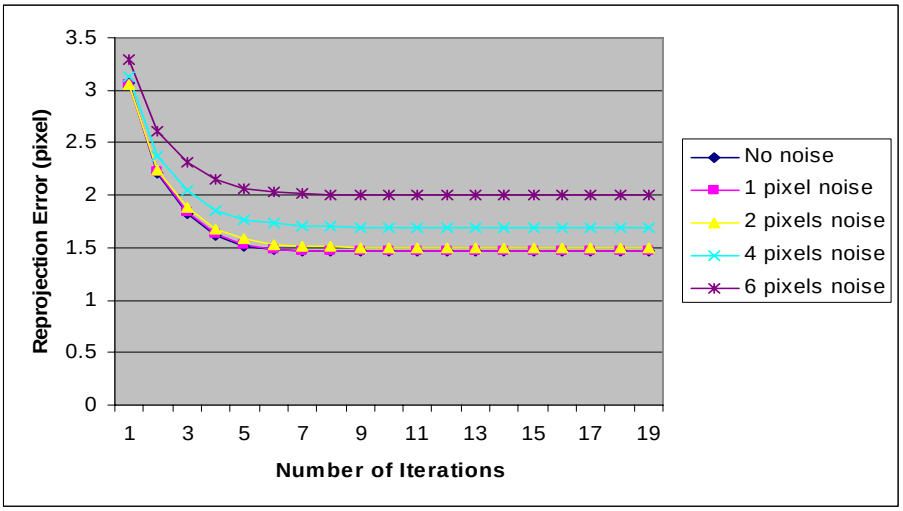


Figure 4.12. Convergence of the calibration/reconstruction iterations when perturbed with random white noise

Figure 4.13. Results for subjects of different shapes and sizes (with 2 pixels noises) – superimposed on images from different views

Figure 4.14. Results for subjects of different shapes and sizes (with 4 pixels noises) – superimposed on images from different views

Chapter 5.

Silhouette Extraction from Images and 3D Model

To improve on the results of the 3D reconstruction from feature points, we can use the silhouette limb information. It is known that the complete set of limbs obtained from all viewing directions allows us to fully reconstruct the 3D shape and even from a small set of views, the use of curves is much richer than points. The multi-view silhouettes of the intermediate puppet model can be improved by registering them to the subject's silhouettes in the real images. This chapter explains how the silhouettes are extracted from the intermediate 3D model and real images. Both sets of information are continuously linked closed contours in image plane (but NOT IN 3D!) and are described by their respective ordered lists.

5.1. Defining the Terminologies

Let us first define the terminologies that we use to describe our 3D model in this chapter, and that will be used for the rest of the dissertation.

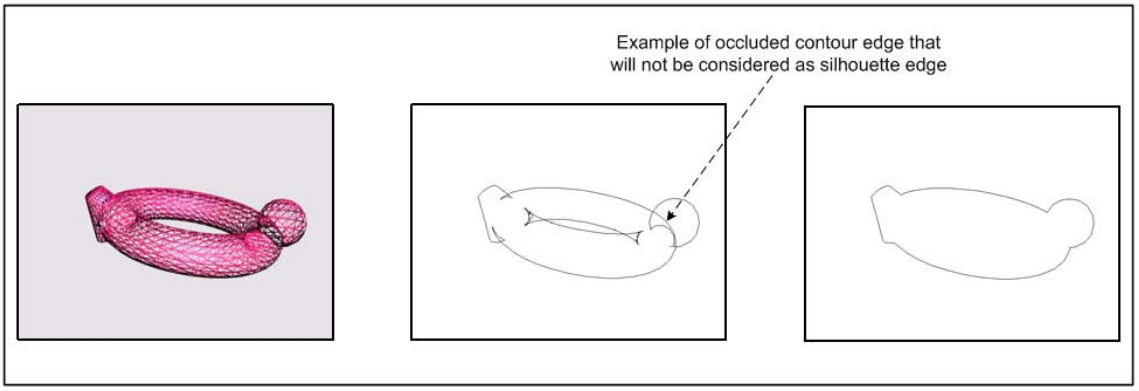
Contour Edge

A contour edge [74] is defined as an edge that is made up of exactly one front facet and one back facet with respect to some viewing direction to the imaging plane. A contour edge may be visible, occluded or partially occluded (see Figure 5.1b). Here, in addition, we

do not consider edges that only have one facet since we are assuming that the model is a closed solid.

Silhouette Edge

A silhouette edge is a subset of a contour edge. The difference between a silhouette edge and a contour edge is that the silhouette edge is the visible part of a contour edge that defines the outside boundary of a closed solid object (with respect to the camera viewpoint, see Figure 5.1c).



(a) (b) (c)

Figure 5.1. Example of (b) contour edge, and (c) external silhouette edge of (a) wire-frame of an object

5.2. Extracting 3D Model Silhouette

From the calibrated cameras, we seek for an ordered list of vertices that form the silhouettes of the 3D model when it is projected onto the respective imaging planes. In the example on Figure 5.2, we must obtain the list of the pink points starting from the projected edges (in blue). Raster image-based methods such as the Z-buffer are unsuitable because they do not provide the 3D geometric topological path of the silhouette along the surface of the 3D model. Detailed background on extracting silhouette from 3D geometric

model may be referred to [67]. In definition, we need to use an object space silhouette tracing algorithm so that we can obtain the 3D silhouette path that travels along the surface of the 3D model.

In this section, we propose a simple method to extract the silhouette vertices of the subject seen in the multiple cameras. The algorithm that we have developed requires the 3D model to have regular surface i.e. without any open edge and be properly triangulated in its 3D mesh data structure. The main steps involve:

1) Finding all the contour edges.

2) Starting with a silhouette vertex.

3) Tracing along the silhouette edges in a counter-clockwise direction until they meet an intersection with a contour edge (which is also a silhouette edge).

4) Moving to the new silhouette edge (via step 3) and continue tracing until it returns to the starting point.

Our algorithm finds the contour edges first so that the possible edges for searching are highly reduced.

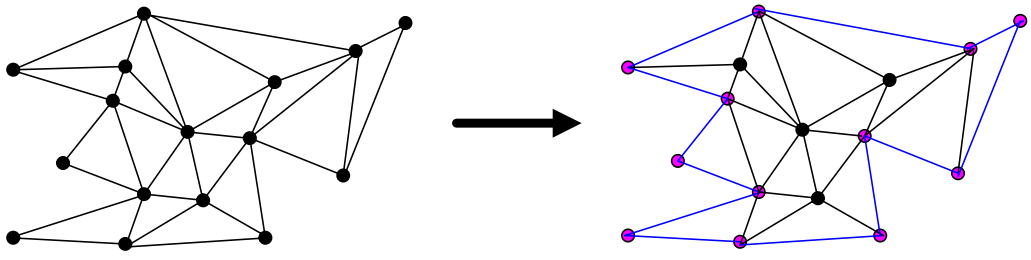


Figure 5.2. Silhouette vertices of a model projection

5.2.1. Tracing the Silhouette

Here we describe our algorithm for creating a list of 3D vertices belonging to a projected model silhouette:

Step 1: Finding contour edges:

- A contour edge has been defined as an edge with exactly one front facet and one back facet with respect to the camera projection.
- We can obtain all the contour edges of a model in a two-pass algorithm by checking all the edges and their respective facets.
- First, we use the camera projection parameters to determine whether a facet is front facing or back facing for all the facets in the model. This is done in a single pass process. The facet information will be stored in a bit-array. A front facet will be stored as a binary '1' whereas for back facet '0'.
- Every edge will consist of two facets since the surface is regular. We store for each edge its pointer index to the facet information in the bit-array. This enables us to check for contour edge efficiently using the XOR bitwise operation. We only scan the model edges in one single pass to get the contour edges of the model.

Step 2: Initialization of silhouette edge tracing:

- The starting point of the silhouette edge tracing must be a silhouette vertex.
- Then we seek the vertex point that, when projected onto the 2D image, gives an extreme value in the Y-coordinate (as compared to the rest of the model vertices). This vertex

point will be the starting point, denoted point of reference P_i (Figure 5.3). This is also a one-pass algorithm which scans all the model vertices.

• An initial reference vector V_i (in blue) is created.

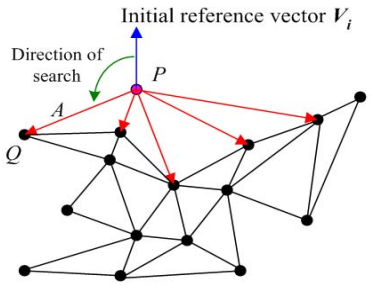


Figure 5.3. Initial stage of the algorithm leading to the next edge

Step 3: Search for subsequent point of silhouette: (*Figure 5.3*)

• From the entire projected candidate edges (in red) related to point P_i, we find the edge *A*. The projected edge *A* (as compared to all the other candidates) will form the smallest angle in the anti-clockwise direction with respect to the initial vector of reference V_i. In reality, we will only need to scan the neighboring edges that are contour edges only.

Step 4: Updating the parameters: (*Figure 5.4*)

• Now the edge *A* takes over as the vector of reference V_i.

• The point of reference P_i is moved to point *Q*.

• Setting the direction of the vector of reference V_i from the point of reference toward its previous projected silhouette point, we search for the next silhouette vertex by going through all the projected candidate vertices in the anti-clockwise manner.

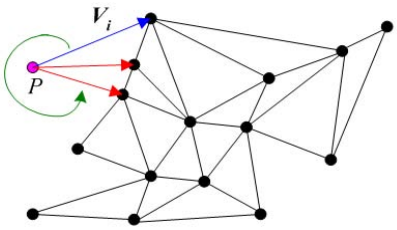


Figure 5.4. Tracing of subsequent vertex

Step 5: Iterating and ending:

• We iterate step 3 and step 4 until the current point of reference is the start point.

5.2.2. Intersection of Silhouette Edges

In the sub-section 5.2.1, when we are running steps 2 to 5 of our algorithm, we may encounter the situation whereby the projection of 2 silhouette edges intersecting with each other. Since our model is a highly concave, there will be discontinuities in 3D depth along the silhouette, and since it is a discrete 3D triangular mesh, there would be intersections of edges when we project them back onto the 2D images. This is illustrated in Figure 5.6a, a typical case where intersection occurs, and on the real scenario in Figure 5.5. In Figure 5.6b and Figure 5.6c, we illustrate some of the examples of intersections of projected edges that we have to consider. The shaded part (in green) belongs to the projected facets of the model. The blue arrowed lines are the silhouette edges and the brown arrowed lines are the contour edges.

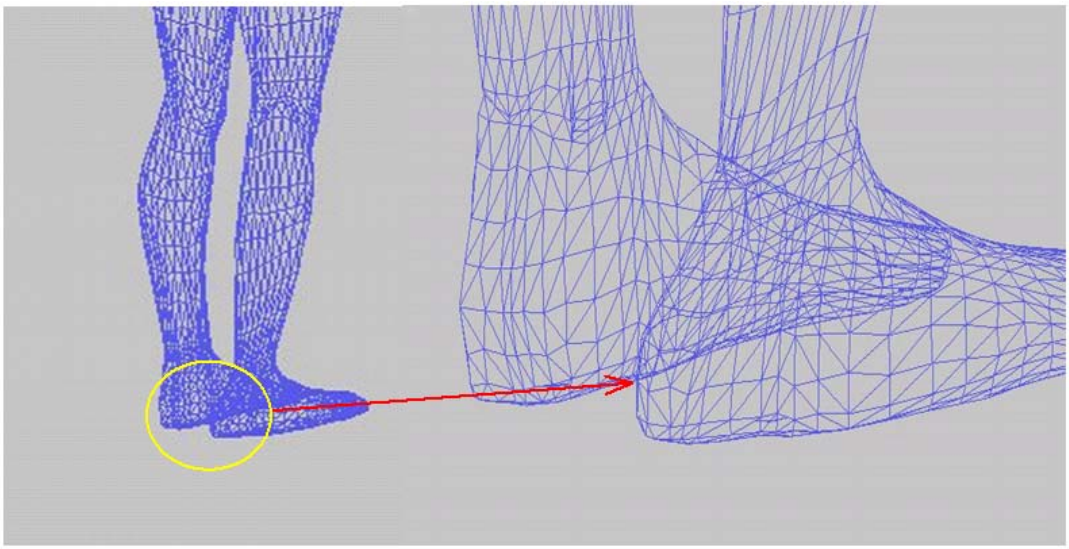


Figure 5.5. Real scenario of a typical intersection of projected silhouette edges

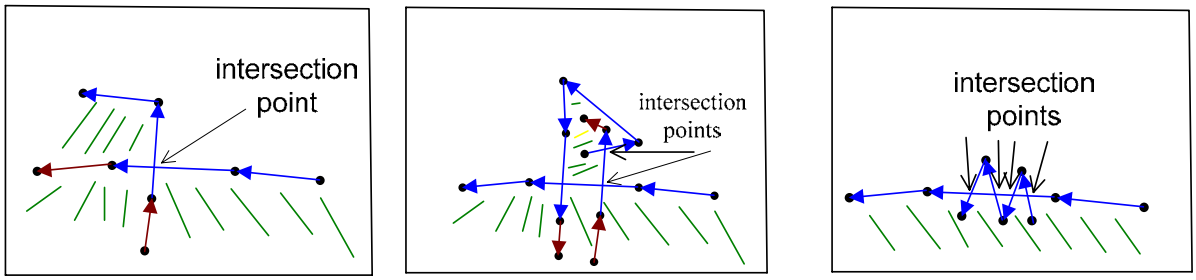


Figure 5.6. (a) typical intersection of projected edges, (b) and (c) 2 particular cases of intersections (note: the shaded parts, in green, belong to the body of the model)

When tracing the silhouette vertices via the path of silhouette edges, we know that the edge represented by the vector of reference V_i can only make intersection with contour edges. We can also assume that for our 3D model, the number of contour edges is relatively small. Here are the details when we are tracing the silhouette vertices:

1) For each instance of the vector of reference V_i along the tracing path, we check for all the possible intersections with all the contour edges. To improve on the

searching efficiency, we do not need to check for possible contour intersection for those contour edges that are already been traced as silhouette edges.

2) As illustrated in Figure 5.6, it is possible to obtain more than 1 intersection point. In this case, the intersection point will be taken as the point that has the smallest projected distance to the point of reference P_i. The corresponding contour edge that gives rise to this intersection point is the *intersected edge B*.

3) Each intersection computation will give us 2 points that have to be determined in 3D. The first one is the new end point on the edge represented by the vector of reference V_i. The second will be the point of edge *B* intersected by the other edge point. The 3D coordinates of these 2 points can be calculated using equation (5.4) given their projected intersection points in 2D. Then, we remove both edges from the list of edges and replace them with four edges, by adding the intersection point to each edge, and so on recursively until there is no more intersection.

4) Steps 3 and 4 (from Section 5.2.1) can then be run without any problem until the tracing of the complete silhouette is obtained.

To further improve the speed of this silhouette tracing algorithm, we can partition the projected contours edges into a 2D bucket array to further reduce the search candidates.

5.2.3. Computing the 3D Silhouette Intersection Points

Now we will present the equation for the computation of the two 3D points along the two intersected edges when projected onto the image plane. From the projected edges, we have an intersection point in 2D at position (u_s, v_s). For each edge, we have the two vertices (at the 3D positions $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$) which project onto the image plane, and we know that the

projected intersection point in 3D $\boldsymbol{W_s}$ must lie along the edge. We also know the 3×3 rotation matrix $\boldsymbol{R}$ and 3×1 translation matrix $\boldsymbol{C}$ of the camera body in the world coordinates. Figure 5.7 shows a simple illustration.

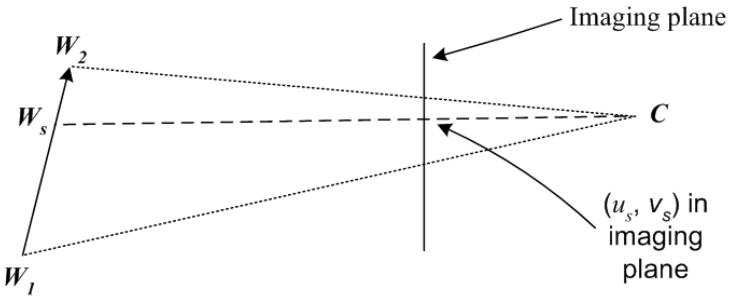


Figure 5.7. Geometry illustrating how to obtain the 3D point via projected edge intersection point

Given that the rotational 3×3 matrix $\boldsymbol{R}$ can be expressed as:

$$\boldsymbol{R} = \begin{pmatrix} r_1^T \\ r_2^T \\ r_3^T \end{pmatrix} \tag{5.1}$$

and

$$\boldsymbol{W_s} = \lambda(\boldsymbol{W_2} - \boldsymbol{W_1}) + \boldsymbol{W_1} \tag{5.2}$$

from

$$u_s = \frac{r_1^T(\lambda(W_2 - W_1) + W_1 - C)}{r_3^T(\lambda(W_2 - W_1) + W_1 - C)} \tag{5.3}$$

we can solve λ by evaluating equation (4.2.3) as:

$$\lambda = \frac{(W_1 - C)(r_1^T - r_3^T)}{(W_2 - W_1)(r_3^T u_s - r_1)} \tag{5.4}$$

We will put λ back into equation (5.2) to get the projected intersection point in 3D, $\boldsymbol{W_s}$.

5.3. Extracting Silhouette Pixels from Images

The segmentation and extraction of silhouette pixels from real 2D images may be done either in an automatic way using edge detection, or interactive way. Many edge detection algorithms for image segmentation had been proposed over the decades e.g. Canny filter [21]. However, using edge detection to segment a continuous close contour of the subject from any cluttered and noisy image is impossible. The only way to achieve this is to acquire the images in a very well controlled environment:

1. Make the subject wear special coloured cloth.
2. Make the background colour of the scene to be contrasting to the foreground subject.
3. Ensure a very good illumination setup.

If a very well controlled environment is unlikely, then we have to resort to bring out the silhouette features interactively. We can make use of the curve digitizing tools (e.g. Bezier curves drawing) available from common commercial software programs like the Adobe Photoshop.

We will perform edge-linking after we have obtained the digitized contours using any one of the above-mentioned methods. Edge-linking is a procedure where we link neighbouring contour pixels. This is necessary so that topological information is maintained when we will have to match the two sets silhouette curves.

5.4. Results

The outcome of the silhouette extraction is the 2 sets of information: (1) the ordered list of 3D vertices and their respective projected pixel location in the 2D images that made up the silhouette of the intermediate 3D model, and (2) ordered lists of continuously linked 2D pixels that form the silhouette of the subject in the real images. These 2 sets of information will be registered with each other and then used for refining the intermediate 3D model.

Figures 5.8 and 5.9 show the silhouettes extracted from the intermediate 3D models of the subjects (shown in figures 4.6 and 4.7) from 6 different views. Figure 5.10 shows the close-up views of the silhouette path traced along the surface of the 3D model seen in different views. Some of the edges of silhouette are jagged since the 3D model is a discrete mesh and has 3D discontinuity on the model's surface (see Figure 5.5 and Figure 5.6). The blue curves show the continuous paths along the model surface, while the yellow line shows the discontinuity. Figures 5.11 and 5.12 show the silhouettes (in red) that are traced along the limbs of the subjects and are superimposed onto their respective images. Each model silhouette is made up by about 1000 to 2000 vertices.

The computing time (without any algorithmic and hardware optimization) to extract the silhouette from the intermediate 3D model is less than one second for each camera view. Should edge detection e.g. Canny filter be used for extracting silhouette from the 2D images, the computing time will not be significant given any modern day computers. Therefore, the total computation needs to extract the silhouettes from both the 3D human model and the real images could be easily fulfilled.

