3D-2D SPATIOTEMPORAL REGISTRATION FOR HUMAN MOTION ANALYSIS

WANG RUIXUAN

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF
PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE

2007

Abstract

Computer systems are increasingly being used to assist coaches in sports coaching. There are two kinds of commercial sports training systems. 3D motion-based systems acquire the performer's 3D motion using an expensive 3D motion capture system in a constrained environment. The performer's 3D motion is then analyzed by the coach or compared with an existing 3D reference motion of an expert by a computer system. 2D video-based systems capture the performer's motion in a single video and display the video beside a pre-recorded expert's video. They do not analyze the performer's video automatically but provide tools for the the coach or the performer to manually compare the performer's motion with the expert's motion. Therefore, these commercially available systems for sports coaching are either not affordable to general users, or do not perform detailed motion analysis automatically.

The goal of this research is to develop an affordable and intelligent sports coaching system for general users. The system captures the performer's motion using a single video camera. It automatically compares the performer's motion with a pre-recorded expert's 3D motion. The performer's motion and the expert's motion may differ in time (e.g., faster or slower) and in space (e.g., different positions and orientations of body parts). So, the system automatically computes the temporal differences and the spatial posture differences between the performer's motion and the expert's motion.

The proposed research problem is by nature very complex. In this thesis, we formulate sports motion analysis as a 3D-2D spatiotemporal motion registration problem. This formulation provides a clear and precise description of the nature and the requirements of the problem, which has not been clearly described in the literature. To solve the problem, a novel framework is developed for analyzing different types of motion by incorporating relevant domain knowledge. We believe that this approach allows us to understand the algorithmic components necessary for analyzing sports motion in general, and to adapt the framework for analyzing various types of motion.

Experiments were designed and performed to quantitatively and qualitatively evaluate the performance of the algorithms using Taichi and golf swing motion as test cases. Test results show that the temporal difference between the two motion sequences can be efficiently and accurately determined. The posture error computed by the algorithms can reflect the performer's actual error in performing the motion. Moreover, the proposed framework can effectively handle ambiguous conditions in a single video such as left-right ambiguity of legs, depth ambiguity of body parts, and partial occlusion. Therefore, this system can provide detailed information for the performer to improve his motion.

Acknowledgements

First of all, I want to give my sincere thanks to my supervisor, Professor Leow Wee Kheng, for his guidance on my research in the past five years. It's Professor Leow who teach me how to do research, how to formulate research problems, and how to write reports, etc. I cannot remember how many detailed and convincing comments and suggestions I have received and accepted from Professor Leow. Without Professor Leow's guidance and help, it is impossible for me to complete the thesis.

I am grateful to Professor Terence Sim, Dr. Ng Teck Khim, and Professor Leong Hon Wai for their valuable suggestions and comments on my research problem and algorithms inside. Thanks to Xing Dongfeng for his collaboration in developing the prototype software system for golf training based on the proposed framework.

During my research, countless discussions with my friends Zhang Sheng, Saurabh Garg, Piyush Kanti Bhunre, and Hanna Kurniawati etc. greatly broaden my knowledge in computer vision and related research fields. At least one or two months were saved by using the prototype software system designed by Saurabh at the beginning of my research. Part of my PhD work in posture estimation is owed to the collaboration with Saurabh. The discussions with Zhang Sheng improve my knowledge and skills not limited to face recognition and related machine learning techniques. In addition, I enjoyed the parties and trips with my friends Ding Feng, Xiaopeng, Xiaoping, etc. I also enjoyed playing badminton with Ehsan and all the other lab mates. Many thanks to Ding Feng, Yingyi, and Zhiyuan for helping to correct errors in the thesis.

Special thanks are given to my family for their infinite love and encouragements in the past years. Thank my wife Jiachao for her understanding and patience over the last several months.

Contents

List of Figures

List of Tables

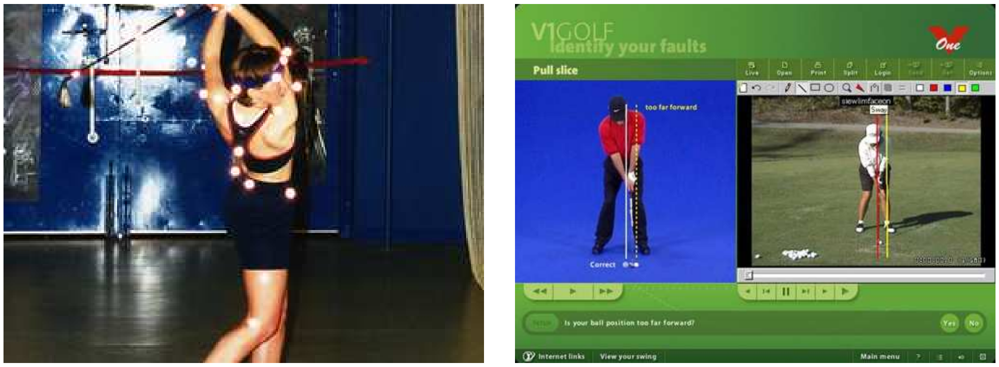
Chapter 1

Introduction

1.1 Motivation

In sports coaching, a coach assesses the movements of a sportsman to provide coaching instructions. The coach analyzes many factors in the sportsman's motion such as the positions, orientations, speeds and motion directions of his body parts. For movements that require precision, such as golf swing, it is very difficult for a human coach to assess the movement quantitatively and precisely without instrumental aids. For long and complex movements such as Taichi, it is impossible for the coach to remember all the mistakes of the performer throughout the whole Taichi sequence. The coach needs to stop the performer's movement to provide coaching instructions, disrupting the smooth flow of the movement. Computer systems can assist the coach in all the above aspects of sports coaching.

There are two kinds of commercially available systems for sports training: 3D motion-based system and 2D video-based system. A 3D motion-based system uses multiple cameras to track the motion of reflective markers attached to the performer's body (Figure 1.1(a)). The markers' 3D positions are recovered and used to compute the performer's 3D motion which includes the temporal sequence of 3D positions and orientations of the performer's body parts. The performer's 3D motion is then analyzed by the coach or compared with an existing 3D reference motion of an expert by a computer system. The coach can then provide timely instructions to the performer. However, such a system is not affordable and suitable for general users. Only professional athletes can afford to pay for the use of such a system within the confinement of a special facility installed with the



(a) (b)

Figure 1.1: Commercial systems for sports motion analysis. (a) Vicon 3D motion capture system captures a performer's golf swing using reflective markers attached to the human body (from http://www.vicon.com/applications/sports.html). (b) V1 Golf Software requires manual comparison of the performer's postures with those of an expert (from http://www.ifrontiers.com/consumer/default.asp).

system.

A 2D video-based system uses a video camera to capture the performer's motion and load the video into a computer system. The computer system typically displays the performer's video and the pre-recorded expert's video side-by-side, and provides tools for the coach or the performer to manually compare the performer's motion with the expert's motion (Figure 1.1(b)). The computer system often lacks the intelligence to perform detailed motion analysis automatically.

The overall goal of this research is to develop an affordable video-based sports coaching system for general use. It should be affordable to general users and can be used any time, anywhere. It should perform intelligent analysis of the performer's motion automatically, and provide detailed feedback to the performer. It helps the performer to understand and improve his motion without the presence of a coach.

1.2 Objectives and Contributions

The specific objective of this research is to develop a system that automatically compares the performer's motion in a single video with the reference motion of an expert (Figure 1.2). The expert's motion is 3D and captured by a motion capture system. The 3D expert motion is captured only once. So, the time and effort spent on 3D motion

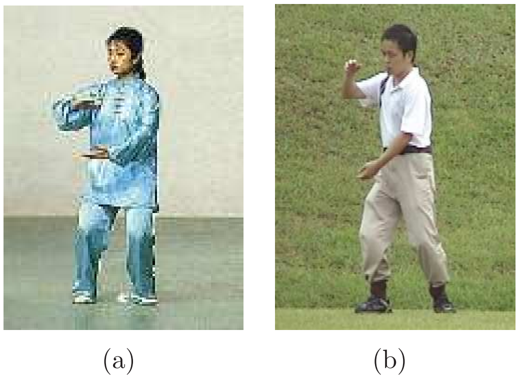


Figure 1.2: Postures of an expert and a performer. (a) An expert's standard posture (from http://www.dzyy.net/books/tjq/24a.htm). (b) A performer's corresponding posture that is slightly different from the expert's posture.

capture is not a major issue. On the other hand, to be practically affordable and easy to use for general users, the performer's motion is 2D and captured by a single video camera. The system can be easily extended to adopt multiple video cameras. In this thesis, we shall focus on the single-camera case which is technically much more challenging. To our best knowledge, this is the first attempt at automatic intelligent computer analysis of sports motion using 2D video as input.

The main contributions of this thesis include the following:

1. Formulate the sports motion analysis problem as a 3D-2D spatiotemporal motion registration problem. In this thesis, we propose a novel and fundamental problem for the analysis of long, complex human motion: 3D-2D spatiotemporal motion registration. The 3D reference motion and the performer's motion in the video may differ in time (e.g., faster or slower) and in space (e.g., different positions and orientations of body parts). The aim of the problem is to automatically determine the temporal differences and the spatial posture differences between the 3D reference motion and the performer's motion in a single video. This problem is by nature a very complex problem, as will be shown in Chapter 2. Our formulation of sports motion analysis as a 3D-2D spatiotemporal motion registration problem provides a clear and precise description of the nature and the requirements of the problem, which has not been clearly described in the literature.

2. Develop a novel framework for analyzing different types of sports motion. Different types of motion have different characteristics. For example, golf swing is a short

and fast motion. The feet are always on the ground, and the hands are close together. In comparison, Taichi is a long, slow and complex motion. The torso is usually upright and every body part is changing position and orientation over time. A straightforward approach for analyzing different types of motion is to develop a specific algorithm for specific motion type. These algorithms cannot be easily extended and adapted to analyze other types of motion. In this thesis, we develop a novel framework that can analyze different types of motion by incorporating relevant domain knowledge. In particular, the 3D reference motion is a form of domain knowledge. Other kinds of domain knowledge can also be incorporated (see Section 2.1.1 for detail). We believe that this approach allows us to understand the algorithmic components necessary for analyzing sports motion in general, and to adapt the framework for analyzing various types of motion.

3. Apply the framework to the analysis of golf swing and Taichi motion. In this thesis, Taichi motion and golf swing are used as the test cases as they represent two very different kinds of sports motion. Successful applications show that the approach of incorporating relevant domain knowledge can indeed allow the framework to analyze different motion types.

1.3 Thesis Organization

The proposed problem is by nature an extremely complex problem. So, it is necessary to analyze the input and output characteristics of the problem, and clearly describe the problem in Chapter 2. Since it is infeasible to directly solve this problem, it is decomposed into four major subproblems (Chapter 2). After formulating the problem, it is now possible to discuss existing work related to it in Chapter 3. This review helps clarify the differences between the proposed problem and those in existing work. Next, the algorithms developed to solve each of the subproblems are described in Chapter 4. The system is applied to the analysis of Taichi motion and golf swing, and its performance is evaluated in Chapter 5. Possible extensions of the system are discussed in Chapter 6. Finally, Chapter 7 concludes the thesis.

Chapter 2

Problem Formulation

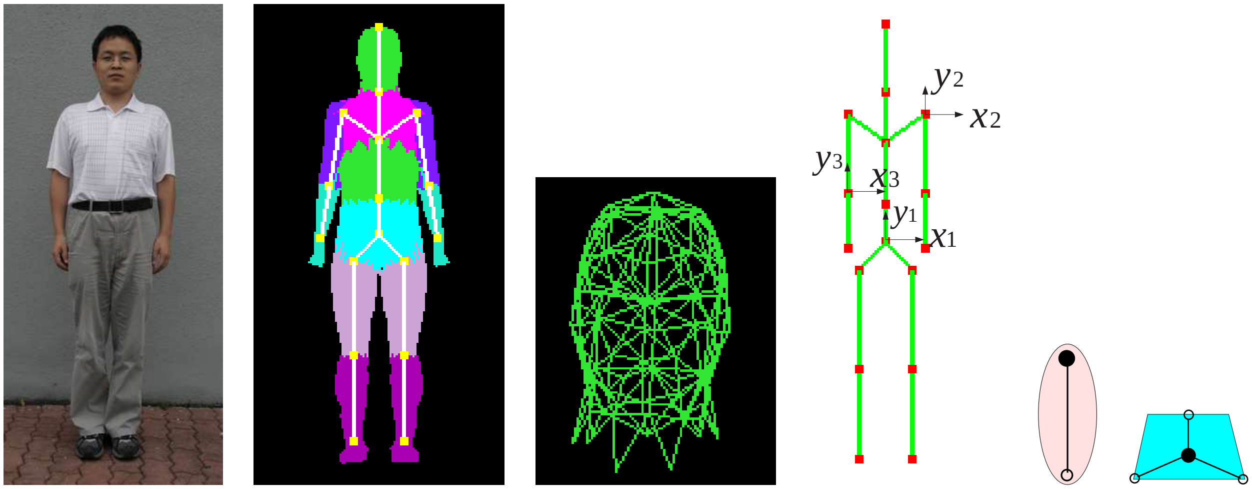
The problem of interest is to determine the difference between the motion of a performer and that of an expert. There are two kinds of difference, temporal and spatial. Temporal difference describes the difference in motion speed, and spatial difference describes the difference in the corresponding postures of the performer and the expert. The determination of temporal and spatial difference is a *spatiotemporal registration* problem. We will formulate the registration problem in detail in Section 2.1. Due to the complexity of the problem, it is infeasible to directly solve the whole problem. Instead, we decompose the problem into a set of subproblems and formulate them respectively (Section 2.2).

2.1 Overall Problem Formulation

To clearly describe the problem, it is necessary to first describe the inputs and the desired outputs of the problem and their characteristics. The inputs consist of 3D reference motion of the expert (Section 2.1.1) and 2D input video of the performer (Section 2.1.2) with complex relationships (Section 2.1.3) between them. The outputs consist of the computed errors between the reference motion and the performer's motion (Section 2.1.4).

2.1.1 3D Reference Motion

The 3D reference motion is the expert's motion. It includes a time-independent component and a time-dependent component:



(a) (b) (c) (d) (e) (f)

Figure 2.1: Human body model and coordinate systems. (a) A real human body image. (b) Human mesh model represents shapes and sizes of human body parts, and human skeleton model includes joints and bones. (c) Triangular mesh for the head. (d) The default posture and the local coordinate system for each body part. Three example local coordinate systems are illustrated. (e) The left upper arm contains one bone (filled circle: parent joint). (f) The root body part contains three bones.

1. Time-independent component: human body model H

   This includes the shapes and sizes of the human body parts (Figure 2.1(b)), joints connecting adjacent body parts and bones connecting adjacent joints (Figure 2.1(d)), and the constraints on the joint rotation angles (Appendix A).

2. Time-dependent component: 3D motion data $\{\mathbf{p}_t, \boldsymbol{\theta}_t\}$

   $\mathbf{p}_t$ is the global position of the human body in the world coordinate system at time t, and $\boldsymbol{\theta}_t$ denotes the rotation angles of the body parts at time t with respect to the default posture (Figure 2.1(d)). $\boldsymbol{\theta}_t$ of default posture is defined as $\mathbf{0}$. Note that $\boldsymbol{\theta}_t$ includes the global orientation of the human body.

   The time-dependent component defines the model's posture at time t denoted as B_t, i.e., $\mathbf{p}_t, \boldsymbol{\theta}_t \in B_t$. The sequence M of B_t, $t \in \mathcal{T} = \{0, \ldots, L\}$, together with the human body model H, is the 3D *reference motion*, and each B_t is a *reference posture* at time t.

The 3D reference motion has the following characteristics:

1. Human body model H consists of a human skeleton model for the body structure

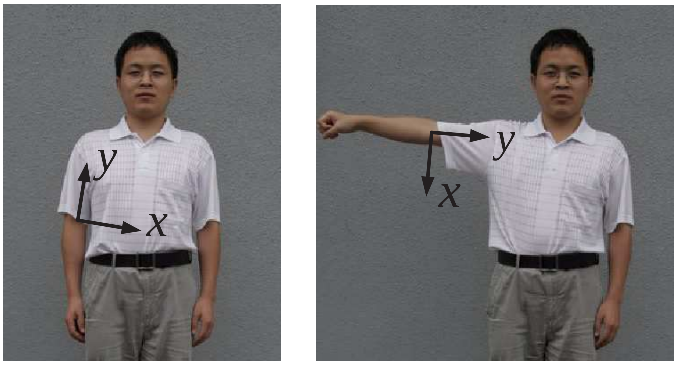
and a triangular mesh model for the body surface (Figure 2.1(b, c)). The human skeleton model consists of 17 joints and end effectors and 16 bones. It is described by a hierarchical structure which is commonly used in Maya [May] and BVH motion files. In the hierarchy, the joint at the root level is called the *root joint*. The root joint is the parent of all the joints at the second level. Every joint at the second level has one or more children. The end effectors, which are the joints at the lowest level in the hierarchy, have no children. In this hierarchical structure, every joint except the root joint is connected to its parent by a bone.

The human body can be divided into 12 body parts (Figure 2.1(b)), namely head, upper chest, lower chest, abdomen (root body part), left/right upper/lower arms, and left/right upper/lower legs. Accordingly, the mesh model is divided into 12 mesh parts and each mesh part corresponds to one unique body part. All the bones connecting to the same parent joint belong to the same body part that is modeled as a rigid object. For example, the root body part contains 3 bones connected to the root joint (Figure 2.1(f)), and the left upper arm contains 1 bone connected to the left shoulder joint (Figure 2.1(e)).

2. In order to determine the global positions and orientations of the body parts easily, both world coordinate system and local coordinate systems are used. Each body part has its own local coordinate system, the origin of which is positioned at the joint connected to its parent body part. The global position of the human body is defined as the 3D coordinates of the root joint in the world coordinate system, and the global orientation of the human body is defined as the orientation of the root body part with respect to the world coordinate system. The rotation angles of other body parts are defined as the rotation of the body parts with respect to the default posture (Figure 2.1(d)) in the corresponding local coordinate systems.

   At the default posture (Figure 2.1(d)), all the local coordinate systems have the same axis directions as those of the global coordinate system, i.e., the x-axis points to the left of the body, the y-axis points up, and the z-axis points to the front (out of the paper, not shown in Figure 2.1(d)). When the human body changes from one posture (Figure 2.2(a)) to another (Figure 2.2(b)), the local coordinate systems may rotate accordingly. At the first frame $t = 0$ in the reference motion, the global coordinate system and the local coordinate system of the root body part are exactly coincident and aligned.

3. The 3D reference motion is a retargetted motion. Motion retargetting is a well-



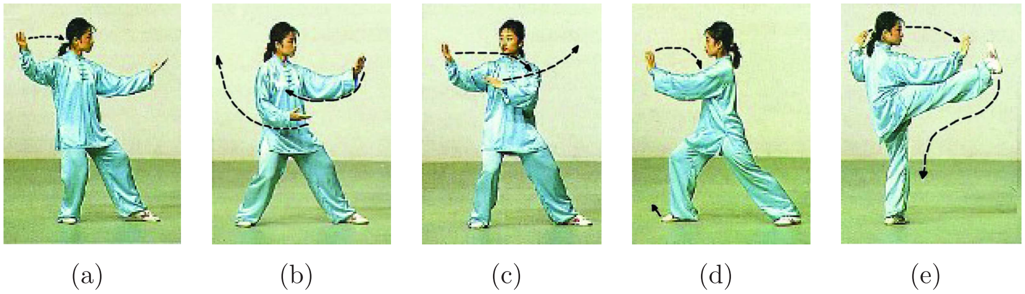
(a) (b)

Figure 2.2: The local coordinate system of the lower arm. (a) The local coordinate system of the elbow at a standing posture. (b) When the parent of the lower arm (i.e., the upper arm) rotates, the local coordinate system of the elbow is also rotated.

known problem in computer animation [Gle98]. It refers to the problem of adapting the motion of a person to another person with a different body size. In general, there are differences in body shape and limb lengths between the expert and the performer. Therefore, the 3D reference motion should be retargetted to fit the performer's body before the reference motion and the performer's motion are compared. Here, we assume that the 3D reference motion has been retargetted to the human body in the input video using, e.g., the algorithm in [Gle98]. That is, the human body model H is that of the performer in the input video and M is retargetted according to H. The shapes and sizes of the performer's body are physically measured in advance. This is a reasonable assumption because retargetting needs to be performed only once for a specific performer. In our application, retargetting adapts the reference motion to the size of the performer. It does not perform any comparison between the reference motion and the performer's input motion.

4. The reference motion can be divided into a set of *motion segments* by a set of *segment boundary* frames $\mathcal{T}_b \subset \mathcal{T}$. These reference segment boundaries are determined based on domain knowledge and are known in advance. Figure 2.3 illustrates some Taichi stances that can be regarded as segment boundaries.

   Based on domain knowledge of the segment boundaries, we find that some body parts change their motion directions significantly across segment boundaries. Let $\mathbf{v}_t$ denote the direction of the 3D velocity of the body part at time t. Then, at segment boundary t, $\mathbf{v}_t \cdot \mathbf{v}_{t+1} < \tau$, where τ is a threshold that depends on the type



(a) (b) (c) (d) (e)

Figure 2.3: Some stances in the Taichi motion that can be regarded as segment boundaries (from http://www.dzyy.net/books/tjq/24a.htm).

of motion. For example, if the direction changes at the segment boundaries are larger than 60°, then τ can be set $\cos 60^\circ = 0.5$.