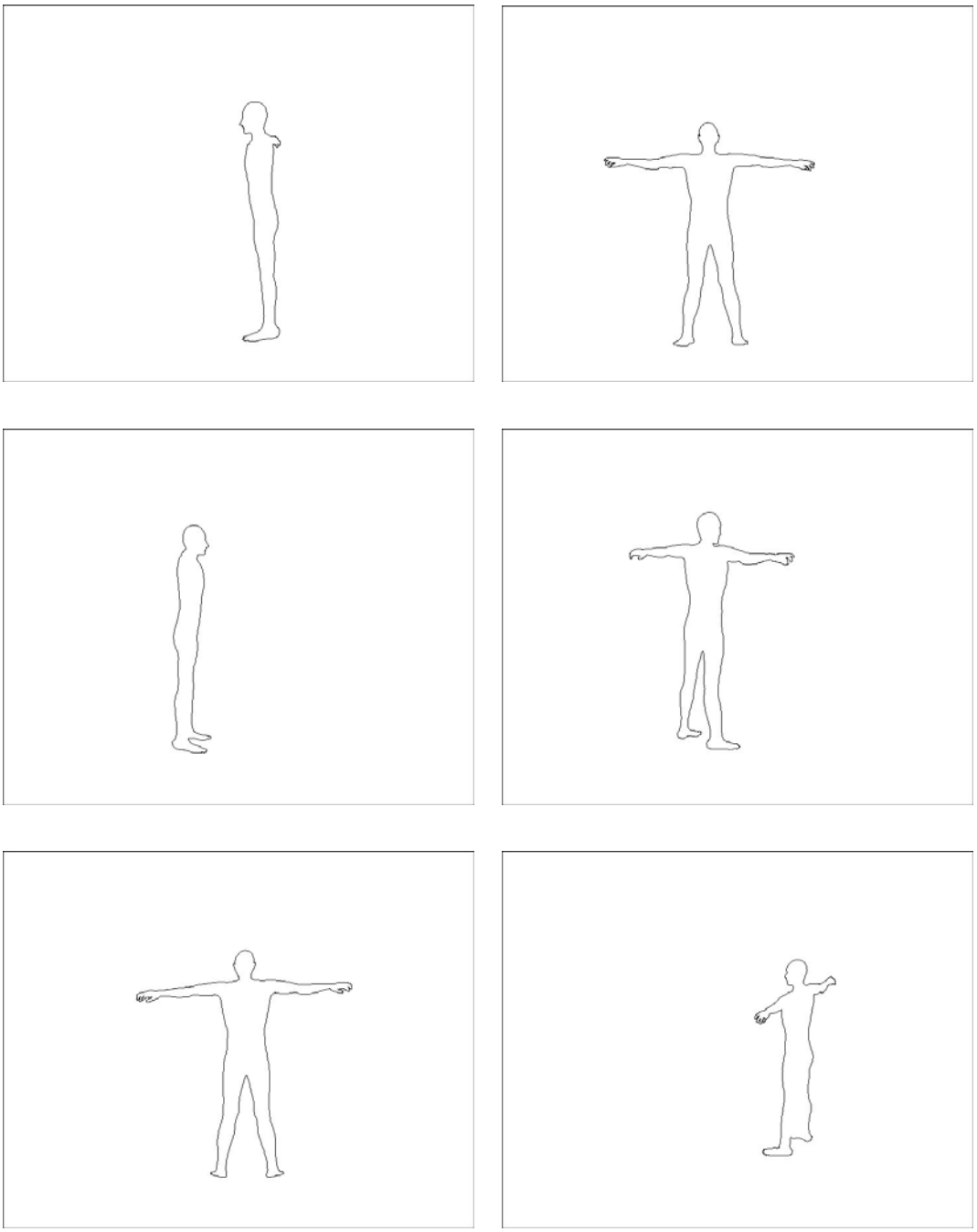


Figure 5.8. Silhouettes extracted from the intermediate 3D model of the ‘big-man’ seen in different cameras

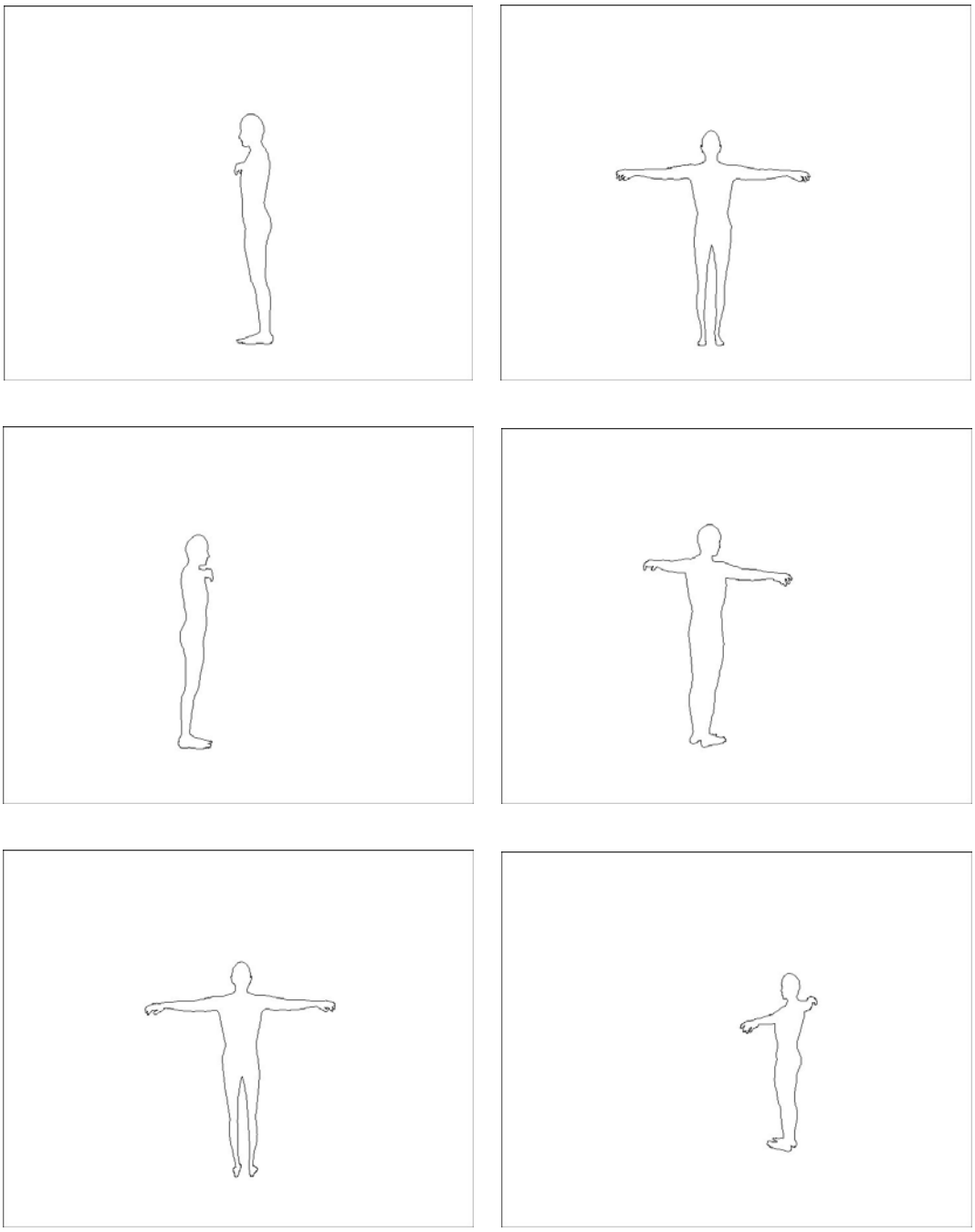
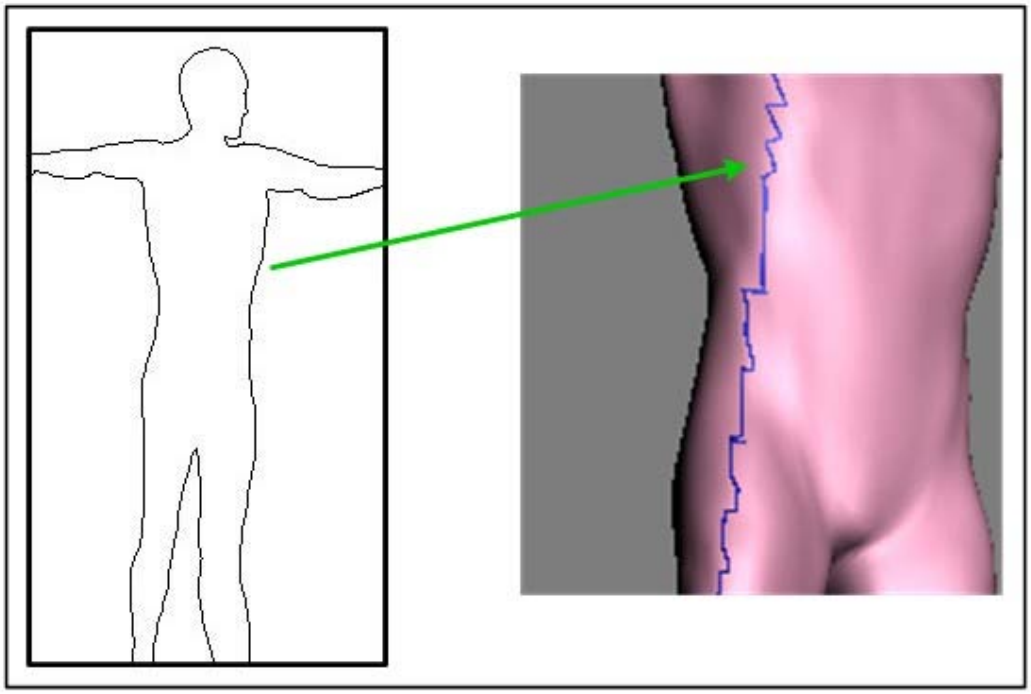
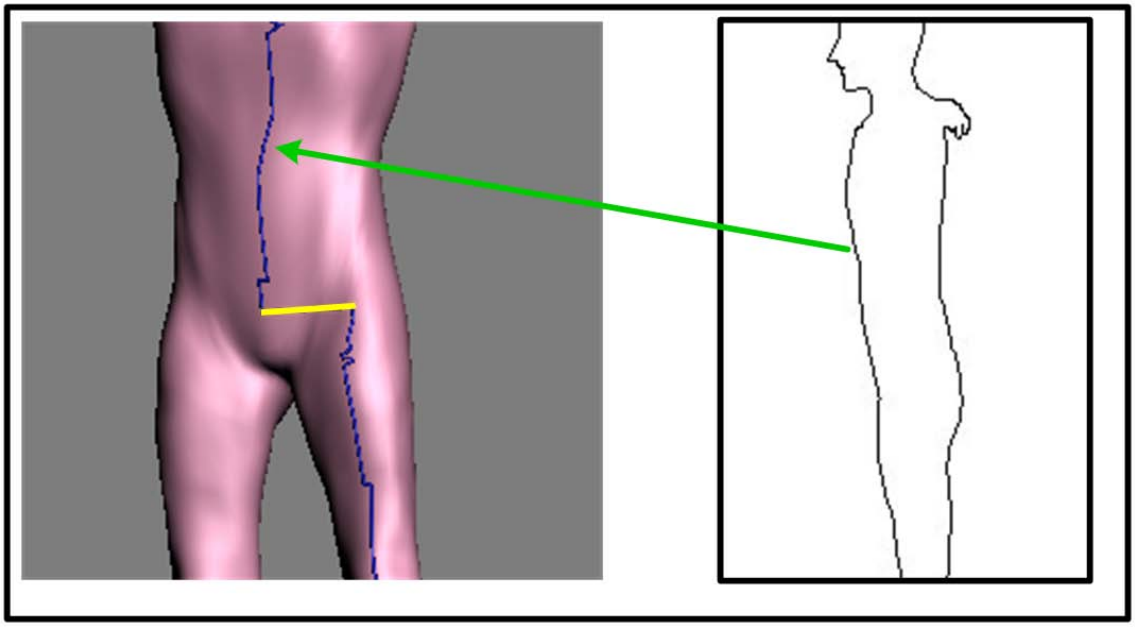


Figure 5.9. Silhouettes extracted from the intermediate 3D model of the ‘small-man’ seen in different cameras



(a)



(b)

Figure 5.10. Close-up views of the silhouette path (blue curves) traced along the surface of the 3D model seen in the different views

Figure 5.11. Silhouette (in red) of the 'big-man' for different camera views in the real images

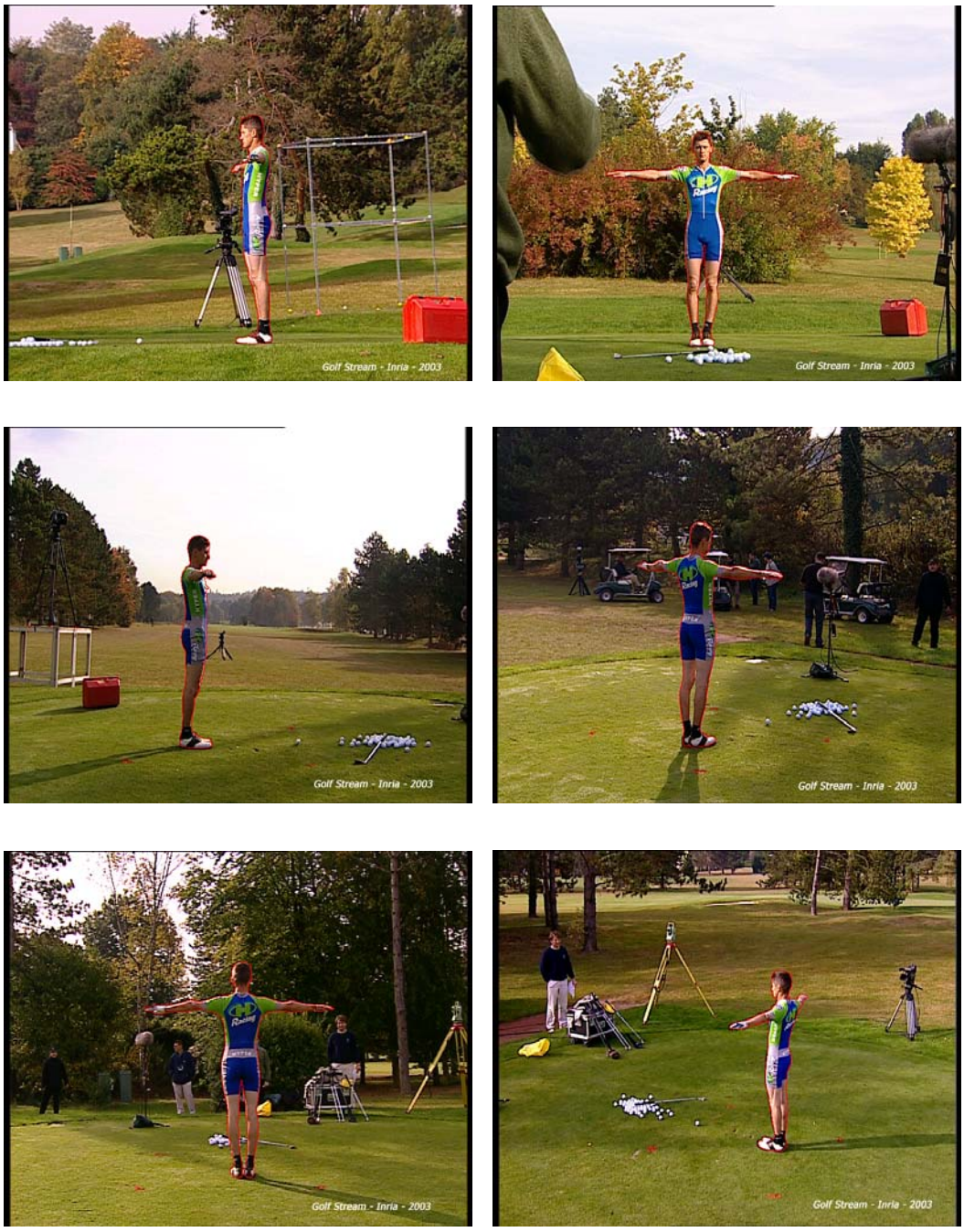


Figure 5.12. Silhouette (in red) of the 'small-man' for different camera views in the real images

Chapter 6.

3D Model Deformation via Silhouette Matching

We have identified that the silhouettes of the subject can be used to reconstruct better 3D human model than with just feature points, which are very sparsely distributed in 3D space. This chapter explains the detail of matching and registration of the silhouettes from the intermediate model to the subject's silhouettes in the real images. The registered information is then used to formulate the RBF deformation vectors for transforming the skin and skeleton of the intermediate 3D model to fit the specific subject. In Section 6.1, we give an overview of our model refinement framework using silhouettes, and in Section 6.2 some general curve matching techniques are reviewed. Then Section 6.3 explains the curve matching method that we developed to register the silhouettes. Next we formulate the deformation vectors in Section 6.4 to improve the intermediate model before concluding our results in Section 6.5.

6.1. Overview

Figure 6.1 shows the overview of the process to improve the 3D intermediate puppet model by matching the silhouette curves. From Chapter 5, we have extracted the multiple-view silhouettes from both the intermediate 3D model and the real images. These two sets of information will be matched and registered.

As shown in the bottom-left picture of Figure 6.1, the two curves must be registered in the correct topological sequence. If the ordered sequence is not correctly followed, self-intersection will occur later on while refining the 3D mesh of the model.

The 2D correspondent deviations between the model and image curve will be used to calculate the 3D deformation vectors for improvement on the intermediate 3D model. Our algorithm will automatically select a subset of the point correspondents along the matching silhouettes so that we can avoid the large amount of point correspondents if we use all of them.

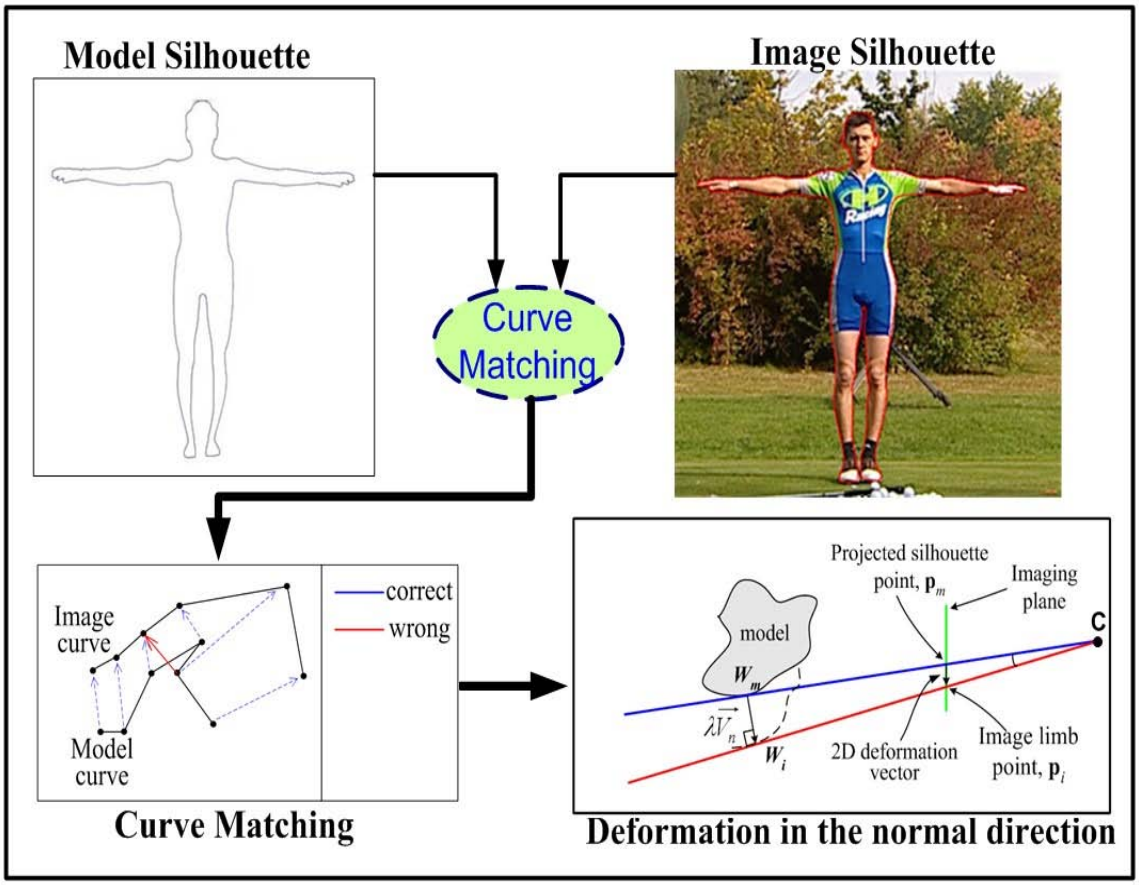


Figure 6.1. Overview of process for improving the 3D model via image silhouettes

6.2. Curve Matching

From Chapter 5, we have extracted the 2 sets of curve information that we will register:

1) Sequence of 2D Points that made up the silhouettes of the intermediate 3D models seen from varied camera viewpoints. For each of the 2D points from the silhouettes, we will also have the 3D position and index identity of its originating vertex on the 3D model. We will call this set of points the model silhouette points.

2) Sequentially ordered lists of 2D Points that made up the silhouettes of the subject in the real 2D images. These points are sampled at least 2 times higher than the model silhouettes to prevent aliasing. We will call this set of points the image silhouette points.

6.2.1. About Curve Matching

Many algorithms had been proposed for matching and registering curve and silhouette. Popular methods are the active contour [71], dynamic programming [133], Fourier descriptors [149], curvature scale space [94]. Contour registration method such as the popular iterative closest point (ICP) [160] is not suitable since matching through the use of rigid transformation and closest correspondent will not always maintain the correct topological order along the curves. Extensive review of curve matching is outside the scope of this dissertation.

At this point of time, the global geometries of the curves had already been aligned through the camera calibration and 3D reconstruction process (in Chapter 4). Our objective is to seek a suitable correspondence of the image silhouette for each model silhouette point.

There are a few important points that we have to take note while performing curve matching:

Topological Sequence among Curves

The 2D points from the model and image silhouettes have to be registered along their arc length so as to enable the correct topological order. Wrong topological sequence will cause severe self-intersection in the 3D model during the final deformation computation.

Inferring Correct Body Parts

Minimizing the global energy or maximizing the similarities between two curves does not guarantee that the inferred relation to the correct part of the 3D object is preserved. This is especially so when the object is highly concave such as the human body. In other words, a projected point on the imaging plane may be coming from different body parts of a human model. A highly concave object may result in many intersections of the projected silhouette edges. Recall Figure 5.5, each point of the intersection is inferred to 2 body parts e.g. left and right foot. We also have to bear in mind that the local surface geometry of the 3D model is not the same as the real subject seen in the images, and these local geometries are in fact the entities that we are reconstructing. Figure 6.2 shows an illustration whereby the wrong body parts from the silhouettes are registered, which will result in reconstruction error in the later stage. It is clear that we have 3D geometrical information for each point from the model silhouette, however, these information is not available for the image silhouette points.

6.2.2. Curve matching from local body segments

Each closed-contour silhouette of the 3D model can be partitioned in several 3D curve segments with respect to the different human body parts. Curve matching could be carried out for each individual curve segment. Next, the image silhouette is partitioned into curve segments which are then registered to the respective model curve segments. Our algorithm partitions the silhouette automatically, although some interactive corrections are permissible in case of major problems.

Recall that in Figure 3.4 of Chapter 3, the human may be labelled into different body parts. For every model silhouette point, we know which part of the body they belong to. Hence we can easily break down the model silhouette into segments of curves by traversing along the sequence of silhouette points. Consecutive silhouette points that belong to a same body part will be parsed as a single curve segment.

The starting and ending points for each model curve segment have to be registered with the appropriate image curve segment. Figure 6.3 shows a sample of the corresponding curve segments between the model and image silhouettes.

The starting point and ending point of each model curve segment can be obtained by scanning along the model silhouette points. An ending point of a model curve segment is obtained when there is a change in the labelling to the body parts at the next silhouette point, which becomes the starting point of the next segmented curve.

The topological sequences that go through the model curve segments will be maintained while finding the correspondence in the image silhouette, which at the same time partitioned the image curve into local segments. By assuming that the model and

image silhouettes are relatively close, for either the starting or end point of a model segment, we will search for the nearest point in the image silhouette that satisfies two conditions:

1) The differences in the traveling direction at both pair of points are less than a threshold. In our implementation, this value is chosen to be around 35 to 50 degrees. This criterion takes care of big 'shifted' misalignment between the 2 data (see Figure 6.4 for an exaggerated example). We have to stress that rigid transformation (of the orientation and position) that acts on the difference between the model and image silhouette is prohibited at this stage (i.e. only the local deformations are permissible) since they had already globally registered via the calibration-reconstruction process in Chapter 4.

2) The matching point in the image silhouette will be searched in a forward sequence based on the ordering index of the previously found points. This condition makes sure that the correct matching order is preserved.

The image silhouette points do not contain any direct information inferring to the 3D model. Parsing the long silhouette curve into multiple local segments confines the curve matching and 3D reconstruction in the latter stages to be more local to their own respective parts.