2.1.2 2D Input Video

The motion of a performer, who is usually a novice, is captured in the input video m' recorded by the camera. The input video m' consists of a sequence of image frames $I'_{t'}$ over time t', $t' \in \mathcal{T}' = \{0, \ldots, L'\}$. Each input image $I'_{t'}$ contains the image of a person generated by the projection of an unknown 3D *performer's posture* $B'_{t'}$ onto the image plane of the camera.

The 2D input video has the following characteristics:

1. Ambiguities exist in the performer's motion captured by a single camera. The ambiguities include the depth ambiguity of the arms (Figure 2.4) and the left-right ambiguity of the legs (Figure 2.5). Depth ambiguity can lead to the same 2D view (Figure 2.4(a, b)) from two different 3D postures (Figure 2.4(a′, b′)). Left-right ambiguity can also lead to almost the same 2D view (Figure 2.5(a, b)) from two different postures. The leg contours inside the body regions in Figure 2.5 are noisy and difficult to extract accurately. As a result, it is difficult to determine with accuracy which leg is in front.

2. Self-occlusion of body parts often happens in the single video. When a body part (the left arm in Figure 2.6) is occluded, its actual pose cannot be determined from the input image. But it can be inferred from the reference motion because the performer tries to perform the same motion as the reference motion.

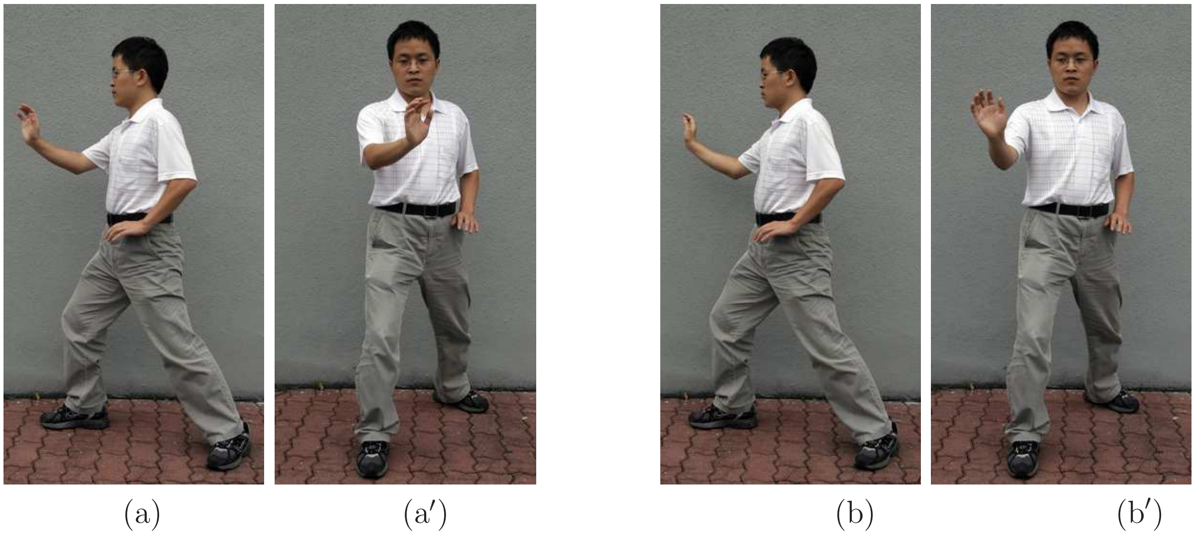


Figure 2.4: Depth ambiguity of the arm. (a) and (b) have the same 2D view although the actual 3D postures (a′, b′) are different.

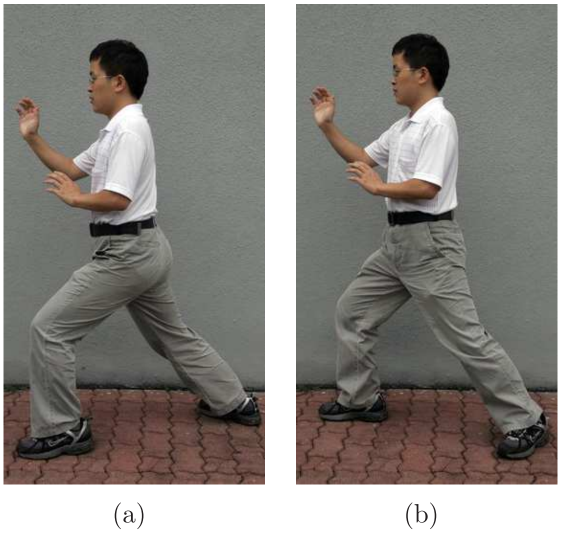


Figure 2.5: The left-right ambiguity of the legs. (a) and (b) have very similar 2D views although the actual 3D postures are different.

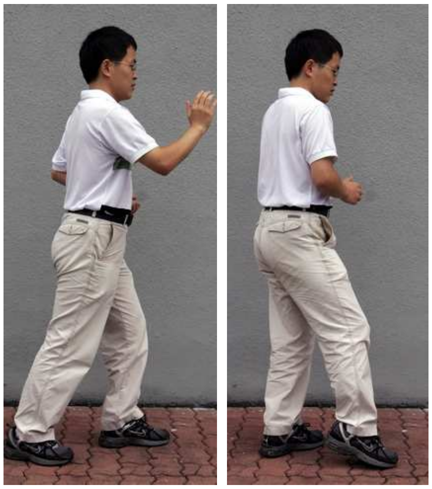


Figure 2.6: Occlusion between body parts in the video sequence. The left arm is occluded and its actual pose cannot be determined from the input images.

3. It is assumed that the input body region S' can be easily separated from the background in the images (Figure 2.7). Please refer to Section 4.1 for details.

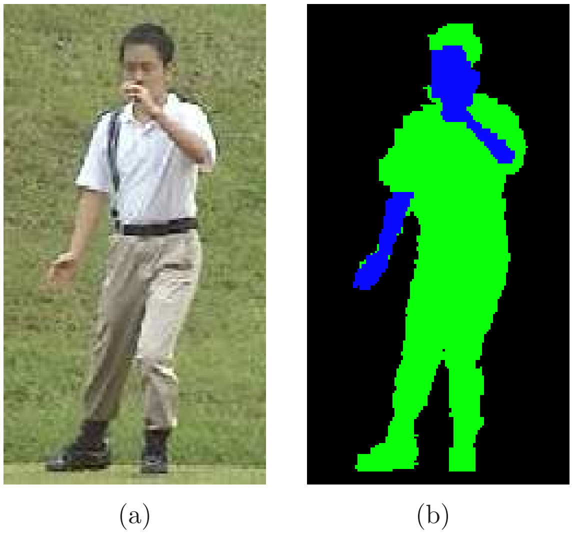
2.1.3 3D-2D Spatiotemporal Relationships

The 3D reference motion and 2D input video have the following spatiotemporal relationships:

1. Let P represent the projection function of the camera and the rendering function of the human body model. It is assumed that the camera is fixed at some location appropriate for capturing the entire motion of the performer. So, P is constant over time.

2. Let $S'_{t'}$ denote the input body region in the input image $I'_{t'}$ at time t'. $S'_{t'}$ is the projection of the unknown performer's posture $B'_{t'}$ by the camera, i.e., $S'_{t'} = P(B'_{t'})$. The human body model H is required to render the projected body region. Since it is fixed for a particular performer, H is omitted from $P(B'_{t'})$ for notational simplicity.

3. Based on currently available hardware technologies, we can assume that the 3D reference motion is sampled at a higher frame rate than the 2D input videos. For example, our Gypsy4 motion capture system captures 3D motion at 120Hz whereas



(a) (b)

Figure 2.7: Foreground extraction from the input image. Input body region (a) can be easily separated from the background in the image (b).

typical video cameras capture at 25 to 30 frames per second. So, it is necessary to establish a temporal correspondence between 2D video time t' (i.e., frame number) and 3D motion time t. Let C denote the mapping function from $\mathcal{T}'$ to $\mathcal{T}$, i.e., $C(t')$ is a particular t that corresponds to t'. We define C as a mapping function from $\mathcal{T}' = \{0, \ldots, L'\}$ to $\mathcal{T} = \{0, \ldots, L\}$ because there are fewer temporal samples in the 2D videos than the 3D reference motion.

Note that C is not a linear function because of possible differences in speed and duration of movement between the reference motion and the performer's motion. For example, compared to the reference motion, the performer in the input video may move faster or slower, or have different limb rotations at different time. In general, C should satisfy the temporal order constraint: for any two temporally ordered postures in the performer's motion, the two corresponding postures in the reference motion have the same temporal order. The performer's motion that violates the temporal order constraint contains drastic errors in the sequence of postures. Analysis of such error is outside the scope of this thesis.

4. It is assumed that the 3D reference motion and the performer's motion begin and end at the corresponding segment boundaries, i.e., $C(0) = 0$ and $C(L') = L$ are segment boundaries. The cases in which these conditions are not satisfied will be discussed in the Future Work section(Section 6.3). It is also assumed that the input video has the same number of segment boundaries as the reference motion

because the performer tries to perform the same motion as the expert. The case in which this condition is not satisfied will be discussed in Section 6.6. Note that the corresponding segment boundary in the input video may be an interval when the human body stops moving for a while at the segment boundary. This can happen to the performer when he stops at the segment boundary and temporarily forgets the subsequent motion. In this case, the postures do not change in the interval. So, a sequence of unchanged postures is indicative of a segment boundary. In this case, the interval can be reduced to a single frame such that the boundary property discussed in page 9 still holds.

5. The performer's unknown posture $B'_{t'}$ is characterized by a global rigid transformation (3D translation and rotation) T and joint articulation A. Articulation function A is a concept that is used to define the problem in Section 2.1.6. Our algorithms do not directly solve for A (refer to Section 4.6 for detail). There are three approaches for defining T and A:

   a. Define T and A with respect to a fixed and default posture B (Figure 2.1(d)): $B'_{t'} = A_{t'}(T_{t'}(B))$. In general, there is large difference between B and $B'_{t'}$. Therefore, an algorithm that tries to infer $B'_{t'}$ from B can encounter many local minima, and the algorithm will take a lot of time to converge.

   b. Define T and A with respect to previously inferred posture $B'_{t'-1}$: $B'_{t'} = A_{t'}(T_{t'}(B'_{t'-1}))$. Compared to approach (a), the difference between $B'_{t'-1}$ and $B'_{t'}$ is smaller. So, an algorithm that infers $B'_{t'}$ from $B'_{t'-1}$ will encounter fewer local minima, and the inference will take a shorter amount of time to converge. However, the inference error can accumulate over time t'.

   c. Define T and A with respect to the corresponding 3D posture $B_{C(t')}$: $B'_{t'} = A_{t'}(T_{t'}(B_{C(t')}))$. Compared to approach (a), and similar to approach (b), the difference between $B_{C(t')}$ and $B'_{t'}$ is smaller. Also, the algorithm will encounter fewer local minima when inferring $B'_{t'}$ from $B_{C(t')}$, and it will take a smaller amount of time to converge. Moreover, there is no accumulation of error over time t'.