6.2.3. Subdivision curve matching

Curve matching will be executed on each individual parse segment of model and image curves. This is done by recursively subdividing the model curve into 2 halves (upper and lower curves, see Figure 6.5) at the point of its half arc-length, and then search for its

closest point on the image curve, which will in turn be subdivided at the same time. The recursive sub-division will proceed until no further points from the model silhouette are left to be matched. Through the subdivision matching, the topological sequence along the model and image curves are always maintained.

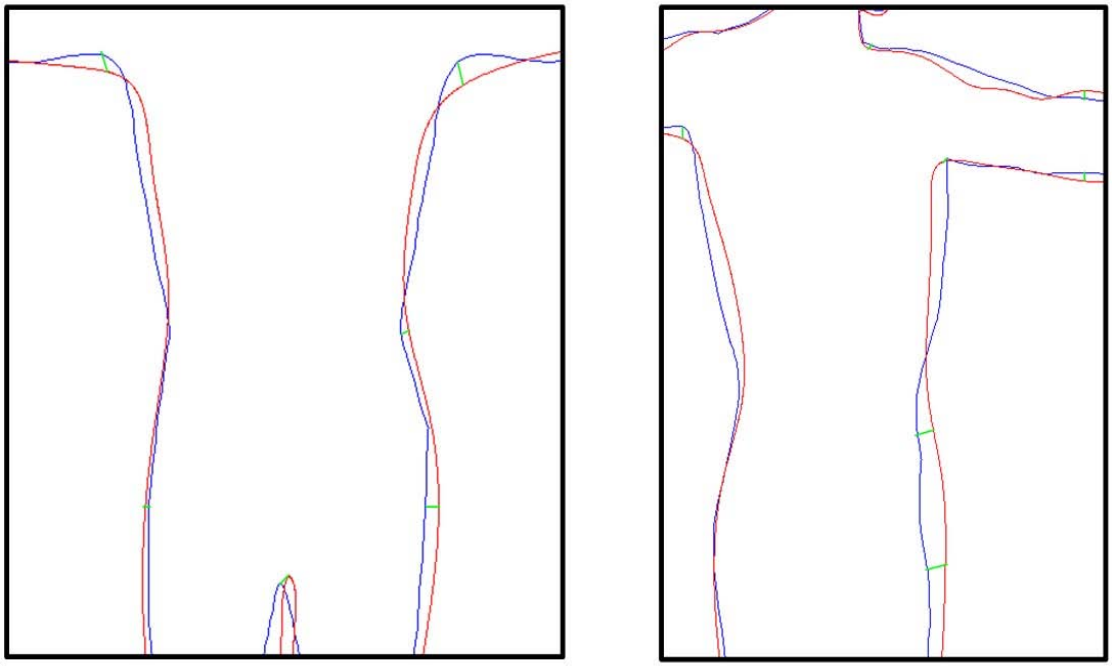


Figure 6.2. Typical scenario of registration of the parse segments between the model and image silhouette – model silhouette (red), image silhouette (blue) & points indicating the parsing (green)

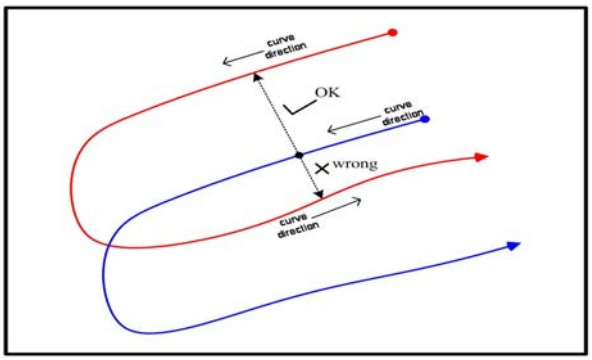


Figure 6.3. Exaggerated example of searching for nearest point constrained by the direction of curves (red – image curve, blue – model curve)

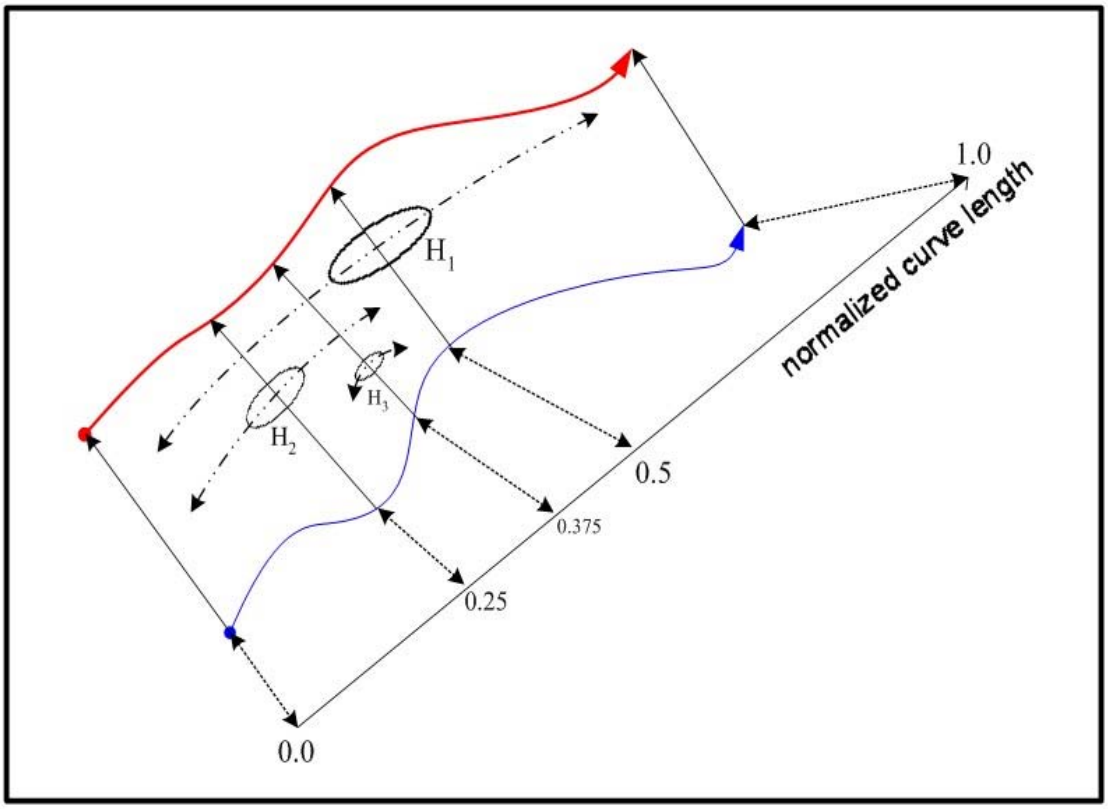


Figure 6.4. Curve matching via subdivision. The model curve in blue and the image curve in red. H_1, H_2 and H_3 indicating the hierarchy within each segment

6.2.4. Curve matching Results

Figure 6.6 shows the typical results of the closed-up view for the matching curves (the model curve is in 'blue' and the image curve is in 'red'). Since the curve matching method is a one-pass algorithm, the energy between the two matching curves may not be minimized. Nevertheless, the matching points are sufficient to build the desired 3D human model.

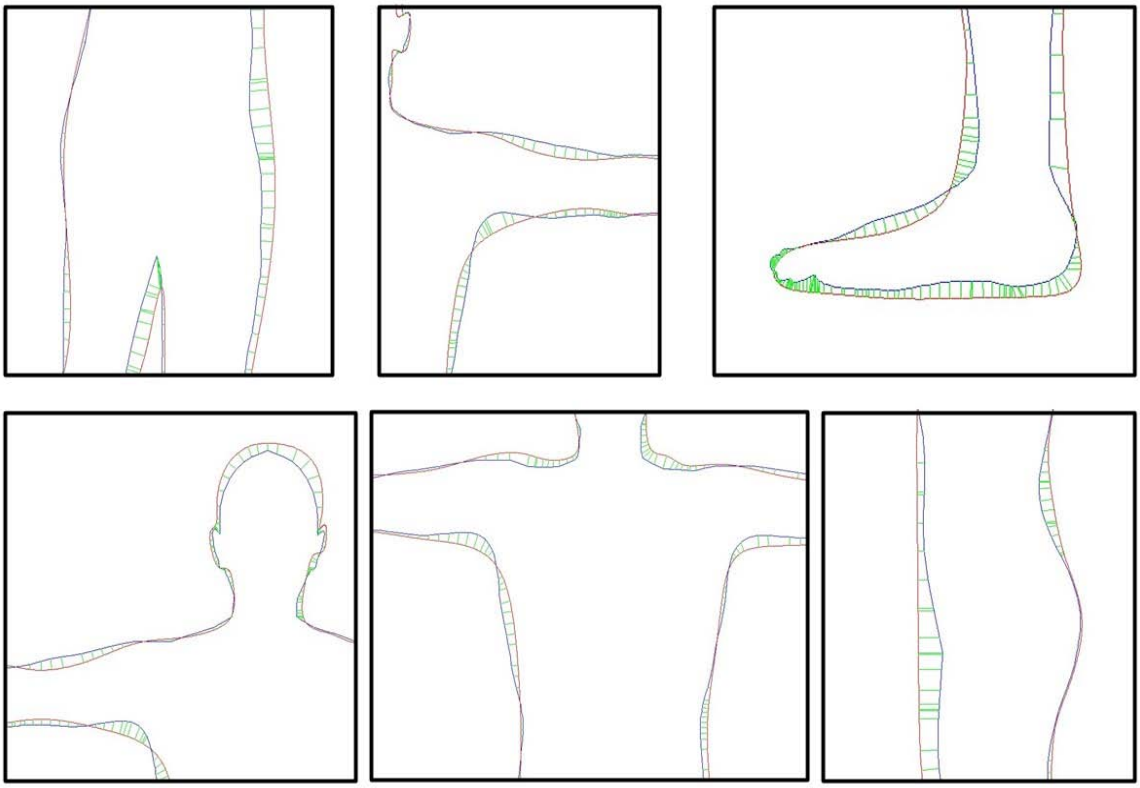


Figure 6.5. Typical close-up results from matching curves ('red' – image curve, and 'blue' – model curve)

6.3. 3D Reconstruction from Registered Silhouettes

When the model and image silhouettes are registered, all the points from the model silhouettes shall find a correspondent in the image silhouette (since the image silhouettes are sampled at a much higher frequency). These sets of correspondents could be used to obtain the deformation variables to yield the final 3D model.

6.3.1. Formulating the 3D Deformation Vectors

Each model silhouette point that finds a matching correspondent on the image silhouette determines a 2D deformation. This 2D deformation is simply the projection of the 3D deformation allowing the reconstruction of the 3D human model. The picture at the

bottom-right of Figure 6.1 illustrates the geometry of each deformation vector that we want to obtain. We seek for the point that is normal to the projected ray of image silhouette while going towards the model silhouette:

1) Given the points $\mathbf{p}_m$ and $\mathbf{p}_i$ on a calibrated imaging plane, the cosine angle α between the projected rays of the corresponding model and image silhouette points can be calculated via taking the dot product of: $\mathbf{p}_m = [u_m, v_m, f]$ and $\mathbf{p}_i = [u_i, v_i, f]$ where the u and v are the coordinates on the imaging plane, and f is the focal length.

2) Next the cosine angle α, and the length d_m between camera centre and model silhouette ($\overrightarrow{\mathbf{CW}_m}$) is used to calculate length d_i between the camera centre and image silhouette ($\overrightarrow{\mathbf{CW}_i}$).

3) Finally, the new 3D point $\mathbf{W}_i$ is computed by scaling $\hat{\mathbf{p}}_i$ which is the unit vector of $\mathbf{p}_i$ by its length d_i after it is transformed back to the same coordinate system of the 3D model. The vector from $\mathbf{W}_m$ to $\mathbf{W}_i$ will be the deformation vector.

From the set of curve matching correspondents, we can have a set of deformation vectors which can add to the radial-basis function as in equations 4.15 to 4.20 (Chapter 4). This RBF will be used to interpolate and refine the intermediate 3D model to yield the final 3D human model.

Some vertices may be visible as silhouette points in more than one camera (frontier point [28]). To deal with these vertices, we simply average the deformation vectors that are associated to the same vertex.

6.3.2. Selecting the Deformation Vectors

The features points from the intermediate 3D model and deformation vectors obtained from the curve matching will be used to build the final 3D model. It can be made more efficient, and still be accurate if we can select only the necessary deformation vectors.

Use of Feature and Silhouette Points

The feature points that were used to construct the intermediate model will be utilized. Since these points define the global geometry of the 3D model, they will not be moved. Therefore in the RBF matrix **A**, their deformation centres will be fixed with zero-magnitude deformation vectors.

On the other hand, if the entire correspondents from the registered silhouettes for RBF interpolation are used, we will have a very large matrix, which we have to invert. From the typical 1000 to 2000 correspondents for each camera view, we will end up with a matrix size of more than 10000×10000. Instead of using all the silhouette corresponds, we just need to use a subset of those.

The 3D reconstruction via silhouette deformation begins by selecting the point correspondent which gives the maximum error. Its 3D deformation vector and its center are appended to the set of deformation variables **D** that already comprises those feature points. It is then used to form the RBF matrix $\mathbf{A}_1$. This allows the computation of the weights necessary for the deformation of the silhouette points of the intermediate 3D model to new locations (call this set of newly deformed silhouette points **S**). We iterate the following steps:

1. Compute the mean and maximum errors in **S** towards the image silhouette points.

2. If the mean error is below a threshold or ceases to change with respect to the previous iteration, the algorithm will stop and the result will be accepted.

3. Or else, use the model silhouette point in **S** that gives the maximum error to append its deformation vector to the existing set of deformation variables **D** before forming the RBF matrix $\mathbf{A}_n$ (*n* signifies iteration count).

4. The deformation weights via $\mathbf{A}_n$ are used to deform the silhouette points of the intermediate 3D model to their new locations (a new set of deformed silhouette points **S** that overwrites those from the previous iteration).

When the iteration ceases, the final radial basis function will be used to deform the 3D mesh of the intermediate model to produce the final external skin. From our results, we show that only a few hundred silhouette correspondents are sufficient to yield the final model.

There may be rare situations when the intermediate model's 3D silhouette edges may not be the same one as of the *true* model's 3D silhouette edge in the real images, or they could even come from different body parts (Figure 6.6). A reliable region for deformation is used to reduce the occurrence of deformation from wrong body parts. Also we can ensure that during the interpolation the vertices that are not deformation centre to be bounded within the image silhouette, which can be done by scaling their respective magnitude in the deformation direction.

Reliable Region for Deformation

Recall that the parsing and matching of curves do not always mean that they infer the correct body parts. This can lead to reconstruction errors. In addition, intersection of

projected edges may occur in concave objects. Nevertheless, we can minimize the occurrence that may arise from curve matching by not using silhouette point correspondents outside the 'reliable' regions. Regions that are considered as unreliable are:

1. The intersection points of the projected edges (which signal a change in body part) are unreliable. Points that are within a few pixels of these intersection points are also considered as unreliable. Since the local geometries of the model are not properly fitted, therefore points within and around the projected intersections are not reliable. In other words, model silhouette points that change in body parts and their near neighborhood are also considered unreliable.

2. Points from the parsed segments that are too short will be considered as unreliable. Consider the situation in Figure 6.2, the visible curve of the arm seen from the side-view is relatively short as compared to its front-view. Therefore, those short segments, which are results of severe foreshortening, should be pruned off, or else matching points from the wrong body parts will cause error as in Figure 6.6.

Since reconstructing the final model needs only a small subset of points from the silhouette matches, omitting unreliable points shall make the model reconstruction to be more reliable and accurate.

Pruning of 2D segments that could be coming from long 3D segments is not a problem, since in practice, we will assume to have enough camera coverage to ensure that all body parts are properly seen. In other words, a long body part may be heavily foreshortened in one view, so we do not use information from this particular view. Then the same body part, given good coverage, should be properly seen in some other cameras, hence information

from these other views will be used. As the deformation technique uses a global model, missing a few edges will not produce geometric disasters.

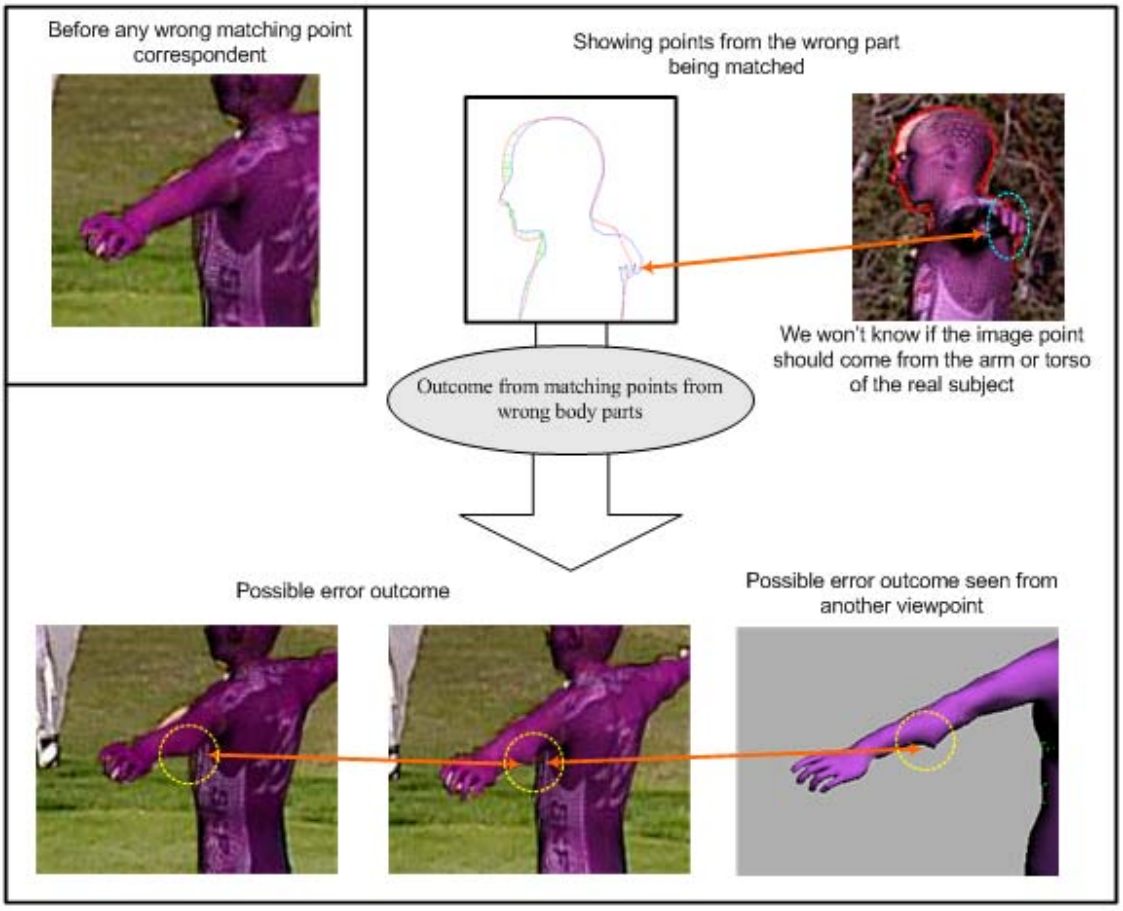


Figure 6.6. Errors in the deformation will occur if matching points are selected from the wrong body parts (image curve – 'red', model curve – 'blue')

6.3.3. Estimation of Skeleton

At the start, we have a generic model that comprises the skin and its skeleton. The skin of the generic model has now been deformed via the radial basis function formulated from 3D feature points to produce an intermediate model of the skin. This RBF will be used to act on the generic skeleton to yield the skeleton of the intermediate model.

From the registration of model and image silhouette, we have formulated the deformation to obtain the final model of the surface skin from the intermediate one. The same deformation will be applied to the intermediate skeleton to produce the final model skeleton. Since the skeleton is always bounded by the external surface skin, and all of the deformation centres (i.e. feature points and silhouette points) come from the external skin, the structure of the skeleton remains the same.

6.4. Results & Discussion

In our setup, silhouettes from the 6 calibrated camera views were used to improve the 3D intermediate human models (from Chapter 5). Our results are illustrated using the same subjects i.e. the 'big-man' and 'small-man'. The customized 3D human model will comprise the external skin and its internal 3D bone-skeleton. Figures 6.8 and 6.9 show the visual results by back-projecting and superimposing the resultant 3D models onto their respective imaging views. The estimated skeletons of subjects were also displayed. Figures 6.10 and 6.11 show some close-up of the customized 3D models. We can notice that the local contours for the two subjects of different shapes and sizes yield much better results than 3D reconstruction that uses only feature points. Figures 6.12 and 6.13 show more close-ups of the superimposed visual results.

Figures 6.14 and 6.15 show the close-up of the resulting models with their skins and skeletons. We can notice the smoothness in the skin and skeleton of both the subjects seen from the virtual views. This gives a better visual and geometrical effect when we make comparison with the popular shape-from-silhouette method that was reviewed in Chapter 2.

See figures 6.16 and 6.17 for more examples of the subjects viewed from other virtual camera poses.

Figure 6.18 is the plot of mean re-projection errors of the silhouette matching versus the number of silhouette correspondent points being used. It validates that we do not need to use all the registered correspondents between the silhouettes for deformation to obtain the final 3D model. In general, from the plot, about 350 corresponding silhouette points were needed i.e. 350 iterations. This means that the radial basis function for the deformation requires at most an invertible 350×350 square matrix (for each iteration), which is manageable using most processors. It took about 150 seconds to execute 350 iterations on a Pentium IV (without code and hardware optimization). In Figure 6.19, we display the visual results for using various numbers of silhouette correspondent points by remapping the 3D model onto their respective views. Since the locations on the intermediate 3D model with larger errors being deformed first, the visual errors are not noticeable after about 100 iterations.

Figure 6.7. 3D Model of the 'big-man' superimposed onto the images from 6 different views (model refinement using silhouette)

Figure 6.8. 3D Model of the 'small-man' superimposed onto the images from 6 different views (model refinement using silhouette)

Figure 6.9. Results of close-up of the ‘big-man’, subject superimposed onto its colour image

Figure 6.10. Results of close-up of the ‘small-man’, subject superimposed onto its colour image

Figure 6.11. More close-ups of subject ‘big-man’, superimposed onto the images from different views

Figure 6.12. More close-ups of subject ‘small-man’, superimposed onto the images from different views

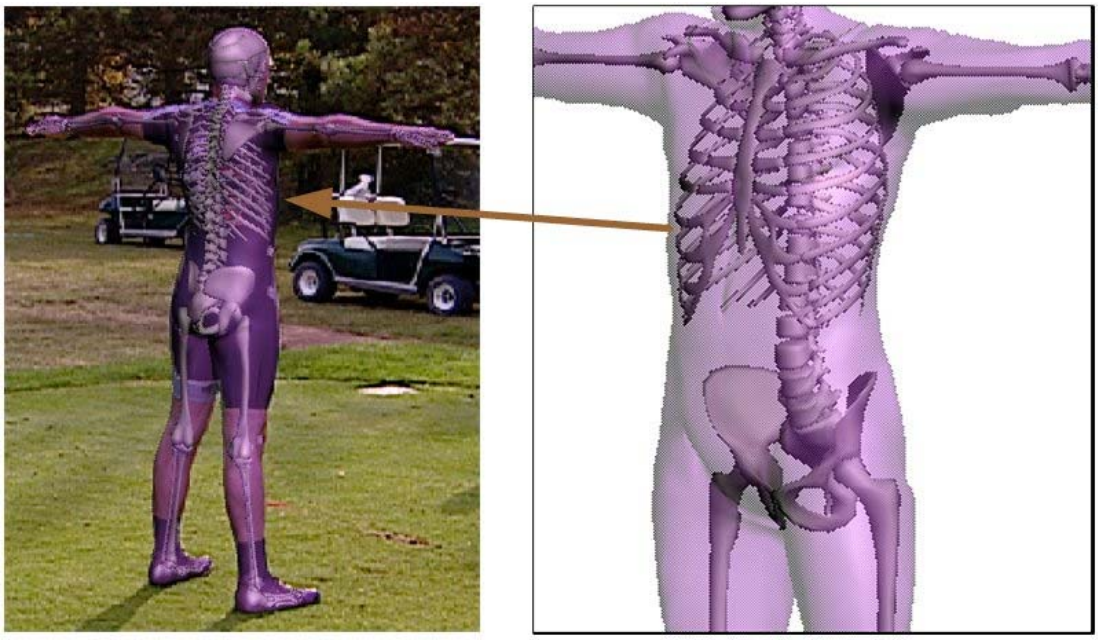


Figure 6.13. Close-up of skin with its skeleton for ‘big-man’ subject showing the smoothness

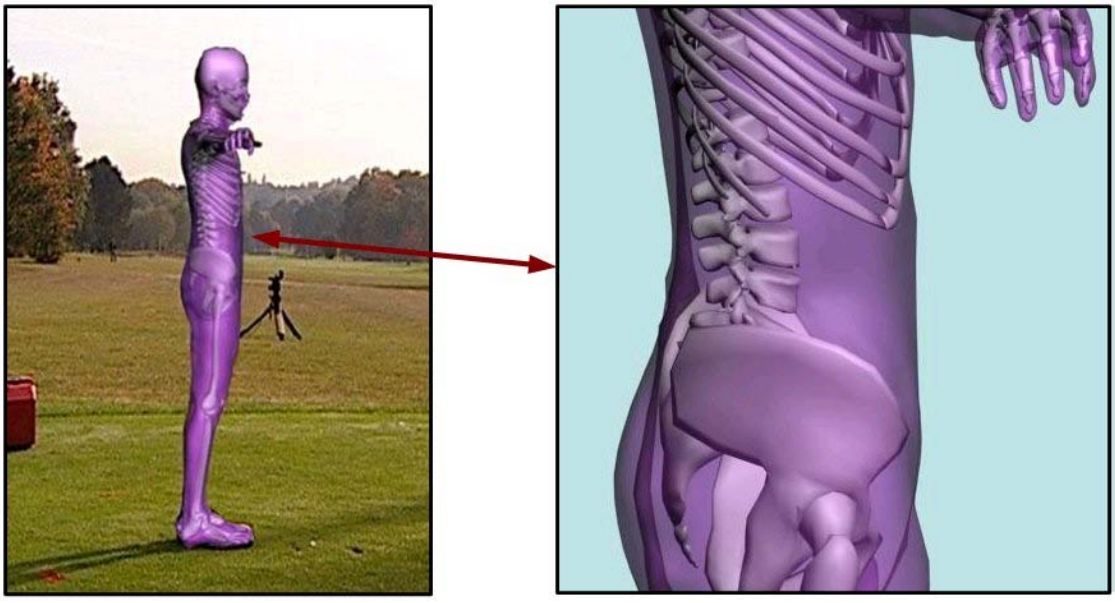


Figure 6.14. Close-up of skin with its skeleton for ‘small-man’ subject showing the smoothness

Figure 6.15. More closed-up example of 3D model of 'big-man' seen from various synthesized virtual views

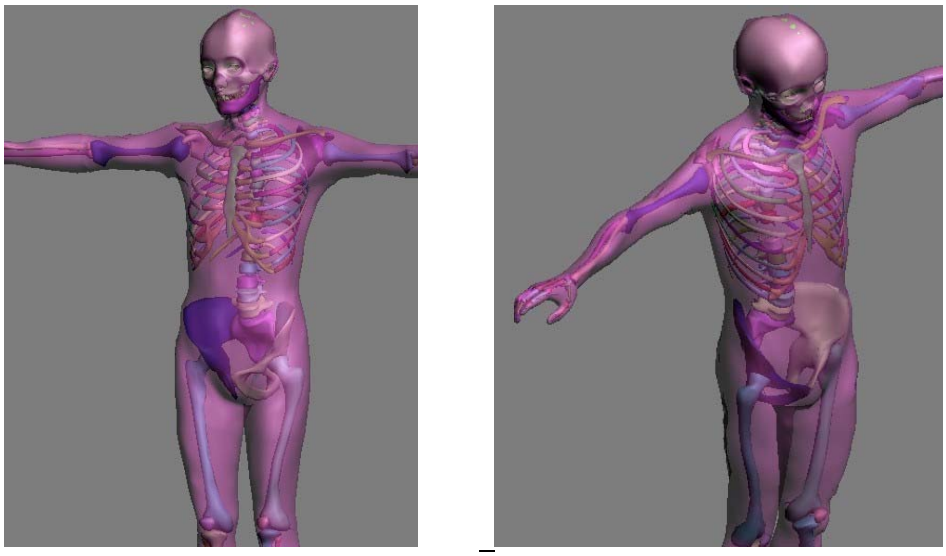


Figure 6.16. More example of 3D model seen from various synthesized virtual views

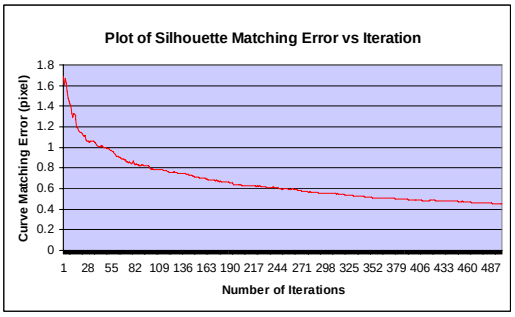


Figure 6.17. Plot of re-projection error of silhouette vs. number of correspondent silhouette used

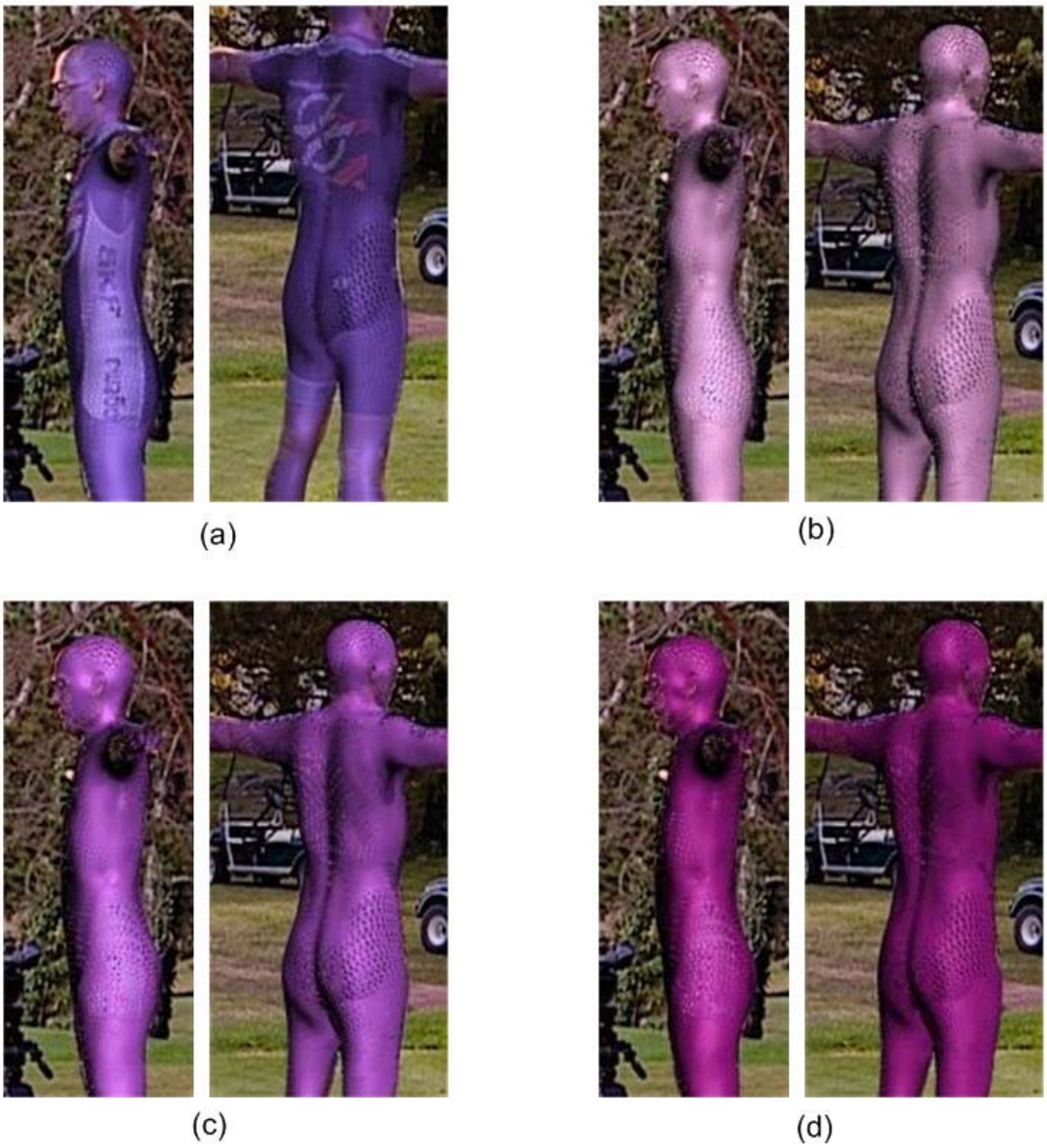


Figure 6.18. Example visual closed-up results of 3D deformation (in different views) using different number of deformation vectors: (a) 25, (b) 80, (c) 160, (d) 300

Chapter 7.

3D Motion Tracking

This chapter will explain in detail the algorithm for capturing the human motion without the need in using marker. Our method, unlike most existing approaches, does not rely heavily on image segmentation. We synthesize and match for the solution by animating the 3D human model to generate visual appearances that are similar to the real scene. Our main focus is on studying the human arms, which have one additional degree-of-freedoms and quantitatively small. The search for the moving human posture was performed by using the simplex simulated annealing, and the visual synthesis of the human posture implemented in the GPU. We show that our algorithm is able to operate in cluttered and moving background, and also cope with partial self-occlusion.

We begin by giving an overview of our algorithm in Section 7.1. Section 7.2 explains the representation of the human posture that we have to search using numerical optimization (Section 7.3). Section 7.4 explains the rendering of the synthesized images for analysis in Section 7.5. In Section 7.6, we deal with possible occlusions. The implementation issues are described in Section 7.7 before we present our results in Section 7.8.

7.1. Overview

Figure 7.1 shows the framework of our human motion capturing algorithm. Before this process begins, the followings are available:

1) The 3D puppet mesh that closely resembles the subject.

2) The positions of the 3D joint nodes of the customized puppet linked by its kinematics chain.

3) The 3D puppet in the initialized posture is pre-positioned and registered onto the first frame images for the mapping of texture.

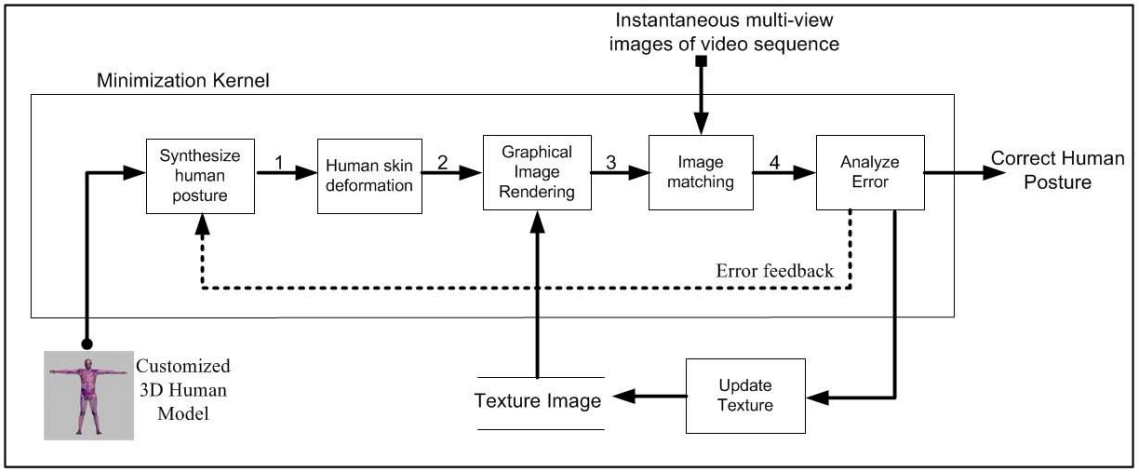


Figure 7.1. Framework of our human motion capturing algorithm

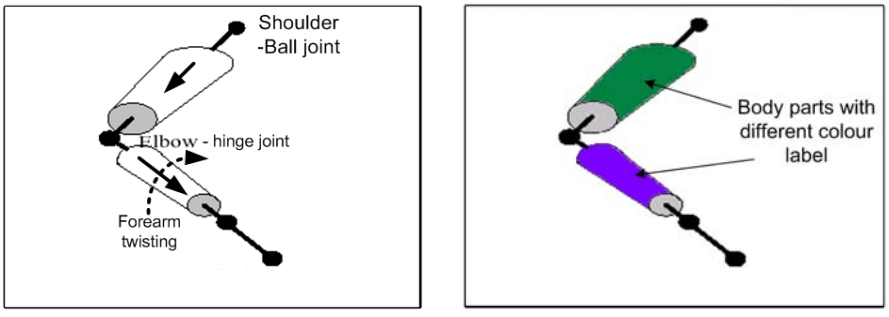
At each time step, the simulated annealing minimization is used as the kernel for the analysis-by-synthesis process to search for the correct human posture. Texture images updated from time to time are used for 'coloring' onto the 3D puppet. The error between the synthesized images representing the 3D human posture and input images is computed by comparing the color texture of the puppet images with the real human images pixel-by-pixel. The minimization criterion is a function of the degrees of freedom of the puppet's kinematics articulations. This error is fedback to the synthesizing process in the minimization loop that iterates until the error reaches a minimum. Then we proceed to the next time instance, and this procedure repeats.

7.2. Human Posture - Kinematics Chain & Skinning

Use of Forward Kinematics

The movement of the human posture may be described by a forward kinematics (FK) chain driving the external skin made up by its 3D mesh. This kinematics chain links the parent joint node to its child nodes and replicates to the following ones. For this forward kinematics chain, in general, only the node at the pelvis joint (the root) is allowed to undergo 3D translational and rotational transforms. For the rest of the nodes, only rotational transform is allowed. For example, the location of the elbow joint is a function of the pelvis's translational and rotational displacement, and replicates through only following rotational components until it reaches the elbow.

Figure 7.2a shows the forward kinematics chain that links the joint node of a human arm, its hierarchical direction and an example of the external skin approximated by 3D cylinders. The shoulder joint is the parent of the elbow joint and so on. Each articulated node on the human body part can be modeled by the Euler rotational angles; a ball joint will has 3 degrees of freedom (DOF) while a hinge joint has 1 DOF. In human anatomical terms, the shoulder is a ball joint, whereas the elbow is a hinge.



(a) (b)

Figure 7.2. Kinematics chain and labeling of the human arm parts

Figure 7.2b presents the upper-arm and forearm, which are our main body parts, labelled with different colors for identification during the hierarchical posture estimation process.

Inverse Kinematics

Inverse kinematics (IK) is the reverse of the forward kinematics. IK formulations are commonly used to solve joint kinematics when given the end-effectors e.g. given the location of the wrist node, we attempt to deduce backwards the parameters of the subsequent parent joints and their respective kinematics. However, replicating the kinematics chain backwards via the end-effectors will result in non-unique solutions. For example, given 2 end-effectors for a system of more than 2 DOFs may result in singularities in the equation that we want to solve; hence IK will not be used in our case for posture synthesis.

Posture Synthesis & Skinning

The human body undergoing kinematics transform causes the external skin in the respective areas to deform accordingly. As mentioned in Chapter 2, non-rigid skin deformation is an extremely difficult problem. For example, inside the forearm are the *ulna* and the *radius* bones. It is in fact the *radius* bone rolling across the *ulna* bone which brings the flip of the palm, which also causes the non-rigid deformation of the forearm surface. This forearm movement may be approximated by an additional twisting function (Figure 7.2a).

In this dissertation, only rigid skinning is considered and problematic surface near to the joints e.g. shoulder and elbow will be ignored. This assumption is valid for the arm movement that we want to study as the surface regions near these joints are quite small.

7.3. Numerical Minimization

The algorithmic framework on Figure 7.1 can be described and realized as a numerical minimization function (also known as optimization). We can refer to this as the minimization kernel for finding the optimal 3D human posture of the subject that appears in images at each time-step. The output of the image matching module and the error feedback to the posture synthesis is the error $E(\mathbf{u})$ between the synthesis images and real scene image. In other words, optimization is used to minimize the error as a function of the degrees of freedom of the puppet's kinematics articulation. Given $\mathbf{u}$ as the N-dimensional vector of the continuous variables representing the DOF of kinematics parameters of the human posture that we seek for, this error is the criterion function that we want to minimize, and generally it is written as:

$$\hat{\mathbf{u}} = \arg\{\min_{\mathbf{u}} E(\mathbf{u})\} \tag{7.1}$$

The common minimization concepts that have been used are the gradient-based methods, the simulated annealing and the particle filter approaches. The choice of which approaches to take basically depends on:

1. The convexity of error function.
2. Rate of convergence (i.e. total number of iterations required)
3. The search for a local or global minimum.
4. The dimensionality of the unknown vector to be searched for.

Doing exhaustive search for the solution of a high-dimensional vector of continuous variables are unlikely even if we have supercomputing ability. Also, the minimization of

this function can be difficult due to large dimensionality and non-linearity of the vector **u**, and the presence of local minima. To ensure that the minimum corresponding to the solution **u** is accurate, we must be able to mimic and synthesize images similar to the real environments. As the human subject moves in cluttered background, covering and uncovering background, and occlusions, the resulting error function can be highly concave and contains many visual discontinuities. In the following, we will briefly summarize the common minimization methods and their suitability for our problems. More comprehensive detail is available in references e.g. [116]. Also there are many other minimization approaches such as those that combine gradient-based method with stochastic search, and the combinatorial optimization. A thorough discussion on minimization methods is beyond the scope of the dissertation.

7.3.1. Gradient-based Minimization

Many non-minimization problems in the domain of computer vision and computer graphics had been tackled by using gradient-based approaches. This is an iterative approach, and is popular because of its fast rate of convergence. Some commonly used methods that come under this approach are the steepest descent, conjugate gradient, Quasi-Newton and Levenberg-Marquardt computations.

During the iteration of this algorithm, the solution vector is always computed to move in a deterministic direction such that the error of the objective function is decreasing when iterating from one step to the next. The direction of the solution vector is based on the derivatives of the criterion function with respect to the optimizing variables. The convergence is taken at the point when the rate of change between the iterations goes below a predefined threshold.

The gradient-based approaches may converge only to the nearest local minimum. One way to achieve better minimum is to start the computation at different initial values. Multi-resolution data representation technique may be applied to speed up the computation. However, the human motion kinematics is of high dimensionality and complex. This results in many possibilities and combinations of the initial values directing to their own respective local minima, especially when the visual motions between the images at the two time instances are fairly large.

7.3.2. Simulated Annealing using Simplex Function

Simulated annealing (SA) belongs to a class of stochastic relaxation algorithms that are essentially prescript for partial random search in the solution space. This algorithm is also iterative, however, unlike the gradient-based iterations that move the solution vector to a decreasing criterion function, the simulated annealing permits changes that increase the criterion function on a random manner. At each iteration, the solution from the previous iteration is subjected to a random perturbation, and the criterion function is re-evaluated. This random perturbation is a function of the 'system temperature', automatically scheduled by the SA algorithm.

A hill climbing move is sometimes necessary to bail the solution from a local minimum. It had been shown in [29] that simulated annealing is a global minimization algorithm for continuous variables, and experiences [48], [130] have indicated its ability to converge correctly. In addition, the SA does not need the criterion function to be smooth or even continuous in their domain. The only drawback is that it is more computationally costly since more iterations are needed due to the hill climbing process.

The simulated annealing algorithm built with the Nelder-Mead simplex [116] has been chosen as our minimization kernel so that reliable and accurate results are obtained. In a criterion function that has thousands of potential minima, it is better to do with a more conservative approach. Moreover, the computing power has got more and more affordable over the years, and it is possible to parallelize the algorithm [122] and apply hardware acceleration. This also serves to ensure that the solution we obtained is good enough as a basis for comparison should we want to move to a faster convergence minimization method such as the gradient-based approach. The main concept of the SA that we used is the temperature controlling of the Metropolis algorithm using Nelder-Mead simplex directional search (please refer to Appendix A for details).

We take the forward kinematics chain to optimize for the posture of the subject hierarchically, starting from the parent joint node and replicate it to its respective child joints. Let us consider the estimation of the human posture's kinematics at a particular node for one image frame to the next. The simplex algorithm may be expressed as:

$$\mathbf{u}_i = \mathbf{u}_0 + \lambda_i * \mathbf{E}_i \tag{7.2}$$

where $\mathbf{u}_0$ is the initial pose, $\mathbf{u}_i$ is the respective vertices in the *N*-dimensional space representing each synthesized posture, the $\mathbf{E}_i$ is the *N*-vector specifying the deformation direction, and the λ_i are initialization values chosen to be large enough to enable the optimizer to skip over local minima.