   In comparison, approach (c) for defining T and A is more appropriate and is thus adopted. In this case, the performer's posture error between $B'_{t'}$ and $B_{C(t')}$ is captured by $A_{t'}$ and $T_{t'}$. In the ideal case without error, the global rigid transformation $T_{t'}$ and the joint articulation $A_{t'}$ are both identity functions.

Table 2.1: Independence of the posture errors in two different motion segments. All four cases are possible.

Input cases	Segment 1	Segment 2
Case 1	correct	correct
Case 2	correct	incorrect
Case 3	incorrect	correct
Case 4	incorrect	incorrect

6. When the performer's motion differs significantly from the 3D reference motion, the posture errors $\varepsilon_{t'}$ are large (refer to Section 4.4.2 for definition of posture error $\varepsilon_{t'}$). However, since the motion is smooth and continuous within a motion segment, the rate of change of posture errors should be small. That is, $\Delta\varepsilon_{t'}/\Delta t' = (\varepsilon_{t'} - \varepsilon_{t'-\Delta t'})/\Delta t'$ is small. Note that the video frame rate should be large enough (e.g., 25 fps) to acquire smooth motion in the input video.

7. The posture errors in two different motion segments are in general independent. This is because the performer can perform the motion segment correctly but makes mistake in a subsequent segment, and vice versa (Table 2.1).

2.1.4 Desired Output Characteristics

The desired outputs have the following characteristics, which describe the requirements of the problem:

1. The posture error $\varepsilon_{t'}$ between a performer's posture $B'_{t'}$ and a corresponding reference posture $B_{C(t')}$ is computed from the difference between their joint rotation angles, which include the difference between the global orientations of the two postures (see Section 4.4.2 for detail). The difference between the global positions is not included because for Taichi motion and golf swing, the difference in global positions is not important.

2. The adjustments of performer's postures required to match the corresponding reference postures should be as small as possible. This requirement matches the intuition of finding the least amount of changes necessary to adjust the performer's motion to match the reference motion during sports coaching, which is the simplest way to correct the performer's motion.

3. The segment boundaries in the performer's motion should correspond to those in the reference motion (Figure 2.8). The expert often coaches a performer segment by segment, and pays more attention to the correctness of the beginning and ending postures of each motion segment. When the performer's postures are correct at the boundaries, the postures inside the segment will be more likely correct. This observation implies that errors at the segment boundaries should carry more importance than errors at non-segment boundaries. Therefore, the performer's segment boundaries should match the reference segment boundaries. Non-segment boundaries should not be matched to segment boundaries, and vice versa.

   In the two example sequences in Figure 2.8, the person pushes the arms forward and then draw them back. The segment boundaries in both sequences lie at the time when the person starts to draw back the arms. The difference is that he pushes forward more in the top sequence compared to the bottom sequence. Since the segment boundaries in the two sequences should be matched, the temporal correspondence indicated by the solid arrows is correct because the segment boundaries are matched. In comparison, the temporal correspondence indicated by the dashed arrows is incorrect because one of the segment boundaries in the second sequence is matched to a non-segment boundary in the first sequence.

2.1.5 Summary of Basic Terms

After introducing the input and output characteristics, we summarize the basic terms used in the preceding discussions for easy reference:

Input video m' : A video of the motion of the performer who is usually a novice.

Input image $I'_{t'}$: A frame at time t' in the input video m'.

Input body region $S'_{t'}$: Segmented body region in the input image $I'_{t'}$.

Performer's motion : The motion of the performer in the input video m'.

Performer's posture $B'_{t'}$: The posture of the performer at time t'.

Performer's segment boundary : The time instances that divide the performer's motion into motion segments.

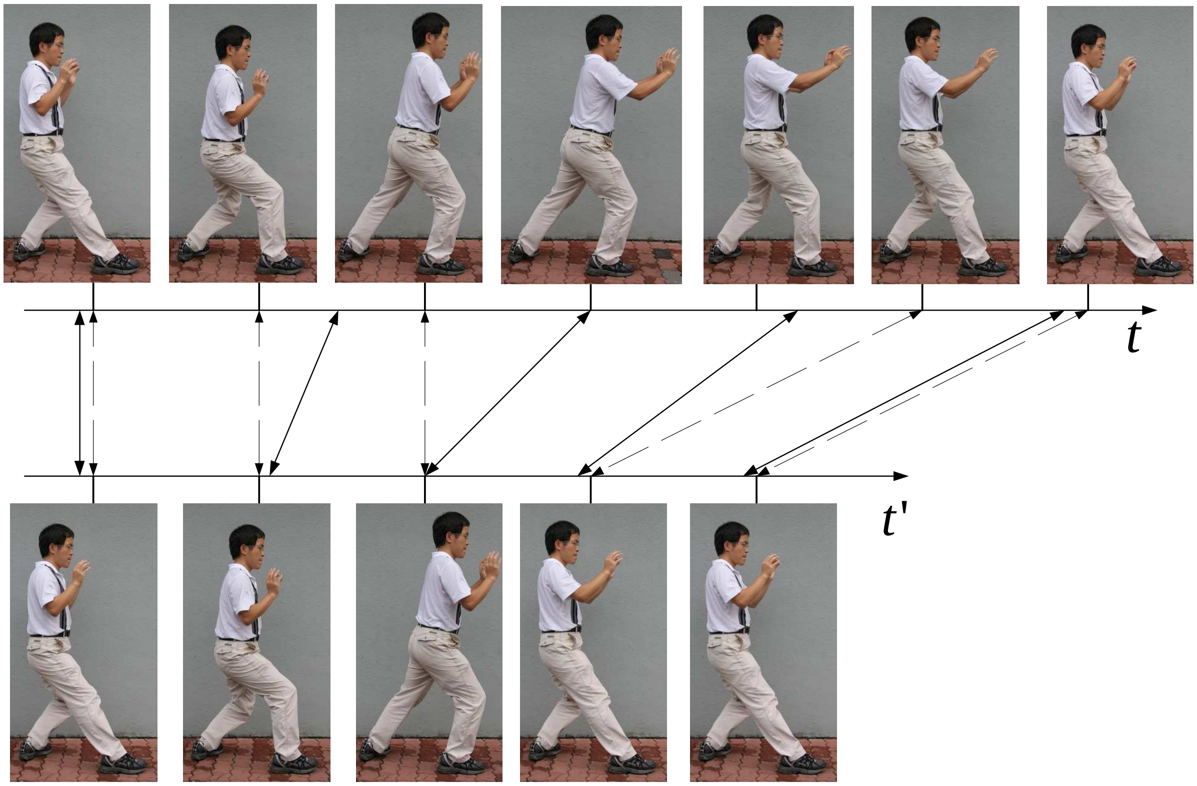


Figure 2.8: Correspondence of segment boundaries. The segment boundary (the third image) in the bottom sequence should correspond to the segment boundary (the fourth image) in the top sequence. The solid arrows represent a correct correspondence in which the segment boundaries are matched, and the dashed arrows represent an incorrect correspondence.

Performer's time t' : The time dimension of the performer's motion, which is also the time dimension of the input video m'.

Skeleton : A set of joints and bones connecting the joints that models the human body.

Reference motion M : 3D motion of the expert. The expert provides only the 3D reference motion.

Reference posture B_t : The posture of the expert at time t in the reference motion M. It consists of global position $\mathbf{p}_t$ and joint angles $\boldsymbol{\theta}_t$, where $\boldsymbol{\theta}_t$ includes the global orientation of the posture.

Reference segment boundary : The time instances that divide the reference motion M into motion segments.

Reference time t : The time dimension of the reference motion M.

Temporal correspondence $C(t')$: The correspondence between the performer's time t' and the reference time t'.

Posture error $\varepsilon_{t'}$: The difference between the estimated performer's posture $B'_{t'}$ at time t' and the corresponding reference posture $B_{C(t')}$.

The symbols next to the terms are used consistently throughout the whole thesis.

2.1.6 Problem Statement

From the discussions in Sections 2.1.1–2.1.4, we can see that the problem of computing the difference between the performer's motion and the expert's motion is by nature very complex. So it is necessary to formulate the problem clearly and precisely to capture all the complexities of the inputs and outputs.

Given the reference motion $M = \{B_t\}$ and the input video $m' = \{I'_{t'}\}$, the problem is to determine the temporal difference between the performer's motion and the reference motion, and the (spatial) posture difference between each performer's posture and its corresponding reference posture, as described in detail in Sections 2.1.1–2.1.4.

Suppose we know the performer's posture $B'_{t'}$. Then, the projection and rendering P of $B'_{t'}$ would match the input body region $S'_{t'}$ exactly, i.e.,

$$P(B'_{t'}) = S'_{t'}\,. \tag{2.1}$$

However, the performer's posture $B'_{t'}$ is unknown and must be inferred from the input body region $S'_{t'}$.

Suppose we know the correct temporal correspondence C between the reference motion and the performer's motion. If the performer does not make any posture error, then the performer's posture $B'_{t'}$ would be identical to the corresponding reference posture $B_{C(t')}$. In practice, the performer's posture can differ from the corresponding reference posture by a global transformation $T_{t'}$ and joint articulation $A_{t'}$ (as described in Section 2.1.3), i.e.,

$$A_{t'}(T_{t'}(B_{C(t')})) = B'_{t'}\,. \tag{2.2}$$

Combining Equations 2.1 and 2.2 yields

$$P(A_{t'}(T_{t'}(B_{C(t')}))) = S'_{t'} \, . \tag{2.3}$$

In practice, $P(A_{t'}(T_{t'}(B_{C(t')})))$ is not exactly equal to $S'_{t'}$ due to algorithmic error. Let us denote the difference between $P(A_{t'}(T_{t'}(B_{C(t')})))$ and $S'_{t'}$ as $d_S(P(A_{t'}(T_{t'}(B_{C(t')}))), S'_{t'})$ (see Section 4.4.1 for the definition of d_S). Then, the problem is to determine the temporal correspondence C and spatial transformations $T_{t'}$ and $A_{t'}$ that minimize the difference $d_S(P(A_{t'}(T_{t'}(B_{C(t')}))), S'_{t'})$. When the difference is minimized, the projected body region $P(A_{t'}(T_{t'}(B_{C(t')})))$ would match the input body region $S'_{t'}$ well.

Given a single camera view, multiple postures can project to the same input body region $S'_{t'}$. To recover the correct posture $B'_{t'}$, additional constraints are required. In particular, the posture error $\varepsilon_{t'} = d_B(B_{C(t')}, B'_{t'})$ (see Section 4.4.2 for the definition of d_B) between the performer's posture $B'_{t'}$ and the corresponding reference posture $B_{C(t')}$ should be minimized to capture the idea of computing the smallest adjustment required for the performer's posture to match the corresponding reference posture (as described in Section 2.1.4). Other constraints are listed below.

When both d_S and $\varepsilon_{t'}$ are minimized, $B'_{t'}$ can be recovered from Equation 2.2. Consequently, the temporal difference is captured in C and $\varepsilon_{t'}$ measures the performer's posture error.