The simplex procedure will deform the current estimation of the solution with respect to the error function by trapping the potential solution in its convex hull before eventually converges to it. This deterministic simplex computation is coupled with the Metropolis

algorithm controlled by a temperature annealing schedule so that after a convex hull deformation, a stochastic search is performed to overcome possible local minima. The convergence is taken when the rate of change of the criterion function between subsequent iterations is below a certain threshold, or when the number of iterations reaches its maximum.

7.3.3. Particle Filter

The particle filter is also another common optimization approach used for searching. This method assumes that the knowledge of the model state **X** is represented by a posterior density distribution $p(\mathbf{X}|\mathbf{Z}_t)$ after incorporating all the measurements, $\mathbf{Z}_t$, up to the current time-step *t*. This density function is represented by a finite set of *N* counts of normalized weighted particles or samples:

$$\{(\mathbf{s}_t^1, \pi_t^1) \ldots (\mathbf{s}_t^N, \pi_t^N)\} \tag{7.3}$$

An estimation of the state $\hat{\mathbf{X}}_t$ at each time step can be easily estimated by the sample mean of the posterior density, $p(\mathbf{X}|\mathbf{Z}_t)$,

$$\hat{\mathbf{X}}_t = E[\mathbf{X}] = \sum_{n=1}^{N} \pi_t^n \mathbf{s}_t^n \tag{7.4}$$

or the mode

$$\hat{\mathbf{X}}_t = M[\mathbf{X}] = (\mathbf{s}_t^j, \pi_t^j) = \max(\pi_t^n) \tag{7.5}$$

In common practice for motion tracking, the model state **X** could include **u**, the N-dimensional vector of the continuous variables representing the DOF of kinematics

parameters of the human posture and other parameters such as angular and translational velocities. Then the measurement $\mathbf{Z}_t$ could be image related features.

The tracking of a particle filter can be done by:

1. Resampling: This is to overcome the depletion of population, casing them to drift far enough from their weight to become insignificant to the a posterior density function.
2. Stochastic movement and dispersion: This is to deal with uncertainties during the movement of the tracked object.
3. Measurement: This is a likelihood function that evaluates each particle, thus producing new weight for each particle with respect to how well it fits to the image data.

7.4. Graphical Image Rendering

The 3D puppet model of the subject has to be 'colored' with texture, and then graphical rendering is carried out to generate the images for comparison with the real ones (Figure 7.3). We are ready to generate images of other postures when the 3D puppet posture is registered correctly with the multi-view 2D images (called the texture images) via the process of human tracking and posture estimating, or initial pre-position. Since the 3D model puppet is available, we will also be able to synthesize images from arbitrary camera viewpoint if there are sufficient texturing images that cover all the facets of the 3D puppet.

The vertices of the 3D puppet are back projected to the pixel coordinates (sometimes also referred as texture coordinates) in the respective views given that the camera had been calibrated (and fixed for simplicity). For each camera view to be used for motion tracking,

we will compute the pixel coordinates for all the visible vertices in its texture image i.e. the (u, v) space.

When the 3D puppet undergoes motion, the texture coordinates of the vertices of the 3D mesh do not change. Only the shape of the puppet changes in accordance to the joint kinematics and skinning function. We also assume that (1) the surface of the 3D model is matte, and (2) the amount of light incident and reflected from this surface does not vary dramatically between each time-frame. Thus the generated visual appearance of the 2D synthesized textured images will be similar to the real ones when given the correct kinematics, appropriate skinning function and texture images.

The rendering of synthesized image is done simply by warping the texture image via the texture coordinates of the deformed 3D model and calibrated camera projection geometry. However, the texture coordinates and their respective newly synthesized screen output coordinates are not integer values. Rounding-off to the nearest neighbor will cause undesirable aliasing. Therefore, in practical implementation, bilinear interpolation and mip-mapping [154] are applied.

When more than one camera has been used for image synthesis, the same object point seen in images filmed from different viewpoints at same instance may have different illumination values. To avoid illumination inconsistency across the views, the prior images captured with the same camera will be used FIRST to synthesize new images. The texture images for each camera will be updated once every few frames to avoid the drastic changes in illumination and visible part of 3D model.

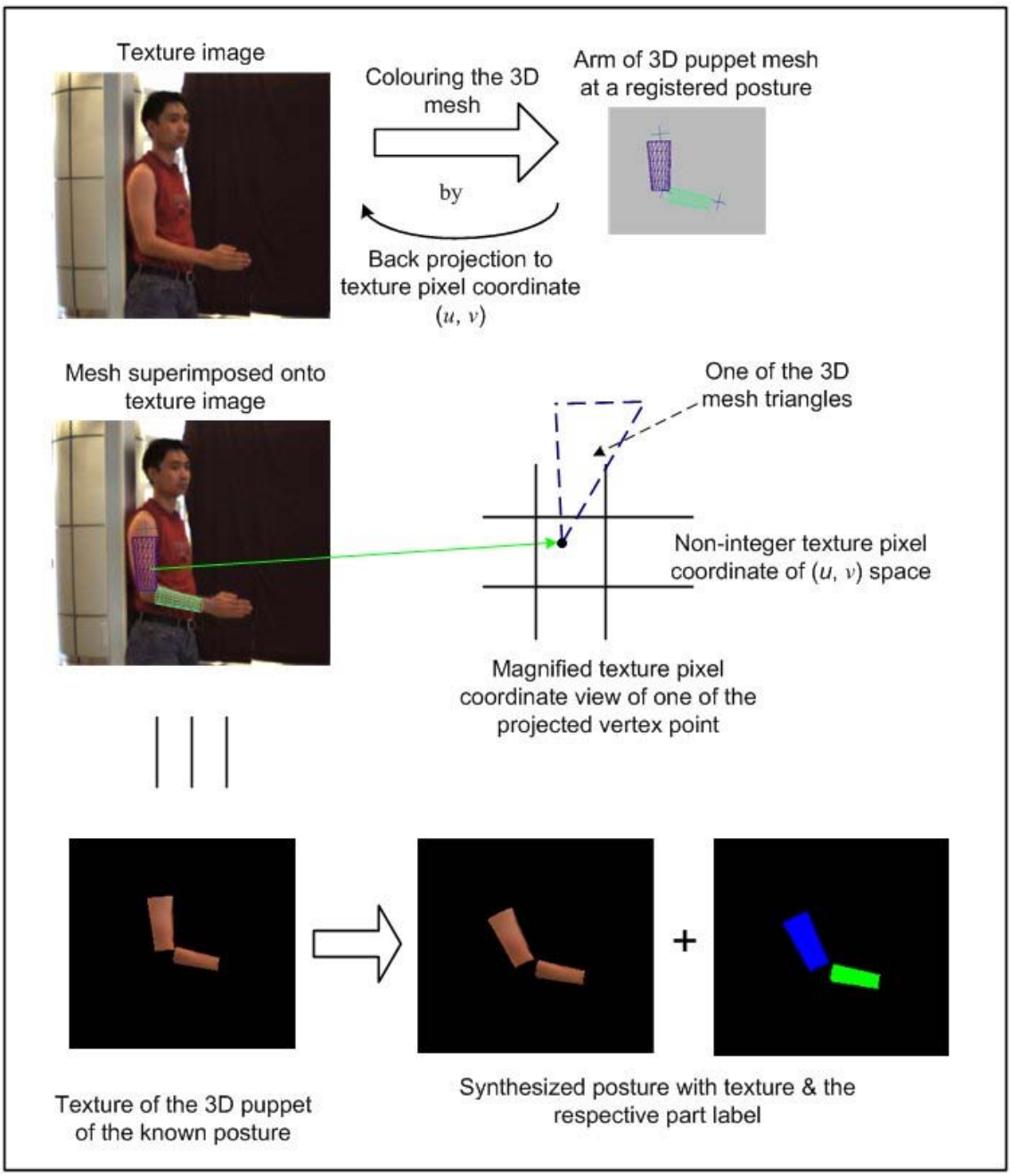


Figure 7.3. Generating the human pose for matching

The rendering of the color labelling image representing the different body parts can be achieved by vertex coloring. The same geometrical information of the scene as in the generation of textured images is available for rendering. The 3D vertices of the puppet are

projected to the image screen while the facets of this model are rasterized to fill it up. All the pixel regions that do not belong to the model puppet will be labelled as 'black' i.e. intensity value '0'. Figure 7.3 presents the right upper-arm labelled with blue color, whereas the forearm is identified with green. This 'part-labelled' image is used for masking out the unnecessary background region during the image matching computation.

7.5. Image Matching and Analysis

This module computes the difference between the graphically rendered images of the synthesized human posture and the real input images. This difference is the criterion function that we like to minimize and is formulated as:

$$E(\mathbf{u}) = \frac{1}{N}\sum_{\forall v}\sum_{r\in L_v} | I_{\text{synt}}(r) - I_{\text{real}}(r) | \tag{7.6}$$

where:

$I_{\text{synt}}(r)$ are R, G, B illumination values of the synthesized image which is a function with respect to the synthesized 3D human kinematics represented by **u** and its texture.

$I_{\text{real}}(r)$ are R, G, B illumination values of the real image.

L_v is the region of the image that is encompassed by the synthesized object. In other words, this is the body part of the subject under investigation. Our motivation is to make sure the error analysis is based on the region of body parts, i.e. to mask out the background that may be cluttered.

v refers to the indices to the set of the multi-view images to be used for evaluation.

Given N_v is the total number of pixels in the regions L_v of each respective image, we have:

$$N = \sum_{\forall v} N_v \tag{7.7}$$

The equation (7.4) is valid for any number of cameras that we desire to use for evaluating the criterion function.

Figure 7.4 describes the process of how equation (7.3) can be realized in practice. This structure allows the possibility of parallelizing the computation easily via simple logical operations. First, the synthesized image is subtracted by the real image for each pixel. Then, the sign bit of the result is removed to obtain the magnitude, which is the error image $E(m, n)$. The error image is masked with the synthesized part-labeling image. For example, if it is the forearm posture that we are estimating, only the synthesized forearm region will be '1' and the rest of the image region will be black '0', which masks out the background region. And finally, to obtain the result, the masked error image is summed up before it is divided by the total number of pixels representing the body region.

7.6. Dealing with Self-Occlusion

Self-occlusion occurs when one part of the body cover another. Synthesized images containing self-occlusion objects are computed using any standard hidden surface removal, such as the z-buffer and a-buffer algorithms. Hidden surface removal is implemented as part of the graphical image rendering.

Although, our tracking algorithm follows the different body parts hierarchically from the parent to the children nodes considering the case for the arms, there is a possibility that for one forearm occludes the other. Figure 7.5 shows an example of the right forearm occluding the left.

It is clear that when matching the synthesized image with the real image, the occluded part should not be used. Both the forearms will fall under the same hierarchical level in the kinematics chain. We shall deal with the least occluded front facing part first so that the foremost information is being processed first. By referring to the part-labeling image, we can calculate how mush an object is occluded by the ratio of:

$$V_j = \frac{\text{the number of non-occluded front-facing pixel}}{\text{total number of front-facing pixel}} \tag{7.8}$$

where *j* refers to the respective body parts.

For example, in Figure 7.5 the right forearm will have a higher ratio than the left forearm; hence it will be processed first.

7.7. Implementation & GPU Acceleration

The motion tracking algorithm is developed by using C++. The graphical image rendering is done by using the Wildmagic 3D graphical engine [39] that utilizes the GPU hardware acceleration driven by OpenGL. The 3D graphical engine provides a better comprehensible structure for software engineering and development of the algorithm. All the equipment that we used comprised cheap off-the-shelf devices. The processing hardware that we used is the Pentium 3.0GHz and the NVidia GeForce 6600 plugged onto the AGP×8 bus. The GPU hardware is used for accelerating the graphical off-screen rendering processes that produces the synthesized images: (1) bilinear interpolation and mip-mapping of texture mapping from 3D mesh, and onto the synthesized images, (2) projection and rasterization of 3D triangles, (3) hidden surface removal during the rendering into 2D graphical images, (4) vertex coloring for rendering of the label of the different body parts.

The simulated annealing and the image matching algorithm are executed on the CPU for ease of debugging and data analysis. This implies that for each iteration of the numerical optimization process, the synthesized data must be transferred back from the GPU to the CPU.

Following is the pseudo code of the implementation:

```
Do for all frames in the video sequences:
    From previous frame to the current frame:
    Do until converge
        Generate u by annealing and simplex search
        Deform 3D puppet's mesh with respect to u
        Render texture image I_synt
        Render body part labeling image I_L
        Read I_synt and I_L back to CPU
        Compute and sort the order of body parts to be tracked
        Compute E(u) i.e. (7.3) using CPU
    End
    Update texture image if necessary
    If texture image is updated
        Compute texture co-ordinate of the 3D puppet mesh
    End
End
```

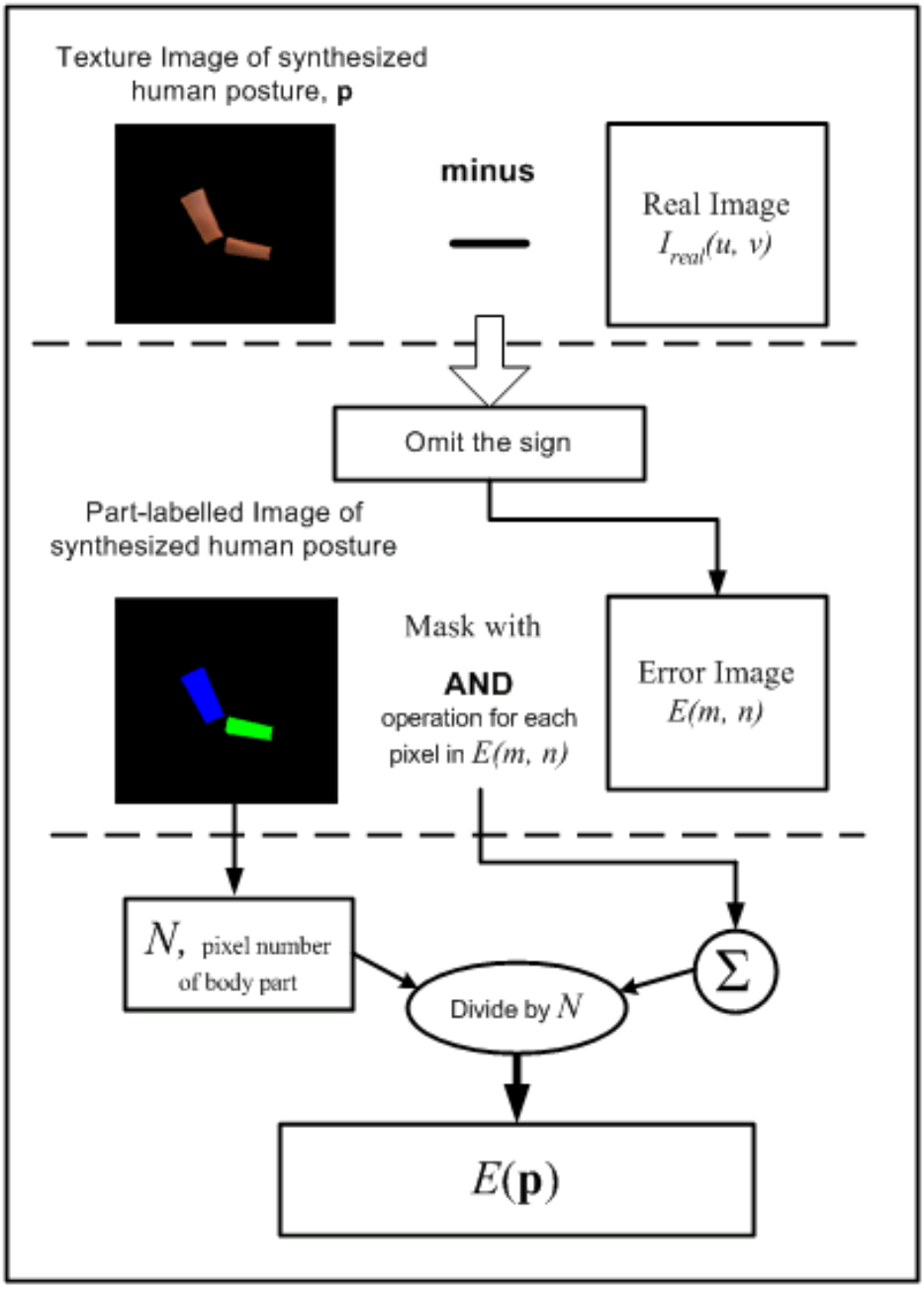


Figure 7.4. One way to realize equation (7.3) and possible algorithm parallelization

Figure 7.5. Example showing right forearm occluding the left.

7.8. Results & Discussions

We have tested our algorithm to track in particular the human arms. To study the movements of the human arms, we need to have prior information of the location of the shoulder joints. One simple way is to 'fix' the preceding links to the shoulder joint in a static location by restraining the subject.

7.8.1. Tracking Results

Figures 7.6 to 7.9 show the respective results of motion tracking in various scenarios: simple arm movements, movements in moving background, tracking in self-occlusion and outdoor tracking in cluttered environment. The test sequences for figures 7.6 to 7.8 are about 10 seconds, and the golf swing in Figure 7.9 is about 1.5 seconds of very fast movement. Our algorithm is able to sustain the tracking for the whole duration by following closely the subject seen in the images. For most the time, a single camera is used. However, tracking reliability is improved when more views are used. We used at most 2 cameras when necessary to prevent the tracking algorithm from divergence. The 3D meshes of the arms are superimposed onto the sequence of images to visualize the tracking

quality. They are overlaid onto 2 different camera views to validate their 3D depth validities. The left images are from camera 1 and the images on the right are from camera 2.

Figure 7.10 shows a quantitative measurement of tracking the elbow bending angles using our method versus the tedious manual in positioning the angles. This is done by using 3DS Max for manual adjustment of arm parts frame-by-frame as the kinematics of the arm movement were manually recorded. From the plot, we can observe that the motion trajectory is closely tracked with error of less than 3 degrees.

In our method, the bones of the model and of the body are matched implicitly and not explicitly. Displaced bones of the model infer a displacement of the textured skin, which is projected on the image. It is the projection of the textured skin that is matched with the textured part of the body in the images. Figure 7.11 presents the plot, at each time step, the error of the red, green and blue intensities when the error function of equation (7.3) is minimized. The mean and variance of the errors over the whole duration for the (1) red component are 3.144 and 0.662, (2) green component are 1.849 and 0.2, and (3) blue are 1.573 and 0.071. On the overall, we can see that the error units are fairly low. The average error in the intensities is about 3 units, and the maximum error is about 6 units. Although, the red band has the higher error than the other, however, the shape of the error functions along the time axis are similar. Regarding the red dominance of the error, we believe that it is due to the specific aspect of human skin colour.

7.8.2. Computation Requirements

For each arm (upper and lower), it took about 800 iterations of numerical minimization to find the correct posture. In total about 1600 iterations are needed to find the posture of both

arms. Overall, we need about 8 to 10 seconds to estimate the arm postures from one image frame to the next. This is because, for each second, we can perform up to 180 iterations of the criterion function computing. We have identified that the bottleneck in the whole process of computing the criterion function is the reading back of data from the GPU to CPU. One possible solution is to implement the whole process of evaluating the criterion function of Figure 7.4 in the GPU.

An alternative to speed up the time to search for the correct posture is to use other kinds of optimization method such as the gradient-based minimization. Although gradient-based methods are very quick to converge, we have to take note that the high-variable error function must be convex along its initial variables. Gradient estimation of this highly complex discrete error function is also a big challenge in itself!

7.8.3. Comparative Discussion

As mentioned in the first two chapters that the human motion tracking algorithms depend very much on the context of their operational needs and environments. Therefore to compare directly the performance in term of accuracy, reliability and computational speed between each individual method is quite subjective and difficult. For example, in [99], they evaluated three different approaches for marker-less motion capture. However, their report also implies the difficulties in direct performance comparison between methods. Nevertheless, we will explain the potential of our proposed framework and realization for bio-mechanical applications with respect to some of the existing approaches.

Comparison with Various Shape-from-Silhouette Methods

Many of the proposed methods based on the shape-from-silhouette approach had been realized in different ways. All these methods were reported for operating in fairly well-controlled and low cluttered environments utilizing several cameras. For example in the methods such as [25], [73], [92], the segmentation of the subject from the background plays a very crucial role to enable the 3D volumetric shape of the subject to be built before the 3D kinematics were estimated. Their subject-background segmentations are usually performed using image background subtraction, with possible spatial cues extracted via edge-based detections. Given that the possible extension of motion capture applications to operate in various illumination environments with higher clutters and moving backgrounds, to rely heavily on subject-background segmentation may be highly problematic. Our approach, in contrast, does not rely strongly on good prior segmentation. Once our 3D human model is correctly textured and animated with the proper shading to produce the synthesized images, we should find the best outcome while matching with the real images. Moreover, our method does not need to use a large number of cameras at one time as compared to the visual hull approach, where a substantial number of cameras are needed.

Comparison with Other Relevant Methods

The extraction of motion and structure through optical flow related method had been suggested [19]. However, optical flow related methods could ultimately lose track due to error accumulation. Other methods such as [55] and [111] rely on estimation of the disparity and depth map, which can be problematic due to clutters. Also, the method by [34] relies on edge and silhouette segmentation be fitted to a series of articulated cones could also faced with the same segmentation problem as the visual hull methods.

The methods such as [126] and [76] that use generative and/or discriminative models require learning or training via human motion database collected a prior. However, we do not assume that this motion database is available, because it is in fact the database of motion which we want to acquire through our marker-less motion capture system. Moreover, as indicated in [126], when the general human motion dynamics are to be learned, the amount of training data, model complexity and computational resources that are required can be impractical.