In summary, the 3D-2D spatiotemporal motion registration problem can be formulated as an optimization problem: determine the functions P, $T_{t'}$, $A_{t'}$, and C that minimize the errors E_S and E_D

$$E_S = \frac{1}{L'} \sum_{t'} d_S(P(A_{t'}(T_{t'}(B_{C(t')}))), S'_{t'}) \tag{2.4}$$

$$E_D = \frac{1}{L'} \sum_{t'} \varepsilon_{t'} \, . \tag{2.5}$$

The minimization of E_S and E_D is subjected to the following constraints, which come from the input and output characteristics (Sections 2.1.1–2.1.4):

A. Joint angle limit. The valid angle between two connected body parts is physically limited to certain ranges (see Appendix A for details).

B. Temporal order constraint. For any t'_1 and t'_2 such that $t'_1 < t'_2$, $C(t'_1) < C(t'_2)$.

C. Similarity of corresponding segment boundaries between the reference and the performer's motion. For any segment boundary frame t', $\mathbf{v}_{C(t')} \cdot \mathbf{v}_{C(t'+1)} < \tau$ and $\mathbf{v}'_{t'} \cdot \mathbf{v}'_{t'+1} < \tau$. In addition, $C(0) = 0$ and $C(L') = L$.

D. Small rate of change of posture errors. For each t', $\Delta\varepsilon_{t'}/\Delta t' = (\varepsilon_{t'} - \varepsilon_{t'-\Delta t'})/\Delta t'$ is small.

The above problem formulation correctly captures all the complexities of the inputs and the outputs. In particular, the error E_D and the constraints A–D are required to capture the characteristics of the inputs and the desired outputs.

From the problem formulation, we can see that it is a high-dimensional optimization problem with long time sequence. The degrees of freedom (DOF) of P is 4, which corresponds to camera scale, camera orientation about Z-axis, and camera position in the X-Y plane (Section 4.2). The DOF of $T_{t'}$ is 3, which corresponds to 3D global rotation. 3D global translation is omitted (Section 2.1.4). The DOF of $A_{t'}$ is 24, which corresponds to joint rotation angles of each body part (Appendix A). C is a mapping function from t' to t. These functions need to be determined over the long time sequence $t' = 0, \ldots, L'$.

2.2 Problem Decomposition

As discussed in Section 2.1.6, the proposed problem is a very complex high-dimensional optimization problem with long time sequence. It is infeasible to directly solve such a complex problem. We decompose the problem into a set of subproblems and then solve them separately.

The camera's projection function P is constant over time. So, P needs to be determined only once at the beginning of the algorithm.

The functions C, $T_{t'}$ and $A_{t'}$ are inter-dependent. From $B'_{t'} = A_{t'}(T_{t'}(B_{C(t')}))$, we can see that for any unknown performer's posture $B'_{t'}$, a different temporal correspondence C will lead to different corresponding reference posture $B_{C(t')}$, and therefore different posture difference between $B'_{t'}$ and $B_{C(t')}$ (Figure 2.9). As a result, the temporal correspondence C needs to be determined first in order to determine the posture difference.

However, the determination of C is, in turn, based on the spatial difference between performer's posture $B'_{t'}$ and reference posture B_t. The output characteristics require that

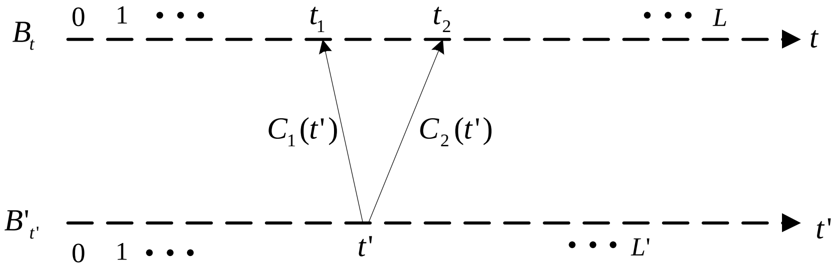


Figure 2.9: Different temporal correspondence C leads to different $T_{t'}$ and $A_{t'}$. The posture difference between $B'_{t'}$ and $B_{C_1(t')}$ differs from the difference between $B'_{t'}$ and $B_{C_2(t')}$.

the amount of adjustments of the performer's posture for matching the corresponding reference postures should be as small as possible. That means, the proper C should be determined by minimizing the error E_D (Equation 2.5). However, since $B'_{t'}$ is unknown, $\varepsilon_{t'}$ in E_D can only be estimated approximately by matching the projection of B_t with the input body region $S'_{t'}$.

The rigid-body transformation $T_{t'}$ represents the global difference between the reference and the performer's postures. A small global difference can potentially lead to a large error E_S in Equation 2.4. In contrast, the articulation $A_{t'}$ represents local difference in joint rotation angles between the reference and the performer's postures, which contributes less to E_S. Moreover, $A_{t'}$ has a much larger DOF than $T_{t'}$. So, we first approximate C and $T_{t'}$ together keeping $A_{t'}$ as an identity function.

After approximating C, $B'_{t'}$ may, in principle, be computed by determining the appropriate $A_{t'}$ and $T_{t'}$ because $B'_{t'} = A_{t'}(T_{t'}(B_{C(t')}))$. However, given a single camera, there can be self-occlusion between body parts and depth ambiguity in the input image. Therefore, there is potentially more than one possible $B'_{t'}$ that gives rise to the same input body region $S'_{t'}$. As a result, a set of posture candidates $\{B'_{t'l'}\}$ that result in small E_S are determined at each time t'. For each candidate $B'_{t'l'}$, $P(B'_{t'l'}) \approx S'_{t'}$.

After finding posture candidates $\{B'_{t'l'}\}$ for each t', the remaining problem is to determine the precise temporal correspondence C based on the best candidate postures $B'_{t'}$ and the reference postures, where the best candidate postures $B'_{t'}$ are selected from the candidate sets $\{B'_{t'l'}\}$ to minimize E_D subject to the constraints. The posture errors for each performer's posture can then be directly computed between $B'_{t'}$ and $B_{C(t')}$.

From the above analysis, the proposed problem can be decomposed into four subproblems (Figure 2.10). The first subproblem is to determine the camera projection. It is a low-dimensional (4 DOF) problem. The second subproblem is to determine the approximate temporal correspondence and global transformation. It is a low-dimensional (2D) problem with long time sequence, which can be more easily solved compared to the overall problem. The third subproblem is to estimate posture candidates for each t'. It is a high-dimensional (27 DOF) problem, but it is formulated for each image frame independently. The last subproblem is to select the best posture candidate for each input image and refine the temporal correspondence between the input sequence and the reference motion. Since the posture error between each posture candidate and each reference posture can be directly computed, this is a low-dimensional (3D) problem with long time sequence. For notational simplicity in problem formulation, input body extraction is considered as a separate problem outside the framework. Therefore, it is not included in the framework in Figure 2.10.

Note that in this framework, iteration of the last three stages is not necessary. As long as the approximate C provides good initial posture estimate that is close to the actual performer's posture, the algorithm in Stage 3 (Section 4.6) will find all the possible posture candidates that match the input body region well. Iterating the last three stages do not produce additional posture candidates.

In the following sections, we will describe the subproblems in detail.

2.2.1 Camera Calibration

The purpose of the calibration stage is to determine the camera projection P. In a sport motion sequence, it is reasonable to assume that the first performer's posture B'_0 is the same as the first reference posture B_0, such that the input body region S'_0 is the projection of B_0 by the camera. As a result, the problem is to determine P that minimizes the error E,

$$E = d_S(P(B_0), S'_0)\,, \tag{2.6}$$

where $d_S(\cdot,\cdot)$ is the image region difference measure (Section 4.4.1). We use human body, instead of a special calibration object, to calibrate the camera for ease of use. In this case, the 3D human model H is regarded as the calibration object.

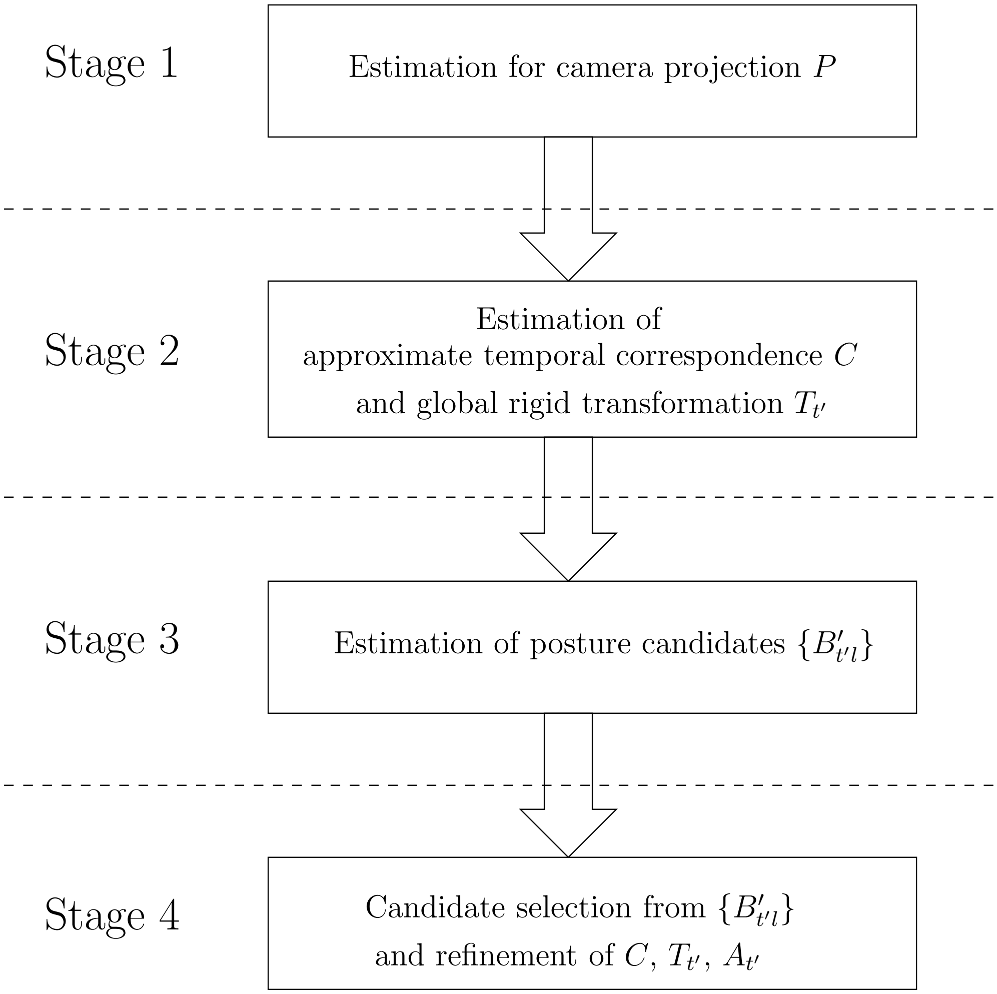

Stage 1
Estimation for camera projection P
Stage 2
Estimation of approximate temporal correspondence C and global rigid transformation $T_{t'}$
Stage 3
Estimation of posture candidates $\{B'_{t'l}\}$
Stage 4
Candidate selection from $\{B'_{t'l}\}$ and refinement of C, $T_{t'}$, $A_{t'}$

Figure 2.10: Problem decomposition. The 3D-2D spatiotemporal registration problem is decomposed into four subproblems.

2.2.2 Estimation of Temporal Correspondence and Global Transformation

The second stage computes an approximation of the temporal correspondence C and the global rigid transformation $T_{t'}$, leaving $A_{t'}$ as identity functions. Given the camera projection P, the reference motion $M = \{B_t, t = 0, \ldots, L\}$ and the sequence of input body regions $S'_{t'}, t' = 0, \ldots, L'$, the problem is to determine the approximate C and $T_{t'}$ such that each $S'_{t'}$ has a good match with the projection of its corresponding reference posture $B_{C(t')}$. That is, determine the C and $T_{t'}$ that minimize the error E_C,

$$E_C = \frac{1}{L'+1} \sum_{t'=0}^{L'} d_S(P(T_{t'}(B_{C(t')})), S'_{t'}) \,, \tag{2.7}$$

where $d_S(\cdot,\cdot)$ is the image region difference measure (Section 4.4.1). The minimization is subjected to the temporal order constraint:

B. Temporal order constraint. For any t'_1 and t'_2, such that $t'_1 < t'_2$, $C(t'_1) < C(t'_2)$.

The primary goal of this stage is to compute the approximate termporal correspondence. The more accurate is the estimation of temporal correspondence, the more efficient is the execution of the subsequent optimization stages because the search spaces of the optimization algorithms can be significantly reduced.

2.2.3 Estimation of Posture Candidates

The third stage determines a set of posture candidates $\{B'_{t'l}\}$ for each input image. Given the camera projection P, and the approximate estimation of C and $T_{t'}$, the problem is to determine possible articulations $A_{t'l}$ and rigid transformations $T_{t'l}$ of $B_{C(t')}$ so that its projection matches the input body region $S'_{t'}$. That is, determine the $A_{t'l}$ and $T_{t'l}$ that minimize the error $E_{t'}$ for each t',

$$E_{t'} = d_S(P(A_{t'l}(T_{t'l}(B_{C(t')}))), S'_{t'}) \,, \tag{2.8}$$

where $d_S(\cdot,\cdot)$ is the image region difference measure (Section 4.4.1). The minimization is subjected to the joint angle constraints:

A. Joint angle limit. The valid joint rotation of each body part is physically limited to possible ranges (Appendix A).

The resulting $\mathcal{B}_{t'} = \{B'_{t'l} = A_{t'l}(T_{t'l}(B_{C(t')}))\}$ are the estimated posture candidates that match $S'_{t'}$ well. When there is no posture ambiguity, $\mathcal{B}_{t'}$ has only one candidate.

2.2.4 Candidate Selection and Refinement of Estimates

The purpose of the last stage is to select the best posture candidate $B'_{t'}$ from the candidate set $\mathcal{B}_{t'} = \{B'_{t'l}\}$ for each time t', to determine the temporal correspondence C, and to compute the posture errors $\varepsilon_{t'}$. The proper C and best posture candidates $B'_{t'}$ should result in small posture errors between the performer's motion and the reference motion, and satisfy the constraints B, C, D. That is, select the best $B'_{t'}$ from $\mathcal{B}_{t'}$ and determine C that minimize $\varepsilon_{t'}$ for each t' subject to the constraints B, C, D:

$$E_D = \frac{1}{L'} \sum_{t'} \varepsilon_{t'} \, . \tag{2.9}$$

The posture errors $\varepsilon_{t'}$ computed at this stage give the errors between the performer's postures and the reference postures at each t' (Section 4.4.2).

2.3 Summary

In this chapter, we have proposed and formulated a new and fundamental problem for the analysis of long, complex human motion: 3D-2D spatiotemporal motion registration. The proposed problem is by nature very complex due to the characteristics of the inputs and the outputs. Since it is infeasible to directly solve such an extremely complex problem, a framework is proposed to decompose the problem into four subproblems.

The first subproblem is to determine the camera projection using the first reference posture and the first input image assuming that the performer's posture in the image is the same as the reference posture. This is a low-dimensional (4 DOF) problem and the camera projection is determined only once for all the input images.

The second subproblem is to determine the approximate temporal correspondence between the 3D reference motion and the performer's motion in the single video. This is a low-dimensional (2D) problem with long time sequence, which can be more easily solved compared to the overall problem.

The third subproblem is to estimate posture candidates for each input image. Given a single camera, there can be occlusions between body parts and depth ambiguity in the input image. Therefore, there are potentially multiple posture candidates that match the input body region in the image. As a result, a set of posture candidates are estimated for each input image. Posture candidate estimation is a high-dimensional (27 DOF) problem, but it is formulated for each image frame independently.

The last subproblem is to select the best posture candidate for each input image and refine the temporal correspondence between the input sequence and the reference motion. Since the posture error between each posture candidate and each reference posture can be directly computed, this is a low-dimensional (3D) problem with a long time sequence. It can be further decomposed into several short sequence problems using the segment boundary property, which will be be discussed in Section 4.7. Once posture candidate selection and temporal correspondence are determined, the posture error for each performer's posture can then be directly computed between the selected posture candidate and the corresponding reference posture.

Chapter 3

Related Work

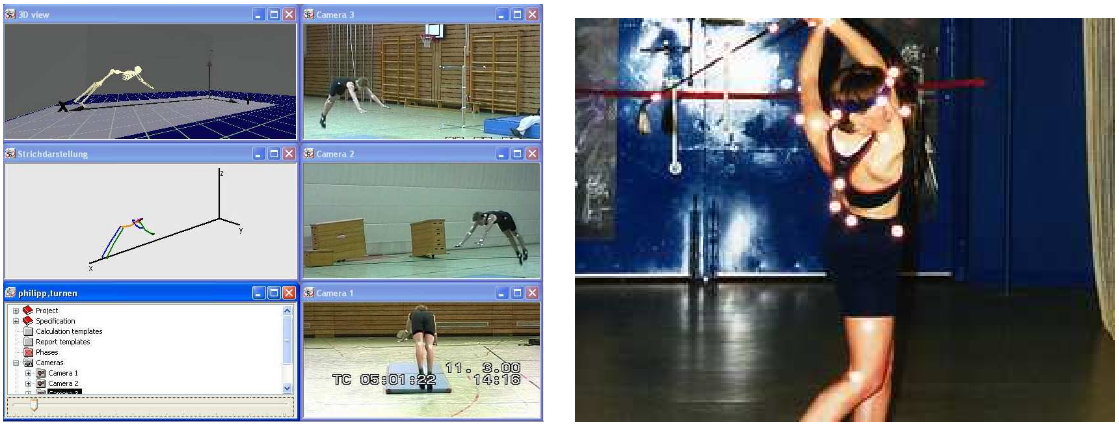
Some commercial products have been developed for sports and training (Section 3.1). However, commercially available systems for sports coaching are either not affordable to general users, or do not perform detailed motion analysis automatically. To our best knowledge, no research has been done on spatiotemporal registration between 3D reference motion and a single 2D video. Nevertheless, several research topics have close relationships with the proposed problem, including human body tracking (Section 3.2), human body posture estimation (Section 3.3), and video sequence alignment (Section 3.6). Since posture estimation is often performed in each frame during human body tracking, existing works that combines human body tracking and posture estimation is also discussed (Section 3.5). Refer to the survey papers [MG01, MHK06] for more detail about human body tracking and posture estimation.

3.1 Commercial Sports Training Systems

There are two kinds of commercial systems for sports training: 3D motion-based system and 2D video-based system.

3.1.1 3D Motion-based Systems

3D motion-based systems use a commercial motion capture (MOCAP) system to capture the performer's 3D motion [Vic, Sima] (Figure 3.1). Then, the performer's 3D motion is analyzed by the coach with the help of the systems' software. The MOCAP system



(a) (b)

Figure 3.1: 3D motion-based sports training system. (a) Simi 3D motion capture and analysis system (from http://www.simi.com). (b) Vicon motion capture system (from http://www.vicon.com/applications/sports.html).

typically uses multiple cameras to track the motion of a number of reflective markers attached to the performer's body (3.1(b)). The markers' 3D positions are recovered and used to compute the performer's 3D motion, which includes the temporal sequence of 3D positions and orientations of the performer's body parts. From the performer's motion, many characteristics can be directly computed by the accompanying software, such as the positions, orientations, speeds and motion directions of the body parts, and the angles and distances between some body parts. The computed characteristics are then visualized using, e.g., diagrams, stick figures and virtual reality representations (Figure 3.1). Based on the computed characteristics and visualized results, a coach can quantitatively and qualitatively analyze the performer's motion to determine the parts that are performed incorrectly. He can be then provided detailed instructions to the performer for further improvement.

3D motion-based systems can provide detailed and accurate 3D information of the performer's motion, but the performer's motion may be interfered by the markers on the performer's body. More importantly, analysis of the motion is left to the coach to provide coaching instructions to the performer. These systems are not affordable (about $\$300,000$) and suitable for general users. Only professional athletes can afford to pay for the use of such a system in a constrained indoor environment.

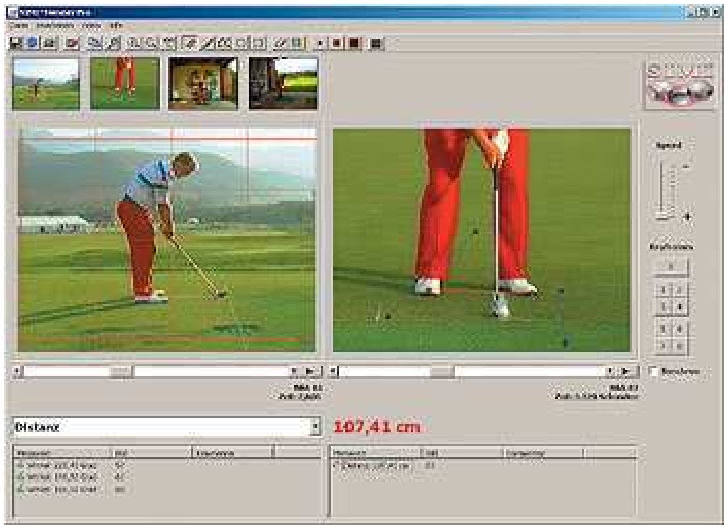


Figure 3.2: 2D video-based sports training system (from http://www.simi.com).