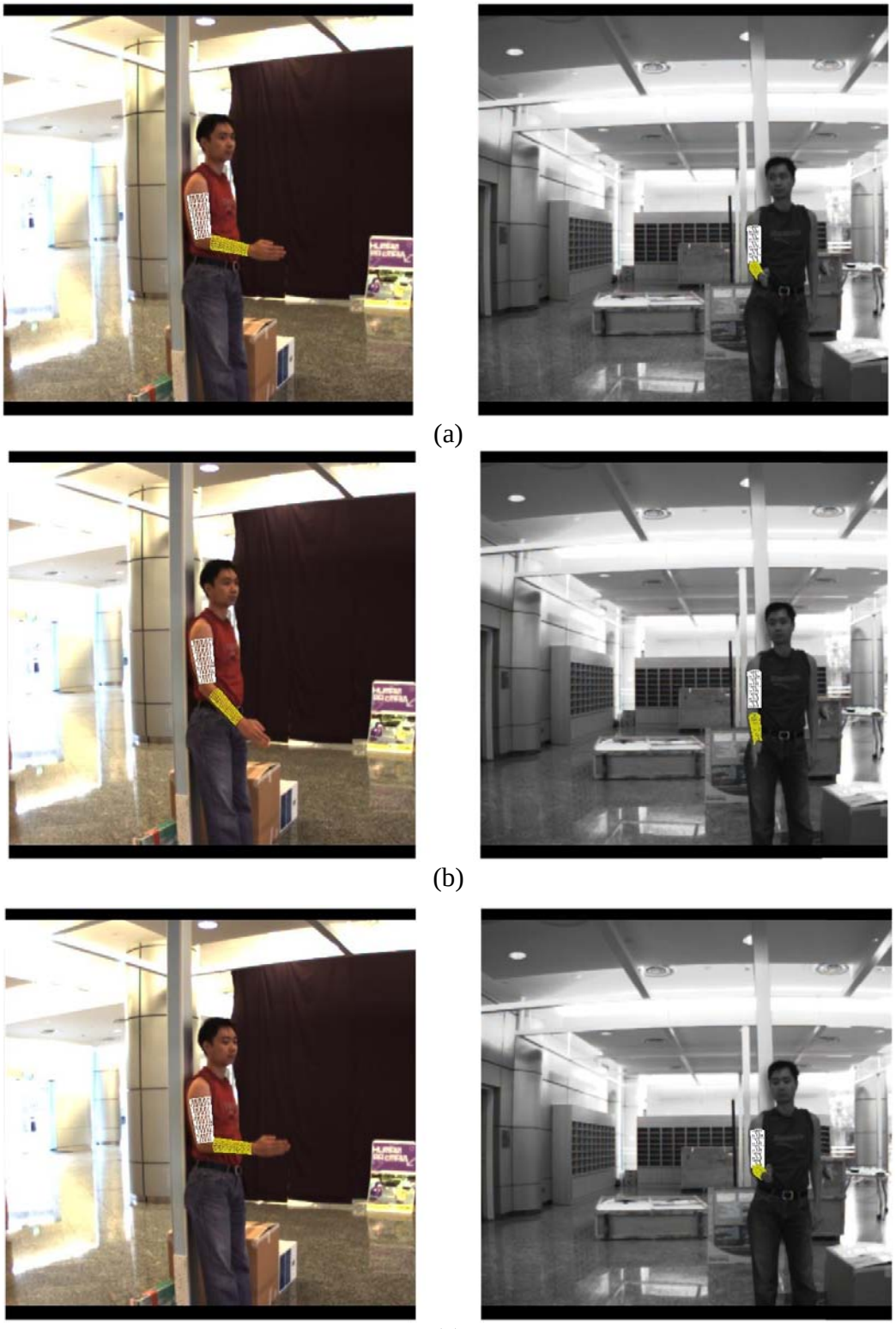


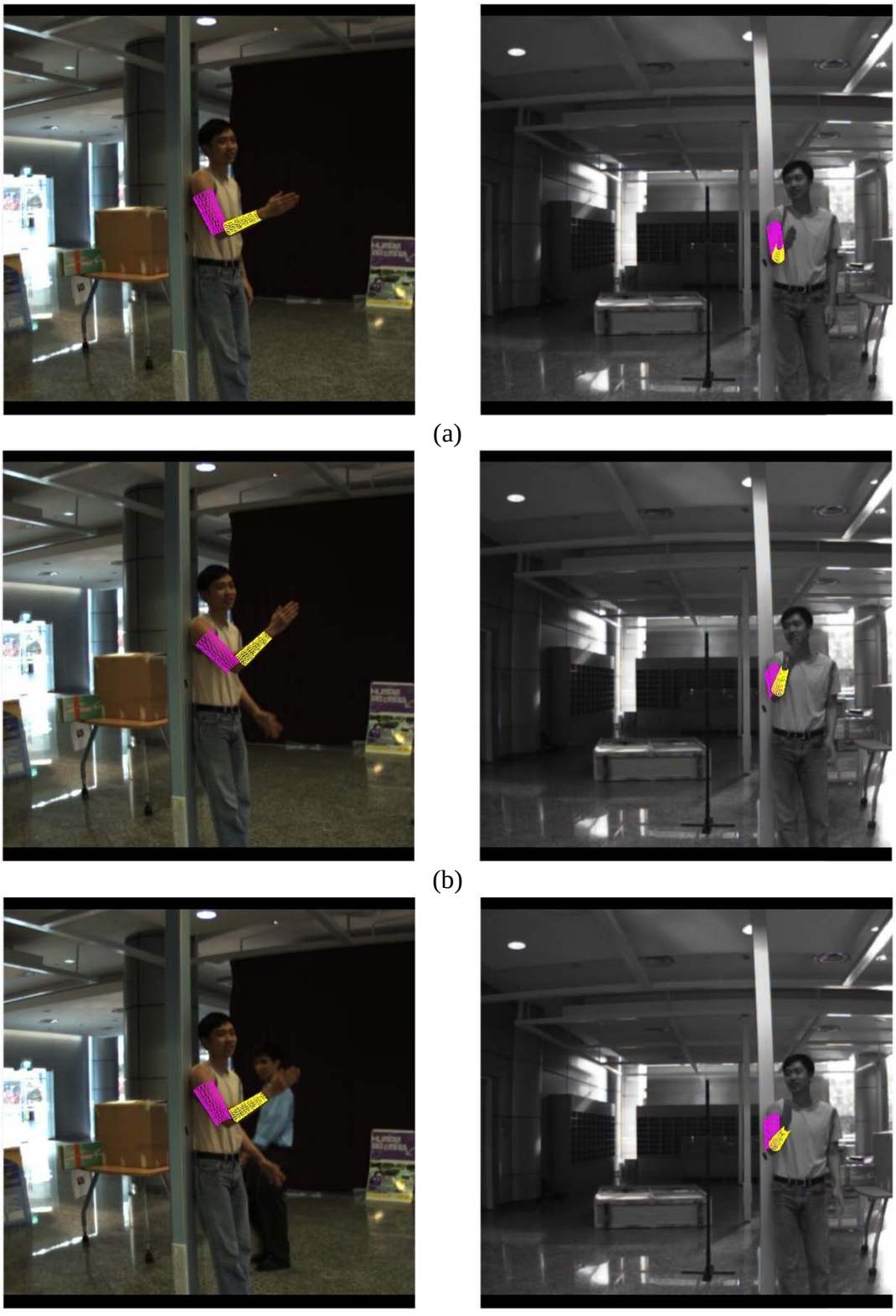
Figure 7.6. Tracking of simple arm movement

(d)

(e)

(f)

Figure 7.6 (con't) Tracking of simple arm movement



(a)

(b)

(c)

Figure 7.7. Tracking in cluttered and moving background

(d)

(e)

(f)

Figure 7.7 (con't) Tracking in cluttered and moving background

(a)

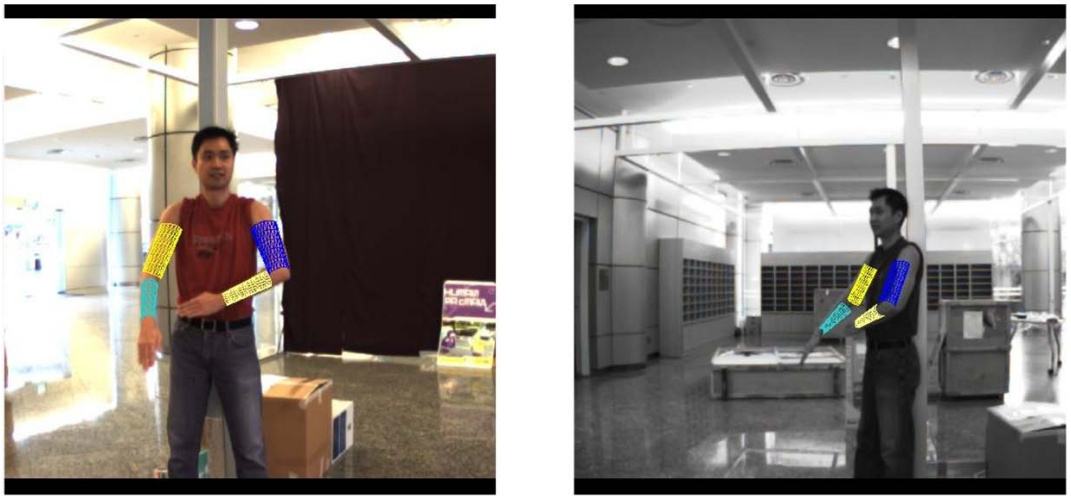
(b)

(c)

Figure 7.8. Tracking with self-occlusion

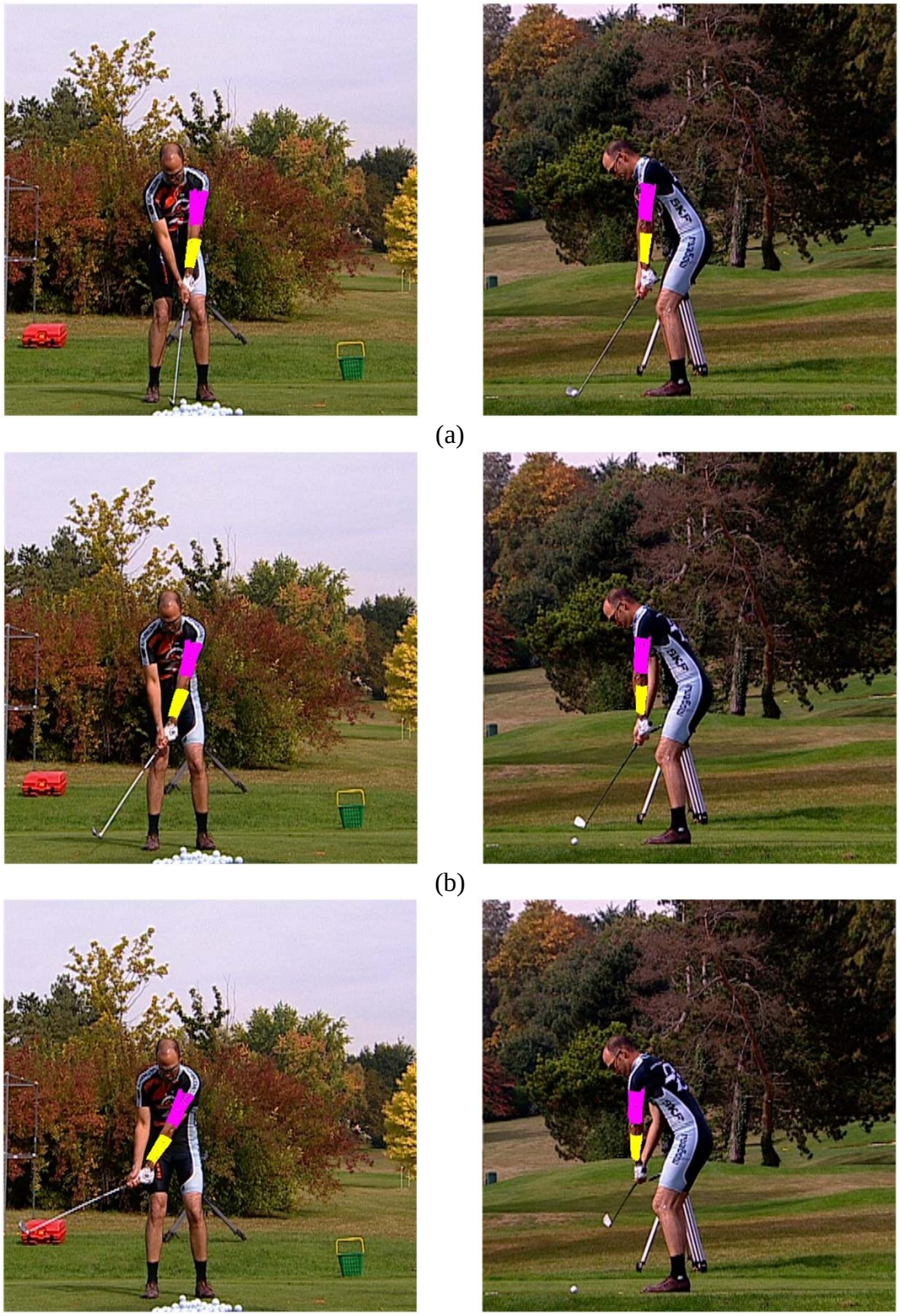
(d)

(e)



(f)

Fig. 7.8 (con't) Tracking with self-occlusion



(a) (b) (c)

Figure 7.9. Tracking in cluttered outdoor environment

(d)

(e)

(f)

Figure 7.9. (con't) Tracking in cluttered outdoor environment

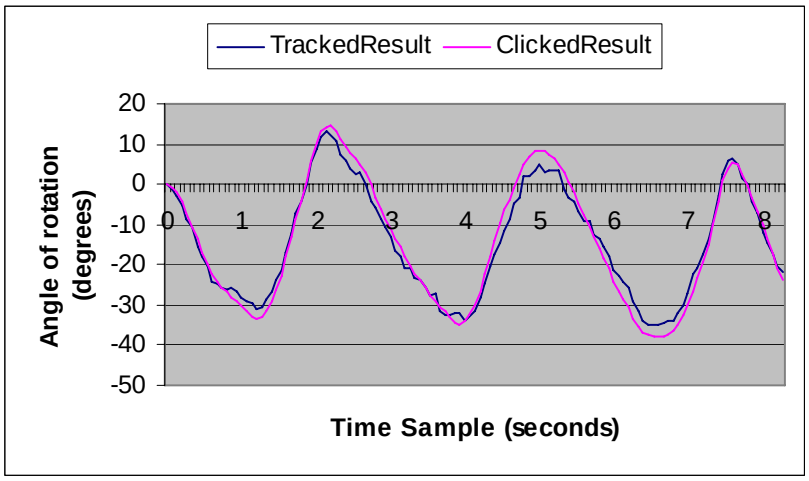


Figure 7.10. Quantitative measurement of automatic elbow bending angle tracking versus manually positioned angles

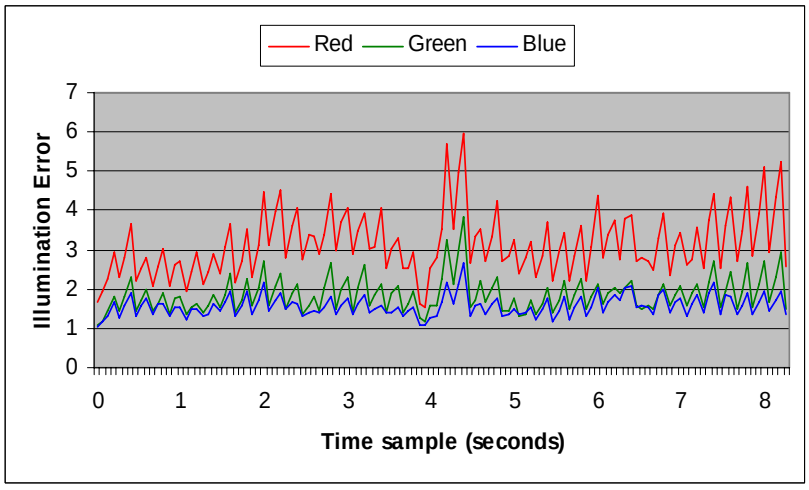


Figure 7.11. Error of the red, green and blue intensities when the error function of equation (7.3) is minimized

Chapter 8.

Discussion and Conclusion

In this dissertation, we had worked on (1) accurate 3D reconstruction of human model from several multi-view wide baseline images and a 3D generic model, of which the average back-projection error of the subject silhouette is less than 1 pixel, and (2) 3D tracking of human motion from video images in the cluttered environment, as we focus on accurate and reliable acquisition of the 3D kinematics of the arms, which is quantitatively small and articulated. This concluding chapter begins by a summary of our work and discussion of the practical aspects of our framework, before finally describing the possible extensions and future directions.

8.1. Summary of Work

We begin in chapter one by indicating our objective and motivation, which is aimed at acquiring the 3D human movements that can be used for bio-mechanical analysis. In chapter two, we reviewed the human motion capture methods in the existing literature, with emphasis on the use of a 3D model. In addition, the issues of skin deformation when the human undergoes motion were discussed. We had also reviewed the methods for construction 3D model of human, which is crucial for motion tracking. We identified that, in general, the need to use 3D model that closely resembles the subject.

A 3D model-based framework was proposed and described in chapter three. We set out to construct 3D human model with the external skin and its internal skeleton that are

customized to specific subject. A cheap and efficient method [118], [119] has been investigated to build the customized 3D human puppet model. It makes use of several uncalibrated wide baseline images and a 3D generic model. This method may be used in disciplines such as sports science [120] and graphical media arts [117]. The 3D model puppet will then be used to facilitate the tracking after it has been pre-positioned.

The first stage of the 3D human model reconstruction is described in chapter 4. After the 2D/3D feature correspondents in the 2D images and generic 3D model are established, 3D reconstruction of feature points and calibration if the cameras are automatically at the same time. This is an iterative process, which at convergence yield the calibrated camera poses and the reconstructed 3D position of the feature points on the model. We showed that our process is able to converge to the correct results as we make analysis with respect to the cases of calibrated cameras, and tested with subjects of different shapes and sizes. The computed 3D characteristic points are interpolated through the radial basis function on the whole generic 3D model to produce the intermediate 3D model. This intermediate model has the global geometry of the subject, but the local limb information needs improvement, which is due to the fact that the feature points alone are too sparse.

We made improvement to the intermediate 3D model via registering its silhouette limb to the silhouettes of the subject in the real images. In chapter 5, the silhouettes from both the intermediate 3D model and 2D images were extracted. And then in chapter 6, the silhouettes were matched to each other before the new 3D position of the model silhouette was computed. The RBF in chapter 4 is appended with the new 3D silhouette information for deforming to produce the external skin of the final model. The same radial basis functions have been used to deform the internal 3D skeleton to complete the customized

3D human model of individual subjects. We showed in our results that the 3D human models are properly fitted through the back-projection onto their respective images, which gave an average error of less than one pixel.

In chapter 7, an analysis-by-synthesis method to track the 3D trajectory of the human movement was developed and investigated. The simulated annealing optimization was used as the kernel to search for the correct human posture at each time step by matching images of the textured 3D model from its synthesized posture with the real images. We showed that our method is able to track the trajectories of the human arms which are quantitatively small and articulated. Our algorithm is also able to perform well in situations of self-occlusion, cluttered background, in outdoors, and be used for sports bio-mechanical applications [121].

8.2. Possible Extensions & Future Work

Starting from a framework for 3D human motion tracking, this thesis covered the elements for 3D human modeling and movement tracking. However, a fully complete 3D human modeling & tracking system for all scenarios covers a very wide scope. There are several research areas that we can explore to furnish a more complete human tracking system. These possible improvements could be easily integrated into the existing framework.

Human Skin Deformation and Shading

In practice, the human undergoing motion will cause the skin to deform in non-rigid manners. The accuracy and reliability of human tracking can be improved if these non-rigid deforms can be closely modeled. In computer graphics animation terms, the surface geometries of the animated characters could have the attributes of: twisting, bulging,

bending, stretching and squashing. The scientific principles of these attributes could be used to derive and synthesize the proper surface geometries for use in motion tracking.

The shading for the synthesized image also must not to be overlooked. Although very realistic images had been synthesized before [11], however, to put use it in practice together with proper skin deformation, different lighting condition and computationally efficient is still a big challenge.

Automatic Posture Pre-positioning

In this thesis, pre-position had been done interactively by using 3D graphical tools. To initiate the tracking automatically, the initial posture of the subject needs to be recognized by the machine vision system. The 3D geometries of the postured model and subject images have to be properly registered. We also have to take into account the skin deformation while manipulating the kinematics of 3D model at the stanza posture to the initialized posture. One suggestion to do this is through machine learning of example postures from multiple subjects, and then synthesize them during an automatic pre-positioning operation.

Full-body and Multiple People Tracking

A natural extension to the work from this dissertation is to perform full-body and eventually multiple people tracking in cluttered environments. For full-body tracking of single subject, the complete 3D body model of the subject will be used, thus more degree-of-freedom and possibly more accurate skinning function have to be used. For extension to multiple people tracking, we have to consider occlusion of subjects and detecting collision

and interactions between subjects. A higher level data topological relationship may also be needed if there are many subjects in the scene.

Computational Efficiency

Computational speed is a factor of consideration if we are to add in more sophisticated operations involving the likes of skin deformation, automatic posture pre-position and multiple 3D tracking. Specific dedication is needed to make sure that the algorithms that manipulate the model geometries, data synthesis and matching take place efficiently so that the whole tracking process can yield the desire results in a realistic time-frame. Efficient data streaming is also required, since the large amount of data transfer within the module of the tracking system can be the bottleneck.

Appendix A

Simplex Simulated Annealing

Simulated annealing (SA) is a class of stochastic relaxation algorithms that employed partial random search of the solution space used for numerical optimization. It is known that SA is able to yield the global minima solution to a non-convex function through an iterative process. Unlike gradient-based methods that always go downhill with respect to the criterion function, SA allows a random basis hill-climbing, which is determined by its instant system temperature. This appendix described the SA's temperature scheduling process, incorporating with the metropolis algorithm and Nelder-Mead downhill simplex to search for the unknown variables.

A.1. Temperature Scheduling

SA's probability of accepting uphill moves is controlled by a temperature parameter. The process starts by first "exciting" the system at a sufficiently high temperature so that almost all random moves are accepted. Then the temperature is lowered slowly according to a "cooling" regime. At each temperature the "simulation" or cooling process must be long and smooth enough for the system to settle into a "steady-state". The sequence of melting temperatures and number of perturbations at each temperature constitute the "annealing schedule". The work by the Geman and Geman in [51] proved that the following temperature schedule:

$$T = \frac{t}{\ln(i+1)} \tag{A.1}$$

where t is a constant and i is the iteration cycle, allows the result to reach a global minimum of the function.