3.1.2 2D Video-based Systems

2D video-based systems [Pro, Mota, Motb, Simb] use single or multiple video cameras to record the performer's motion and load the video into a computer. The computer typically displays the performer's video and the pre-recorded expert's video side-by-side, and provides tools for the coach or the performer to manually analyze and compare the performer's motion with the expert's motion (Figure 1.1(b) and 3.2). The video cameras are often mounted on a bracket placed at a distance from the performer to capture the motion of the whole body. The software tools in the computer provide many options for the coach or the performer to qualitatively and quantitatively analyze the video, e.g., move the video to any specific frame for comparing with the expert's video, draw points at body joints or straight lines along some body parts in a frame image, and compare any body part's orientation with that in the expert's video by the straight lines, synchronously display the performer's video and the expert's video such that the coach can directly compare trials of some body parts, etc. 2D video-based systems allow the coach or the performer to directly view the performance of the motion and quickly assess body positions and orientations from the video.

2D video-based systems can be used in any environment, e.g., laboratories, outdoor, and even under water. So, they can be used for both indoor and outdoor sports training, e.g., golf swing, gymnastics, swimming, football, and volleyball. Moreover, the systems are much cheaper (about $3,000) than 3D motion-based systems. Therefore, they are more affordable and suitable for general users. However, 2D video-based systems cannot provide any 3D information of the performer's motion for the coach to analyze. So, the analysis is often approximate and not accurate enough compared to 3D motion-based systems.

3.2 Human Body Tracking

3.2.1 Overview

The objective of articulated human body tracking is to determine the state of the human body throughout the whole video [Bra99, DBR00, IB96, aF98, SBF00a, SBS02, ST01, ST03, WN99]. The state of human body often contains position and orientation of each body part in each video frame. Compared to the proposed problem, human body tracking does not make use of a reference motion. So there is no temporal correspondence problem.

In general, there are two main approaches to human body tracking: Kalman filtering and CONDENSATION (particle filtering). Both approaches perform the following steps:

Repeat for $t = 0, 1, 2, \dots,$

1. Predict the state of the current frame t using motion model and the estimated state of the previous frame.

2. Transform the human model according to the current state.

3. Measure the difference between the human body model and the image in the current frame. Edge, silhouette, and intensity are often used as image features for the measurement.

4. Update the current state based on the measurement.

Figure 3.3 illustrates the schema for human body tracking. Human body posture at the current frame (the second point on the time axis) is determined given the posture estimate at the previous frame (the first point). The schema is used to compare the proposed problem later in Figure 3.7 (Section 3.6).

Figure 3.3: Schematic diagram for human body tracking. Human body tracking is to determine the body's posture over time (arrow) given estimation of its state at the previous time.

3.2.2 Kalman Filtering

Kalman filter [Bro98] represents the state of human body by a mean state vector and a covariance matrix [WN99]. It uses a motion model to predict the state for the current frame from the state in the previous frame. The motion model is often a low-order dynamics, e.g., the second-order dynamics that describe position and velocity. Based on the difference between the extracted features in the current frame and those in the transformed model, the state of the human body is updated.

As an example, Wachter and Nagel [WN99] used an extended Kalman filter for human tracking in a single video. They used a motion model of constant velocity in the prediction step, and used edge and region information for the measurement. They showed that simple motion (e.g., walking) parallel to the camera image plane can be tracked from a single video.

3.2.3 CONDENSATION

CONDENSATION [IB96] represents probability distribution of the state of human body by a set of weighted samples. In the first step, it generates samples to represent the predicted probability distribution in the current frame using a motion model and the probability distribution in the previous frame. The differences between the image features and the samples are computed. Then, the weights of the samples are updated according to the measured differences. After normalizing the weights of the samples, the set of weighted samples represent the updated probability distribution for the current frame.

In CONDENSATION, the number of samples required to represent the probability distribution increases exponentially with the dimensionality of the problem [FP03]. Since articulated body often has a high degree of freedom, strong prior models or efficient sample generation techniques are often used to reduce the required number of samples in articulated body tracking [SBF00a, DBR00, ST01]. For the use of strong prior model, Brand [Bra99] used HMM model for 3D motion recovery. Sidenbladh et al. [SBF00a] used a nonlinear temporal model to constrain the state space of human body. Leventon et al. [aF98] used example-based motion models to constrain the state space. For efficient sample generation techniques, Cham et al. [CR99] represented only the peaks of the probability distribution. Since the number of peaks is often small, only a few samples are required to be generated. In the work of Sminchisescu et al. [ST01, ST03], the uncertainty of probability distribution is estimated. More samples are generated at the state space with higher uncertainty such that a relatively small number of samples can be used to approximate the probability distribution.

3.2.4 Summary

In Kalman filter, a single state is predicted for the current frame, and the measurement is computed for the single state. So the computation time is often small compared to CONDENSATION. However, since Kalman filter represents a unimodal distribution of the state of human body, it cannot be used to track human motion when the probability distribution is multi-modal, which often happens in articulated body tracking due to depth ambiguity, self-occlusion, and cluttered background in the video.

In comparison, the set of weighted samples used in CONDENSATION can represent arbitrary multi-modal probability distributions. But the number of samples required to represent a distribution often increases exponentially with the dimensionality of the problem. Strong motion model or efficient sample generation techniques have to be used when applying CONDENSATION to human body tracking.

3.3 Human Body Posture Estimation

Human body posture estimation is to estimate the 2D or 3D body posture from single or multiple images (see Figure 3.4 for schematic illustration). There are two approaches for posture estimation: model-free approach [AT04, AS00, AS00, AS03, AASK04, Bra99, EL04, FL95, HS03, HLF99, How04, IF99, LYST06, MOB05, MM02, Mor05, RFZ05, RBM05, RST02, RMR04, RAS01, RS00a, RS06, SVD03, TNS$^+$06, UFHF05, UFF06] and model-based approach [BM98, CR99, DCR01, FH05, HW04, Isa03, JBY96, LC04, RFZ05, RK95, ST01, SIFW03, SMFW04, WL05].

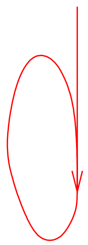


Figure 3.4: Schematic diagram for human posture estimation. It determines the body posture at a single time based on the input images. There is often an iteration process (arrow) that require estimation until convergence.

3.3.1 Model-free Approach

This approach does not use human body model. It includes three main kinds of methods: mapping function-based methods, exemplar-based methods, and probabilistic assemblies of parts.

3.3.1.1 Mapping Function-based Methods

These methods [AT04, EL04, LYST06, RAS01, RS00a, RS06, TNS$^+$06, UFHF05, UFF06] learn a nonlinear mapping function that represents the relationships between body image features and body postures. During learning, a rich set of image features (e.g., silhouette [EL04], histogram of shape context [AT04]) are extracted from each training image as the input, and the output is the known 3D posture in the corresponding training image. During posture estimation, the features in the input image are extracted and then input to the mapping function to predict the body posture.

When using these methods for body posture estimation, researchers focus on the techniques about how to learn the mapping functions. For example, Agarwal and Triggs [AT04] used 100-dimensional input vector that encodes local shapes of a human image silhouette, and 55-dimensional vector to represent 3D full-body posture. Given a set of labelled training examples, they used relevance vector machine (RVM) [Tip00] to learn a nonlinear mapping function that consists of a set of weighted basis functions. Recently, RVM has been extended to multivariate RVM for posture estimation [TNS$^+$06].

Rosales et al. [RS00a, RAS01] used input vectors that encode Hu moment of the body image output vector that encodes 22 joint angles. Given the training set, this method learned a set of forward mapping functions, each of which is a combination of sigmoidal and linear functions, using Expectation Maximization technique. Using their algorithm, a complex many-to-many mapping can be obtained which consists of the combination of learned mapping functions.

Recently, manifold method is used in posture estimation. A manifold is a topological space that is locally Euclidean. If the input image comes from a known type of 3D motion model (e.g. walking), the 3D motion model can be represented as a nonlinear manifold in a high-dimensional space. By mapping the manifold into a lower-dimensional space using embedding technique, and learning two nonlinear mappings between the embedded manifold and the visual input (i.e., silhouette) space and 3D body posture space, 3D body posture can be estimated from each input image by the two mapping functions. For example, Elgammal and Lee [EL04] used Generalized Radial Basis Function (GRBF) interpolation framework for the nonlinear mapping. Urtasun *et al.* [UFHF05, UFF06] used Gaussian Process Latent Variable Models (GPLVM) to learn the mapping. Li *et al.* [LYST06] used Locally Linear Coordination (LLC) to learn the mapping.

3.3.1.2 Exemplar-based Methods

Exemplar-based methods [AS00, AS03, AASK04, FL95, HS03, How04, MM02, SVD03] store a set of exemplar images whose corresponding 3D postures are known, and estimate the posture in the input image by searching for exemplars similar to the input image. Since multiple body postures may have very similar images, the methods often output multiple 3D body posture estimations for the input image. For example, Howe [How04] computed the Chamfer distance between the silhouettes of the input image and each exemplar and used a lookup table to select multiple posture candidates for the input image.

Since matching the image and each exemplar is often computationally expensive, researchers often save the computation time by constructing an embedding [AASK04, FL95, HS03]. Embedding technique [RS00b, TdSL00] maps a point in the image space into another low-dimensional space such that the similarity measurement between images can be efficiently computed in the embedded space. For example, Athitsos et al. [AASK04] used AdaBoost to construct an embedding, by combining a set of one-dimensional embeddings that preserves the rankings of the similarity between any input image and all the exemplars in the embedded space.

3.3.1.3 Probabilistic Assemblies of Parts

These methods [IF99, MOB05, Mor05, RFZ05, RBM05, RST02, RMR04] applies low-level feature detectors (e.g., rectangle detectors [IF99]) to detect likely body parts, and assemble them to obtain the configuration of 2D body posture that best matches the detected features. Individual body parts are detected using 2D shape [RMR04], SVM classifiers [RST02], AdaBoost [MOB05], and locally initialized appearance models [RFZ05]. For example, Mikolajczyk et al. [MSZ04] introduced a robust AdaBoost part detector to provide coarse 2D localizations of body parts in the image.

Once body part candidates are detected, body postures are assembled from the part candidates by applying prior knowledge or constraints such as joint connectivity and length ratio between parts. Mori [Mor05] used superpixels as the element to represent the input image. Based on the boundaries of superpixels and constraints between body parts, a rough 2D posture configuration was obtained. Ren et al. [RBM05] used pairwise constraints between body parts to assemble detected body parts into 2D pose configurations. These pairwise constraints include aspect ratio, scale, appearance, orientation, and connectivity. Ramanan et al. [RFZ05] learned a global body part configuration model based on conditional random fields to simultaneously detect all body parts.

3.3.1.4 Summary

The model-free approach avoids the need for explicit human body modelling. Mapping function-based methods can directly estimate body posture from single image. However, they are useful for a small set of body postures due to the complexity of the postures. Moreover, they can only recover the body postures which are similar to those in the training set.

Exemplar-based methods do not need to train a complex mapping function. But it needs to store a large amount of exemplars. Since the exemplars record a limited number of body postures, it is not possible to obtain a good posture estimation if the body posture in the input image is different from those in the exemplars.