A.2. The Metropolis Algorithm

In the Metropolis method, for a vector **u** of n-dimensional unknown variables that we seek, a new candidate solution is generated randomly for each iterative step. If the new candidate reduces the criterion function, it is accepted; otherwise, it is accepted according to an exponential probability distribution, given the instantaneous system temperature T:

$$P = \begin{cases} e^{-\Delta E/T} & \text{if } \Delta E > 0 \\ 1 & \text{if } \Delta E \le 0 \end{cases} \qquad \text{(A.2)}$$

where E is the current value of the criterion, which depends on T.

However, we have to choose a new candidate solution at an iteration m to (m+1) based on $\mathbf{u}^m$ i.e. for $\mathbf{u}^{(m+1)} = \mathbf{u}^m + \Delta\mathbf{u}$. As explained in [116], a generator of random changes is inefficient, when local downhill moves exist. A good generator for candidate solutions should not become inefficient in narrow valleys; nor should it become more and more inefficient as convergence to a minimum is approached. One way to take care of the possible inefficiency is to perform simulated annealing minimization incorporated with modification to the Nelder-Mead downhill simplex.

A.3. Nelder-Mead Simplex Directional Search

The Nelder-Mead Simplex performs a search in n-dimensional continuous space without the calculation of function derivatives. The simplex method tends to operate well up to 10 unknown variables, above this number convergence becomes difficult. A simplex is a structure in n-dimensional space formed by (n + 1) points that are non-planar, i.e. triangle

for a 2-dimensional simplex, and tetrahedron in 3-dimensional simplex. The simplex is also known as hyper-tetrahedron, and is always a convex hull.

The Nelder-Mead optimization starts with a simplex of (n+1) points for n-dimensional minimization, and then modifies the simplex at each iteration using five simple operations: reflection, expansion, shrinking, inside contraction and outside contraction. Its aim is to trap the solution inside its convex hull, as it homes into it. A typical iteration simplex minimization is as follows:

Denote $\mathbf{u}$ as the vector unknown variables, we first find $\mathbf{u}_w$, $\mathbf{u}_b$ that give rise to the worst and best criterion functions $E(\mathbf{u})$ respectively from the sets of $(n+1)$ $\mathbf{u}$ that make up the hyper-tetrahedron, and then we calculate the average $\mathbf{u}_a$, which exclude $\mathbf{u}_w$:

$$\mathbf{u}_a = \frac{1}{n} \sum_{i=1, i \neq w}^{n+1} \mathbf{u}_i \tag{A.3}$$

Next $\mathbf{u}_a$ forms the centroid of *reflection* and the vector from $\mathbf{u}_w$ to $\mathbf{u}_a$ give the direction of *reflection*, a new candidate solution $\mathbf{u}_r$ is obtained:

$$\mathbf{u}_r = \mathbf{u}_a + \alpha(\mathbf{u}_a - \mathbf{u}_w) \tag{A.4}$$

where α is usually selected to be 1.0.

If the function evaluation $E(\mathbf{u}_r)$ is better than $E(\mathbf{u}_b)$, then the reflected point will be accepted, and we attempt to step further in the same direction by performing an *expansion*:

$$\mathbf{u}_e = \mathbf{u}_r + \gamma(\mathbf{u}_r - \mathbf{u}_a) \tag{A.5}$$

where γ is usually set to 1.0.

If, however the function evaluated via the reflected point is worse than the current worst point, the computation assumes that a better solution exist between $\mathbf{u}_w$ and $\mathbf{u}_a$, thus executing an *inside contraction*:

$$\mathbf{u}_c = \mathbf{u}_a - \beta(\mathbf{u}_a - \mathbf{u}_w) \tag{A.6}$$

the inside contraction factor β is usually set to 0.5.

If the function evaluated via the reflected point is not worse than the worst point but still worse than the second worst point $\mathbf{u}_l$, then an outside contraction is performed:

$$\mathbf{u}_o = \mathbf{u}_a + \beta(\mathbf{u}_a - \mathbf{u}_w) \tag{A.7}$$

And if all the reflection, expansion and contractions fail, the *shrinking* operation will take place. This operation retains the best point and shrinks the simplex i.e. for all points of the simplex except the best one, we compute a new solution:

$$\mathbf{u}_i = \mathbf{u}_b - \rho(\mathbf{u}_i - \mathbf{u}_b), \ \forall i, i \neq b \tag{A.8}$$

where the shrinking factor ρ is usually chosen to be 0.5.

Figure A.1 shows the simplex operation for 2-dimensional computation, and Figure A.2 is a flow chart of the Nelder-Mead simplex minimization.

A.4. Putting Together the Simplex Simulation Annealing

One practical method for simulated annealing minimization in continuous-space variables by [51] uses the scheduled annealing temperature to control the probability of accepting the uphill move in the Nelder-Mead simplex determined by a modified Metropolis procedure. The outer loop of the minimization is run by the temperature schedule, which controls an inter loop of the downhill simplex.

At a given temperature, the downhill simplex iterates until convergence or until the maximum number of iterations is reached. Perturbation noises, proportional to the temperature *T*, are added to (1) the values of the error function evaluated at all the vertices from the simplex, and (2) to the function values of every simplex "moves" that tries for a

candidate solution. In the case when the temperature T → 0, this algorithm reduces to exactly to the original downhill simplex.

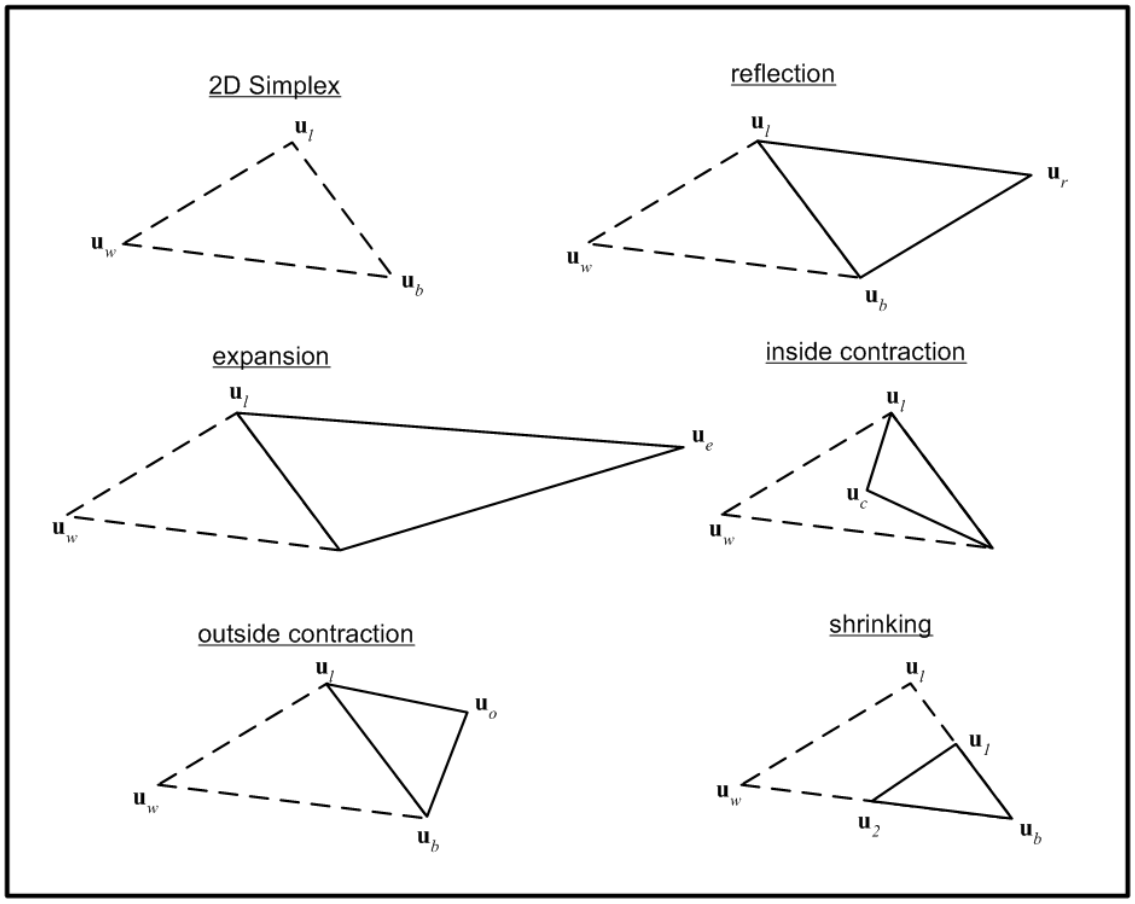


Figure A.1. Illustrating simplex operations for 2-dimensional computation

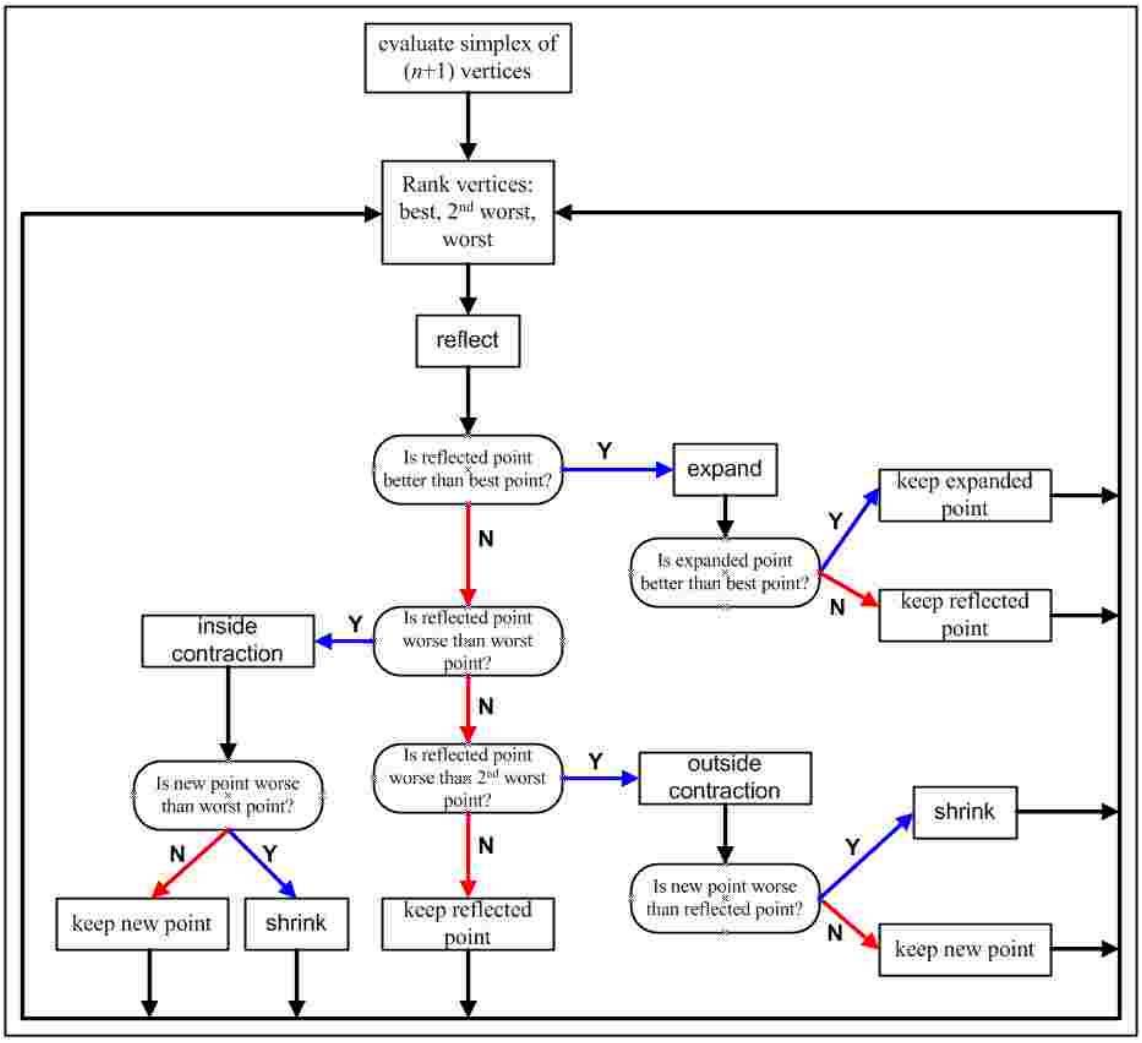


Figure A.2. Flow chart of the Nelder-Mead simplex minimization

Appendix B

Feature Points on the Subject's Images

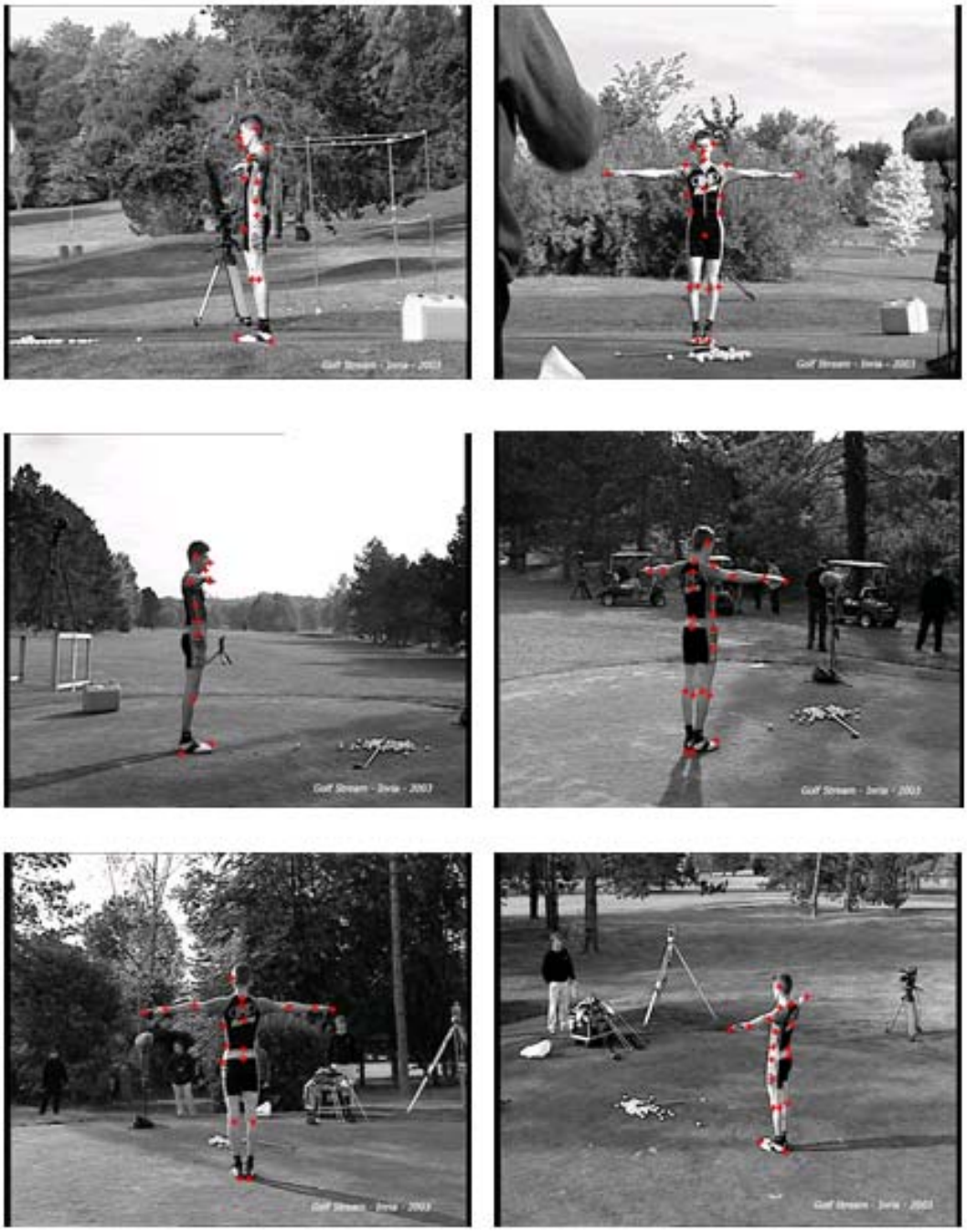


Figure B.1. Feature points on the 'small-man' subject in the 2D images

Figure B.2. Selected feature points on the 'big-man' subject in the 2D images

References

1. Agarwal A and Triggs B. Monocular Human Motion Capture with a Mixture Classifier. *IEEE Conf. on Computer Vision and Pattern Recognition - Workshop on Vision for Human Computer Interaction*, pp 72-79, 2005.
2. Aggarwal J K and Cai Q. Human Motion Analysis: A Review. In *Proc. of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, Austin, pp 2-14, 1994.
3. Allen B, Curless B and Popovic Z. Articulated Body Deformation from Range Scan Data. *ACM Trans. on Graphics*, 21, pp 612-619, 2002.
4. Allen B, Curless B and Popovic Z. The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans. *SIGGRAPH*, pp 587-594, 2003.
5. Aubel A. Anatomically Based Human Body Deformation. *PhD thesis, Ecole Polytechnique Fererale de Lausane*, Switzerland, 2002.
6. Azarbayejani A and Penland A. Recursive Estimation of Motion, Structure and Focal Length. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp 562-575, 1995.
7. Baron C and Kakadiaris I A. On the Improvement of Anthropometry and Pose Estimation from a Single Uncalibrated Image. *Machine Vision and Applications*, pp 229-236, 2003.
8. Baumberg A. Reliable Feature Matching Across Widely Separated Views. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 774-781, 2000.
9. Blake A and Isard M. Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion. *Springer*, 2000.
10. Blanz V and Vetter T. A Morphable Model for the Synthesis of 3D Faces. *SIGGRAPH*, pp 187-194, 1999.

11. Boivin S and Gagalowicz A. Image-based Rendering of Diffuse, Specular and Glossy Surface from a Single Image. *SIGGRAPH*, pp 107-116, 2001.

12. Bottino A and Laurentini A. A Silhouette Based Technique for the Reconstruction of Human Movement. *Computer Vision and Image Understanding*, vol. 83, pp 75-95, 2001.

13. Bottino A and Laurentini A. Introducing a New Problem: Shape-from-silhouette When tbe Relative Positions of the Viewpoints is Unknown. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, pp 1484-1493, 2003.

14. Bouguet J-Y. Camera calibration toolkit for Matlab and in OpenCV.

15. Boyer E. On Using Silhouettes for Camera Calibration. *Asian Conference on Computer Vision*, pp 1-10, 2006.

16. Boykov Y and Kolmogorov V. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, pp 1124-1137, 2004.

17. Bradski G R and Davis J W. Motion Segmentation and pose recognition with motion history gradients. *Machine Vision and Applications*, vol. 13, pp 174-184, 2002.

18. Bregler C. Learning and Recognizing Human Dynamics in Video Sequences. *IEEE Conf. on Computer Vision and Pattern Recognition 7*, pp 568-574, 1997.

19. Bregler C, Malik J and Pullen K. Twist Based Acquisition and Tracking of Animal and Human kinematics. *International Journal of Computer Vision*, vol. 56, pp 179-194, 2004.

20. Byrnes D and Li L. Joining NURBS-based Body Sections for Human Character Animation. *International Conference on Computer Vision and Graphics*, Poland, 2005.

21. Canny J. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp 679-698, 1986.

22. Cappozzo A, Cappello A, Croce U D and Pensalfini F. Surface-Marker Cluster Design Criteria for 3-D Bone Movement Reconstruction. *IEEE Trans. Biomedical Engineering*, vol. 44, pp 1165-1174, 1997.

23. Carranza J, Theobalt C, Magnor M and Seidel H-P. Free-Viewpoint Video of Human Actors. *SIGGRAPH*, pp 569-577, 2003.

24. Chetvrerikov D, Megyesi Z and Janko Z. Finding region correspondence for wide baseline stereo, *Intl. Conf. of Pattern Recognition*, pp 276-279, 2004.

25. Cheung G, Baker S and Kanade T. Shape-from-silhouette for articulated objects and its use for human body kinematics estimation and motion capture. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 77-84, 2003.

26. Cheung G, Baker S and Kanade T. Visual Hull Alignment and Refinement Across Time: A 3D Reconstruction Algorithm Combining Silhouette with Stereo. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 375-382, 2003.

27. Christy S and Horaud R. Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, pp 1098-1104, 1996.

28. Cipolla R, Astrom K E and Giblin P J. Motion from the frontier of curved surfaces. *Intl Conf. on Computer Vision*, pp 269-275, 1995.

29. Ciraba A, Marchesi M, Martin M and Ridella S. Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm. *ACM Trans. Mathematical Software*, vol. 13, pp 262-280, 1987.

30. Corazza S, Mundermann L and Andriacchi T. The Evolution of Methods for the Capture of Human Movement Leading to Markerless Motion Capture for Biomechanical Applications. *Journal Neuroengineering and Rehabilitation*, vol. 3, online version, 2006.

31. Davis J, Nehab D, Ramamoothi R and Rusinkiewicz S. Spacetime Stereo : A Unifying Framework for Depth from Triangulation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp 290-302, 2005.

32. Dellas F and Reveret L. Video Capture of Skin Motion using Calibrated Fabrics. *Workshop on Modelling and Motion Capture Techniques for Virtual Environments (CAPTECH)*, 2004.

33. Dementhon D F and Davis L. Model-based Object Pose in 25 Lines of Code. *Intl. Journal of Computer Vision*, vol. 15, pp 123-141, 1995.

34. Deutscher J and Reid I. Articulated Body Motion Capture by Stochastic Search. *International Journal of Computer Vision*, vol. 61, pp 185-205, 2005.

35. Demirdjian D, Ko T and T Darnell T. Constraining Human Body Tracking, *Intl Conf. on Computer Vision*, pp 1071- 1078, 2003.

36. Devebec P. Modeling and Rendering Architecture from Photographs. *PhD dissertation*, University of Berkeley, USA, 1996.

37. Dorai C and Venkatesh S. Media Computing: Computational Media Aesthetics (The International Series in Video Computing). *Springer*, Edition 1, 2002.

38. Duchon J. Splines Minimizing Rotation Invariant Semi-norms in Sobolev Spaces. *Constructive Theory of Functions of Several Variables*, Lecture Notes in Mathematics 571, pp 85-100, 1977.

39. Eberly D. 3D Games Engine Architecture. *Elsevier, Morgan Kaufmann Publisher, 2005*.

40. Eisert P, Steinbach E and Girod B. Automatic Reconstruction of Stationary 3-D Objects from Multiple Uncalibrated Camera Views. *IEEE Trans. Circuits and Systems for Video Technology: Special Issue on 3D Video Technology*, vol. 10, pp 261-277, 2000.

41. Elgammal A M and Davis L. Probabilistic Framework for Segmenting People under Occlusion. *Intl Conf. on Computer Vision*, pp 145-152, 2001.

42. Fablet R and Black M. Automatic Detection and Tracking of Human Motion with a View-based Representation. *European Conf. on Computer Vision*, pp 476-491, 2002.

43. Faugeras O and Robert L. What can two images tell us about a third one? *Intl. Journal of Computer Vision*, vol. 18, pp 5-19, 1996.

44. Gagalowicz A. Collaboration between Computer Graphics and Computer Vision. *Intl. Conf. for Computer Vision*, pp 733-737, 1990.

45. Gagalowicz A. Modeling Complex Indoor Scenes using an Analysis/Synthesis Framework. *Scientific Visualization Advances and Challenges, Springer Verlag*, 1994.

46. Gagalowicz A. Tools for Advanced Telepresence Systems. *Computers & Graphics*, vol. 19. pp 73-88, 1995.

47. Gagalowicz A. Applications of Analysis/Synthesis Collaboration Techniques to Post-production. *Tutorials of IEEE Multimedia Conf.*, Austin, USA, 1998.

48. Gagalowicz A and Gerard P. Three Dimensional Object Tracking using Analysis/Synthesis Techniques. *Confluence of Computer Vision and Computer Graphics, Kluwer Academic Publishers*, chapter 17, 2000.

49. Garvrila D M and Davis L. 3-D Model-based Tracking of Humans in Action: A Multi-view Approach. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 73-80, 1996.

50. Gavrila D M. The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding*, vol. 73, pp 82-98, 1999.

51. Geman S and Geman D. Stochastic Relaxation, Gibbs distribution, and Bayesian restoration of images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp 721-741, 1984.

52. Gerard P and Gagalowicz A. Human Body Tracking using a 3D Generic Model Applied to Golf Swing Analysis. *MIRAGE 2003*, 2003.

53. Goh W B and Chan K Y. The Multiresolution Gradient Vector Field Skeleton. *Pattern Recognition*, vol. 40, pp 1255-1269, 2007.

54. Goncalves L, Di Bernardom E, Ursella E and Perona P. Monocular Tracking of the Human Arm in 3D. *Intl Conf. on Computer Vision*, pp 764-770, 1995.

55. Grest D, Woetzel J and Koch R. Nonlinear Body Pose Estimation from Depth Images. *DAGM - German Association for Pattern Recognition*, pp 285-292, 2005.

56. Guo Y, Xu G and Tsuji S. Tracking Human Body Motion Based on a Stick Figure Model. *Journal of Visual Communication and Image Representation*, vol. 5, pp 1-9, 1994.

57. Harley R. Calibration using Line Correspondence. In *Proc. of DARPA Image Understanding Workshop*, pp 361-366, 1993.

58. Hartley R. Projective Reconstruction and Invariants from Multiple Images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp 1036-1041, 1994.

59. Heikkilä J. Geometric Camera Calibration Using Circular Control Points. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp 1066-1077, 2000.

60. Herda L, Fua P, Plankers R, Boulic R and Thalmann D. Using Skeleton-Based Tracking to Increase the Reliability of Optical Motion Capture. *Human Movement Science Journal*, vol. 20, pp 313 - 341, 2001.

61. Herda L, Urtasun R and Fua P. Heirachical Implicit Surface Joint Limits for Human Body Tracking. *Computer Vision and Image Understanding*, vol. 99, pp 189-209, 2005.

62. Hilaga M, Shinagawa Y, Kohmura T and Kunii T L. Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes. *Proc. of SIGGRAPH 2001*, pp 203-212, 2001.

63. Horaud R and Dornaika F. Object Pose: Link between Weak Perspective, Paraperspective, and Full Perspective. *Intl. Journal of Computer Vision*, vol. 22, pp 173-189, 1997.

64. Huber P J. Robust Statistics. *Wiley*, 1981.

65. Hyun D-E, Yoon S-H, Chang J-W, Seong J-K, Kim M-S and Juttler B. Sweep-based Human Deformation. *The Visual Computer*, vol. 21, pp 542--550. 2005.

66. Isard M and Blake A. Condensation – Conditional Density Propagation for Visual Tracking. *Intl. Journal of Computer Vision*, vol. 29, pp 5-28, 1998.

67. Isenberg T, Freudenberg B, Halper N, Schlechtweg S and Strothotte T. A Developer's Guide to Silhouette Algorithms for Polygonal Models. *IEEE Computer Graphics and Applications*, vol. 23, pp 28-37, 2003.

68. Kakadiaris I A and Metaxas D. 3D Human Body Acquisition from Multiple Views. *Intl. Journal of Computer Vision*, vol. 30, pp 191-218, 1998.

69. Kakadiaris I A and Metaxas D. Model-based Estimation of 3D Human Motion. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, pp 1453-1459, 2000.

70. Kanade T, Rander P and Narayanan P J. Virtualized Reality: Constructing Virtual Worlds from Real Scene. *IEEE Trans. Multimedia*, vol. 4,pp 34-47, 1997.

71. Kass M, Watkin A and Terzopoulos D. Snake: Active contour models. *Intl. Journal of Computger Vision*, vol. 1, pp 321-331, 1988.

72. Kavan L and Zara J. Spherical Blend Skinning: A Real-time Deformation of Articulated Models. In *Proc. of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pp 9-16, 2005.

73. Kehl R and van Gool L. Markerless Tracking of Complex Human Motions from Multiple Views. *Computer Vision and Image Understanding*, vol. 104, pp 190-209, 2006.

74. Kettner L and Welzl E. Contour Edge Analysis for Polyhedral Analysis. *Geometric Modeling: Theory and Practice*, Springer, pp 379-394, 1997.

75. Kim K, Chalidabhongse T H, Harwood D and Davis L. Real-time Foreground-Background Segmentation using Codebook Model. *Real-time Imaging*, vol. 11, pp 167-256, 2005.

76. Kim M and Pavlovic V. Discriminative Learning of Dynamical Systems for Motion Tracking. *IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

77. Knowssow D, Ronfard R and Horaud R. Tracking with the Kinematics of External Contours. *Asian Conference on Computer Vision*, page 664-673, 2006.

78. Krahnstoever N, Yeasin A and Sharma R. Automatic Acquisition and Initialization of Articulated Models. *Machine Vision and Applications*, vol. 14, pp 218-228, 2003.

79. Lan X and Huttenlocher D. A Unified Spatio-temporal Articulated Model for Tracking. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 722-729, 2004.

80. Laurentini A. How Far 3D Shapes Can Be Understood from 2D Silhouettes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp 188-195, 1995.

81. Laurentini A. The Visual Hull of Smooth Curved Objects. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, pp 1622-1632, 2004.

82. Lenyel J. The Convergence of Graphics and Vision. *Computer*, vol. 31, pp 46-53, 1998.

83. Lerasle F, Rives G and Dhome M. Tracking of Human Limbs by Multicular Vision. *Computer Vision and Image Understanding*, vol. 75, pp 229-246, 1999.

84. Lewis J, Cordner M and Fong N. Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-driven Deformation. *SIGGRAPH*, pp 165-172, 2000.

85. Liebowitz D and Carlsson S. Uncalibrated Motion Capture Expoliting Articulated Structure Constraints. *Intl. Journal of Computer Vision*, vol. 51, pp 171-187, 2003.

86. Luong Q T and Faugeras O D. Self-calibration of Moving Camera from Point Correspondence and Fundamental Matrices. *Intl. Journal of Computer Vision*, vol. 22, pp 261-289, 1997.

87. Ma C M and Sonka M. A Fully Parallel 3D Thinning Algorithm and Its Applications. *Computer Vision and Image Understanding*, vol. 64, pp 420-433, 1996.

88. Matsuyama T, Wu X, Takai T and Nobuhara S. Real-time 3D Shape Reconstruction, Dynamic 3D Mesh Deformation, and High Fidelity Visualization for 3D Video. *Computer Vision and Image Understanding*, vol. 96, pp 393-434, 2004.

89. McKenna S J, Jabri S, Duric Z, Rosenfeld A and Wechsler H. Tracking Groups of People. *Computer Vision and Image Understanding*, vol. 80, pp 42-56, 2000.

90. Menier C, Boyer E and Raffin B. 3D Skeleton-Based Body Pose Recovery. *Intl. Symposium on 3D Data Processing, Visualization, and Transmission*, pp 389-396, 2006.

91. Metaxas D and Terzoulos D. Shape and Non-rigid Motion Estimation through Physics-based Synthesis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp 580-591, 1993.

92. Mikic I, Trivedi M, Hunter E and Cosman P. Human Body Model Acquisition and Tracking using Voxel Data. *Intl. Journal of Computer Vision*, vol. 53, pp 199-223, 2003.

93. Mittal A and Davis L. M2Tracker: A Multi-view Approach to Segmenting and Tracking People in a Clutter Scene. *Intl. Journal of Computer Vision*, vol. 51, pp 189-203, 2003.

94. Mokhtarian F and Mackworth A K. A Theory of Multiscale, Curvature-Based Shape Representation for Planar Curves. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, pp 789-805, 1992.

95. Moselund T B and Granum E. Multiple Cues used in Model-based Human Motion Capture. *Intl. Conf. on Automatic Face and Gesture Recognition*, pp 362-367, 2000.

96. Moeslund T B and Granum E. A Survey of Computer Vision-based Human Motion Capture. *Computer Vision and Image Understanding*, vol. 81, pp 231-268, 2001.

97. Moselund T, Hilton A and Kruger V. A Survey of Advances in Vision-based Human Motion Capture and Analysis. *Computer Vision and Image Understanding*, vol.104, pp 90-126, 2006.

98. Mukasa T, Nobuhara S, Maki A and Matsuyama T. Finding Articulated Body in Time-series Volume Data. *4th Intl. Conf. on Articulated Motion and Deformable Objects*, LNCS 4069, pp 395-404, 2006.

99. Mundermann L, Corazza, Chaudhari A M, Andriachhi, Sundearesan A and Chellappa R. Measuring Human Movement for Biomechanical Applications using Markerless Motion Capture. *In Three-Dimensional Image Capture and Applications VII, Prod. SPIE*, vol. 6065, pp 246-255, 2006.

100. Ng T K. PALM: Portable Sensor-Augmented Vision System for Large Scene Model. *PhD dissertation, CMU-RI-TR-99-27*, Carnegie Mellon University, USA, 1999.

101. Nobuhara S and Matsuyama T. Deformable Mesh Model for Complex Multi-Object 3D Motion Estimation from Multi-Viewpoint Video. *3rd Intl. Symposium on 3D Processing, Visualization and Transmission (3DPVT 2006)*, pp 264-271, 2006.

102. Nobuhara S, Tsuda Y, Matsuyama T and Ohama I. Multi-viewpoint Silhouette Extraction with 3D Context-aware Error Detection, Correction and Shadow Suppression. *4th European Conf. on Visual Media Produection (CVMP 2007)*, 2007.

103. O'Brien J F, Bodenheimer Jr R E, Brostow G J and Hodgins J K. Joint parameter estimation from magnetic motion capture data, *Proceedings of Graphics Interface '00*, pp 53-60, 2000.

104. Ozer I B and Wolf W H. A Hierarchical Human Detection System in (Un) Compressed Domains. *IEEE Trans. Multimedia*, vol. 4, pp 283-300, 2002.

105. Panjabi M M, Goel V K and Walter S D. Errors in the Centre and Angle of Rotation of a Joint: An Experimental Study. *Journal of Biomechanics*, vol. 15, pp 537-544, 1982.

106. Park J, Park S and Aggarwal J K. Human Motion Tracking by Combining View-based and Model-based Methods for Monocular Video Sequences. *Computational science and its applications – ICCSA'03*, LNCS 2669, pp 650-659, 2003.

107. Park S and Aggarwal J K. Simultaneous Tracking of Multiple Body parts of Interacting Persons. *Computer Vision and Image Understanding*, vol. 102, pp 1-21, 2006.

108. Park S I and Hodgin K J. Capturing and Animating Skin Deformation in Human Motion, *ACM Trans. on Graphics*, vol. 25, pp 881-889, 2006.

109. Pavlovic V, Rehg J M, Cham T J. A Dynamic Bayesian Network Approach to Tracking using Learned Switching Dynamic Models. *Advances in Multimodal Interfaces – ICMI 2000*, LNCS 1948, pp 308-316, 2000.

110. Perales F J, Buades J M, Mas R, Varona X, Gonzalez M, Suescun A, Aguinaga I, Foursa M, Zissis G, Touman M and Mendoza R. A New Human Motion Analysis

System using Biomechanics 3D Model. *Poster Proc. in Intl. Conf. on Computer Graphics and Interaction Techniques SIGGRAPH*, pp 83, 2004.

111. Planker R and Fua P. Tracking and Modeling People in Video Sequences. *Computer Vision and Image Understanding*, vol. 81, pp 285-302, 2001.

112. Plankers R and Fua P. Articulated Soft Objects for Multiview Shape and Motion Capture. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, pp 1182-1187, 2003.

113. Poelman C J and Kanade T. A Paraperspective Factorization Method for Shape and Motion Recovery, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp 206-218, 1997.

114. Polleyfrey M, Koch R and Gool L V. Self-Calibration and Metric Reconstruction Inspite of Varying and Unknown Intrinsic Camera Parameters. *Intl. Journal of Computer Vision*, vol. 32, pp 7-25, 1999.

115. Ponce J, Papadopoulo T, Teillaud M and Triggs B. On the Absolute Quadric Complex and its Application to Autocalibration. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 780-787, 2005.

116. Press W H, Teukolsky S A, Vetterling W T and Flannery B P. Numerical Recipes in C: The Scientific Computing 2nd Edition. *Cambridge Press*, 1992.

117. Quah C K, Gagalowicz A and Seah H S. 3D Modeling of Human Actors for Free-Viewpoint Video. *Multimedia Art Asia Pacific Conference (MAAP)*, 2004.

118. Quah C K, Gagalowicz A and Seah H S. Modeling 3D Human from Uncalibrated Wide Baseline Views. *MIRAGE 2005*, pg 163-171, INRIA Rocquencort, France, 2005.

119. Quah C K, Gagalowicz A, Roussel R and Seah H S. 3D modeling of humans with skeletons from uncalibrated wide baseline views. *Intl Conf. on Computer Analysis of Images and Patterns*, LNCS 3691, pp 379-389, 2005.

120. Quah C K, Gagalowicz A and Seah H S. An Efficient and Accurate Method for Constructing 3D Human Models from Multiple Cameras. *Asia-Pacific Congress on Sports Technology*, pp 113-119, 2007.

121. Quah C K, Gagalowicz A and Seah H S. Marker-less 3D Video Motion Capture in Cluttered Environments. *Asia-Pacific Congress on Sports Technology*, pp 121-126, 2007.

122. Ram D J, Sreenivas T H and Subramaniam K G. Parallel Simulated Annealing Algorithm. *Journal of Parallel and Distributed Computing*, vol. 37, pp 297-212, 1996.

123. Rehg J, Morris D and Kanade K. Ambitguities in Visual Tracking of Articulated Objects using Two- and Three-dimensional Models. *International Journal of Robotics Research*, vol. 22, pp 393-418, 2003.

124. Remondino F. 3-D Reconstruction of Static Human Body Shape from an Image Sequence. *Computer Vision and Image Understanding*, vol. 93, pp 65-85, 2004.

125. Rohr K. Human Movement Analysis Based on Explicit Motion Model, *Klumer Academic*, chap. 8, pp 171-198, 1997.

126. Rosales R and Sclaroff S. Combining Generative and Discriminative Models in a Framework for Articulated Pose Estimation. *Intl Journal of Computer Vision*, vol. 63, pp 251-276, 2006.

127. Rosenhahn B, He L and Klette R. Automatic Human Model Generation. *Intl Conf. on Computer Analysis of Images and Patterns*, LNCS 3691, pp 41-48, 2005.

128. Rosenhahn B, Klette R and Sommer G. Silhouette Based Human Motion Estimation. *DAGM - symposium German Association for Pattern Recognition*, LNCS 3175, pp. 294-301, 2004.

129. Roussel R and Gagalowicz A. Morphological Adaptation of a 3D model of Face from Images. *MIRAGE 2003*, INRIA Rocquencort, France, 2003.

130. Roussel R and Gagalowicz A. A Hierarchical Face Behavior Model for a 3D Face Tracking without Markers. *Intl Conf. on Computer Analysis of Images and Patterns*, LNCS 3691, pp 854-863, 2005.

131. Sand P, McMillan L and Popovic J. Continuous Capture of Skin Deformation. *ACM Trans. on Graphics*, vol. 22, 578-586, 2003.

132. Scheepers F, Parent R, Carlson W E and May S F. Anatomy-based Modeling of the Human Musculature. *SIGGRAPH*, pp 163-172, 1997.

133. Serra B and Berthod M. Subpixel Contour Matching using Continuous Dynamic Programming. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 202-207, 1994.

134. Shan Y, Liu Z, and Zhang Z. Model-based Bundle Adjustment with Application to Face Modeling. *Intl Conf. on Computer Vision*, pp 644-651, 2001.

135. Shen J, Thalmann N and Thalmann D. Human Skin Deformation from Cross-sections. *Proc. Computer Graphics International '94*, 1994.

136. Shi J and Tomasi C. Good Feature to Track. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 593-600, 1994.

137. Sloan P J, Rose C F and Cohen M F. Shape by Example. *In Proc. of the 2001 Synmposium on Interactive 3D Graphics*, pp 135-143, 2001.

138. Sminchisescu M and Trigg B. Mapping Minima and Transitions in Visual Model. *Intl Journal of Computer Vision*, vol. 61, pp 81-101, 2005.

139. Starck J and Hilton A. Model-based Multiple View Reconstruction of People. *Intl Conf. on Computer Vision*, pp 915-922, 2003.

140. Stauffer C and Grimson W. Adaptive Background Mixture Models for Real-time Tracking, *IEEE Conf. on Computer Vision and Pattern Recognition*, pp 246-253, 1998.

141. Sundaresan A and Chellappa R. Multi-camera Tracking of Articulated Human Motion Using Motion and Shape Cues. *Asian Conference on Computer Vision*, pp 131-140, 2006.

142. Taylor C J. Reconstruction of Articulated Objects from Point Correspondences in a Single Uncalibrated Image. *Computer Vision and Image Understanding*, vol. 80, pp 349-363, 2000.

143. Terzopolus D. Visual Modeling for Computer Animation: Graphics with a Vision. *Computer Graphics*, vol. 33, pp 42-45, 1999.

144. Theobalt C, Aguiar E, Magnor M, Theisel H and Seidel H-P. Marker-free Kinematic Skeleton Estimation from Sequence of Volume Data, *Proc. ACM Virtual Reality Software and Technolog*, pp 57-64, 2004.

145. Triggs B, McLauchlan P, Hartley R and Fitzgibbon A W. Bundle Adjustment – A Modern Synthesis. *Proc. of the Intl. Workshop on Vision Algorithms: Theory and Practice*, LNCS 1883, pp 298-372, 1999.

146. Tsai R Y. An Efficient and Accurate Camera Calibration Technique for 3-D Machine Vision. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 364-374, 1986

147. Ude A. Robust Estimation of Human Body Kinematics from Video. *IEEE intelligent Robots and Systems*, pp 1489-1494, 1999.

148. Ueshiba T and Tomita F. A Factorization Method for Projective and Euclidean Reconstruction from Multiple Perspective Views via Iterative Depth Estimation, *European Conf. on Computer Vision*, pp 296-310, 1998.

149. Wallace T P and Wintz P A. An Efficient Three-Dimensional Aircraft Recognition Algorithm Using Fourier Descriptors. *Computer Graphics and Image Processing*, vol. 13, pp 99-126, 1980.

150. Wechsler H, Duric Z and Li F. Hierarchical Interpretation of Human Activities using Competitive Learning. *Intl. Conf. of Pattern Recognition*, pp 338-341, 2002.

151. Wei S Q and Ma S. Implicit and Explicit Camera Calibration: Theory and Experiments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp 469-480, 1994.

152. Weik S. A Passive Full Body Scan using Shape from Silhouette. *Intl. Conf. of Pattern Recognition*, pp 99-105, 2000.

153. Wilhelms J and Gelder A V. Anatomically based modeling. *SIGGRAPH*, pp 173-180, 1997.

154. William L. Pyramidial Parametrics. *Computer Graphics*, vol. 7, pp 1-11, 1983.

155. Wingbermuhle J, Liedtke C-E and Solodenko J. Automated Acquistion of Lifelike 3D Human Models from Multiple Posture Data. *Intl Conf. on Computer Analysis of Images and Patterns*, LNCS 2124, pp 400-409, 2001.

156. Wren C, Azarbayejani A, Darrell T and Pentland A. Pfinder: Real-time Tracking of the Human Body. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp 780-785, 1997.

157. Wren C and Pentland A. Dynaman: A Recursive Model of Human Motion. *MIT Media Laboratory Technical Report*, no. 451, USA, 1997.

158. Young G S J and Chellappa R. 3-D Motion Estimation using a Sequence of Noisy Stereo Images: Models, Estimation and Uniqueness Results. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp 735-759, 1990.

159. Zhang S and Huang P S. High-resolution, Real-time Three-Dimensional Shape Measurement. *Optical Engineering*, vol. 45(12), pp 123601(1-8), 2006.

160. Zhang Z. Iterative Point Matching for Registration of Free-Form Curves and Surfaces. *Intl. Journal of Computer Vision*, vol. 13, pp 119-152, 1994.

161. Zhang Z. A Flexible New Technique for Camera Calibration. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp 1330-1334, 2000.

162. Zhao J, Ling L and Keong K C. 3D Posture Reconstruction and Human Animation from 2D Feature Points. *Computer Graphics Forum* 24, pp 759-771, 2005.

163. 3DS Max. http://www.autodesk.com

164. Ascension. http://www.ascension-tech.com

165. CLAPACK. http://www.netlib.org

166. Cyberware. http://www.cyberware.com

167. Hamamatsu. http://usa.hamamatsu.com

168. Kington Museum Collection. http://www.kingston.ac.uk/Muybridge.

169. Motion Analysis. http://www.motionanalysis.com

170. Organic mocap. http://www.organicmotion.com

171. Peak Performance. http://www.peakperform.com

172. Polhemus. http://www.polhemus.com

173. Qualisys. http://www.qualisys.com

174. $(TC)^2$. http://www.tc2.com

175. Vicon. http://www.vicon.com

176. Vitronic http://www.vitronic.de

177. Wicks & Wilson. http://www.wwl.co.uk