Probabilistic assemblies of parts can estimate 2D body posture even in cluttered natural scenes from a single view. But it cannot estimate 3D body posture and the estimated 2D posture is often not very accurate.

3.3.2 Model-based Approach

Model-based approach [BM98, CR99, DCR01, FH05, HW04, Isa03, JBY96, LC04, RFZ05, RK95, ST01, SIFW03, SMFW04, WL05] estimates body posture in a analysis-by-synthesis framework as follows:

Iterate until convergence:

1. Predict a posture state.
2. Transform and project the human model according to the state to generate a synthesized image.
3. Measure the difference between the synthesized image and the real input image.
4. Update the state based on the measurement.

According to how the state is updated and predicted, model-based approach can be divided into two main classes of methods: continuous methods and probabilistic methods. Continuous methods use continuous optimization algorithms to estimate the posture. Probabilistic methods include particle filtering (CONDENSATION), belief propagation (BP) and Markov Chain Monte Carlo (MCMC). CONDENSATION has been introduced in Section 3.2.3, but here it is used to estimate posture in an iterative process for a single image frame.

3.3.2.1 Continuous Methods

Continuous methods [BM98, CR99, DCR01, JBY96, RK95, ST01] ensure the error between the synthesized image and the real input image and applies continuous optimization algorithms to determine the locally optimal solution. Many continuous optimization algorithms can be used. For example, Bregler and Malik [BM98] used Quasi-Newton method for 3D posture estimation. Ju et al. [JBY96] used gradient descent method for 2D posture estimation. Rehg and Kanade [RK95] used Levenburg-Marquardt to estimate 3D articulated posture.

3.3.2.2 Belief Propagation

Belief propagation (BP) [YFW02] estimates body posture by estimating the pose distribution of each body part [FH05, HW04, Isa03, RFZ05, SIFW03, SMFW04, WL05].

Each pose distribution is represented by a set of weighted samples similar to that in CONDENSATION.

BP generates samples that represent the current state based on the pose distributions of body parts and the joint connectivity constrains between neighboring body parts. It then transforms the body parts according to the samples and project them to generate synthesized images. Finally the weights of the samples are updated according to the measurements and this process is iterated until convergence. The weights measure the similarity between the synthesized images and the input image as well as how well neighboring body parts satisfy the joint connectivity constraint.

BP decomposes the high-dimensional human posture estimation problem into multiple low-dimensional body part pose estimation problems. Hua et al. [HW04] applied BP to estimate 2D body posture without self-occlusion. Sudderth et al. [SMFW04] used it for 3D articulated hand tracking from a single video. Wang and Leow [WL05] applied BP to estimate 3D body posture even under partial self-occlusion. Hua *et al.* [HYW05] used detected part candidates to provide better pose initialization for some body parts, which are then used to estimate postures even under partial self-occlusion.

3.3.2.3 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) [LC04] generates a sample at each iteration. Usually, the sample is a vector that represents the whole body posture. The sample is generated based on the distribution of possible poses inferred from the image features. The similarity between the sample and the image features is measured. Then, the ratio of the similarity between the current and previous iterations is computed and the sample is accepted probabilistically based on the ratio. A sample with a larger ratio has a larger probability of being accepted. This process is iterated for a large number of iterations. All the remaining samples are used to represent the probability distribution of the posture.

Note that in addition to the above model-free and model-based approaches, action recognition (e.g., [DB98, EBMM03]) can also be used to approximately estimate human postures by recognizing the categories of human actions in the input video and then directly transferring the sequence of human postures in the categorized action to the human motion in the input video.

3.3.2.4 Summary

Continuous methods can efficiently find local minima of the posture, but cannot guarantee global optimal solution especially when the initial posture estimate is far from the global optimal solution.

Belief propagation (BP) decomposes the a high-dimensional posture state into a set of low-dimensional pose states. In the lower-dimensional state spaces, a smaller number of weighted samples can be used to represent the pose distribution of a body part. The number of required samples increases *linearly* with respect to the number of body parts. Furthermore, prior constraints may be more easily represented in BP compared to other algorithms [SMFW04]. The shortcoming of BP is that it is more complex than CONDENSATION. BP method is adapted and extended in this thesis (Chapter 4).

CONDENSATION uses sampling technique to estimate body posture and therefore is more likely to find the global optimal solution of body posture. However, it estimates posture distribution in a high-dimensional posture space. So, the number of required samples increases exponentially with respect to the number of body parts.

Like CONDENSATION, MCMC needs to generate a large number of samples to represent the probability distribution of posture. Nevertheless, it is possible to decompose the posture into a set of body parts' poses, and apply CONDENSATION and MCMC to each part for estimating postures.

3.4 Combined Human Body Tracking and Posture Estimation

In practice, human posture estimation is often performed in each frame during human body tracking (Figure 3.5). In such a case, the techniques for human body tracking and posture estimation are often combined to solve the problem.

3.4.1 Examples of Combination

There are a number of methods that combine the techniques for human body tracking and posture estimation. For example, Deutscher et al. [DBR00] applied CONDENSATION for tracking and CONDENSATION for posture estimation to track 3D human motion in

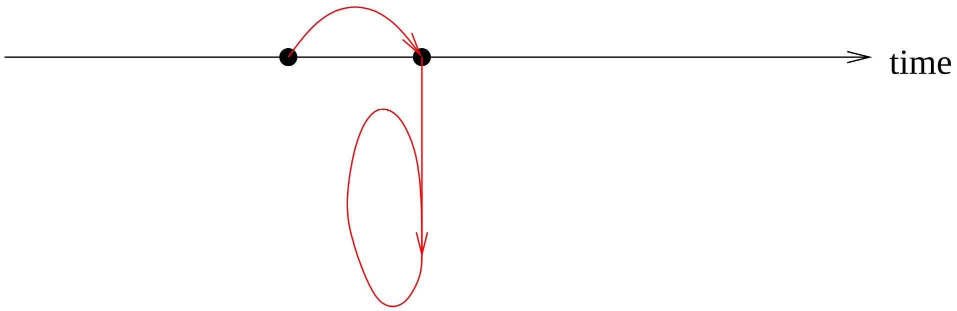


Figure 3.5: Schematic diagram for combined human body tracking (Figure 3.3) and posture estimation (Figure 3.4). Human posture estimation is often performed in each frame during human body tracking.

multiple video sequences. Hua et al. [HW04] applied CONDENSATION for tracking and BP for posture estimation to track 2D articulated body in a single sequence. Cham et al. [CR99] applied CONDENSATION for tracking and Gauss-Newton continuous optimization for posture estimation to track 2D human motion in a single video. Sminchisescu et al. [ST01] applied CONDENSATION for tracking and Levenburg-Marquardt optimization for posture estimation to track 3D human motion in a single video.

3.4.2 Learnt Motion Model

3D human body tracking from single video is challenging because multiple 3D postures can correspond to the same 2D image, and posture tracking error may accumulate over time. In order to solve such problems, strong prior motion model are often used to constrain the state space for posture estimation in each frame [HLF99, KHM00, SBF00b, SB01, SB03, SBR$^+$04, UFHF05]. The motion model is often learnt from 3D motion data captured by a commercial system. For example, Sidenbladh et al. used learnt model of walking motion [SBF00b]. Karaulova used a leant hierarchical PCA model of motion for tracking in a single video [KHM00]. Rutasun et al. used a scaled Gaussian process latent variable models (SGPLVM) [UFHF05] to learn a low-dimensional embedding of posture state space for tracking 3D golf-swing or walking from a single video.

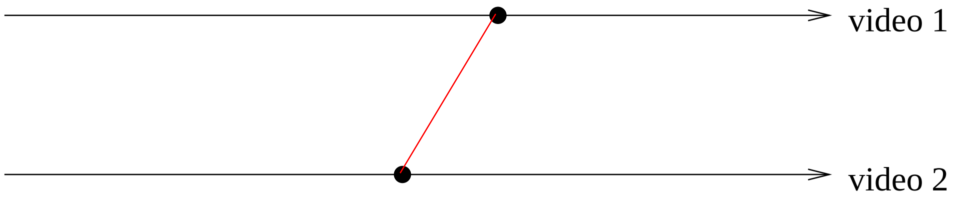


Figure 3.6: Schematic diagram for video sequence alignment. Each frame in one video (point on video 1 axis) corresponds to a frame in the other video (point on video 2 axis).

3.4.3 Summary

The techniques for human body tracking and posture estimation are often combined to solve the human body tracking problem. In order to correctly track 3D human body from single video, strong motion model is often used to constrain the posture space during posture estimation in each frame. However, the learnt motion is limited to specific motion model with relatively small variation in motion. If the motion in the video is different from the learnt motion model, the estimated postures in the frames will be biased to the postures in the training motion.

3.5 Video Sequence Alignment

Video sequence alignment [Ste98, GP99, LRS00, CI00, CI01, CI02, CSI02, RGSM03] is to establish 2D image point correspondence both in time and in space between two video sequences (see Figure 3.6 for schematic illustration). Compared to the proposed problem, video sequence alignment just needs to find a constant spatial transformation and a temporal correspondence between the images in two video sequences. It does not use 3D motion information. Therefore, it cannot analyze the detailed difference of two human motion in two video sequences.

Before sequence alignment, 2D feature points are often tracked in each video sequence. According to whether two video cameras record the same motion of the same person or the motion of two persons, two main kinds of methods are used: linear temporal correspondence and dynamic time warping.

3.5.1 Linear Temporal Correspondence

When a motion is recorded by two fixed video cameras, it is reasonable to assume that there is a linear temporal correspondence between the two video sequences, which includes two factors: the time offset between the two videos and the ratio of frame rates of the two cameras [Ste98, GP99, LRS00, CI00, CI01, CI02, CSI02]. In solving the problem, trajectory correspondence is often used instead of point correspondence [CI02, CSI02]. In this case, each motion is considered to be composed of a set of feature point trajectories. Each feature point trajectory is a trajectory of an object point representing its location in each frame along the temporal sequence. The main idea of solving video sequence alignment is as follows. First a number of pairs of possibly corresponding point trajectories are randomly selected. Then, a pair of spatial transformation and temporal correspondence for each trajectory pair is estimated. Finally, the best pair of spatial transformation and temporal correspondence is selected such that all the feature points in the two sequences are registered best.

The main difficulty in this method is to estimate spatial transformation and temporal correspondence for each pair of trajectories. One often assumes that the the ratio of frame rates is known [Ste98, CSI02] (e.g., between PAL and NTSC sequence, it is $25/30 = 5/6$). So far, existing work only estimates the time offset between two sequences. Stein's method [Ste98] exhaustively searched all possible real-valued time offsets. On the other hand, the method of Caspi et al. [CSI02] exhaustively searched all possible integer time offsets. For each time offset, they estimated spatial transformation by minimizing the difference between one trajectory and the other transformed trajectory.

3.5.2 Dynamic Time Warping

Dynamic Time Warping (DTW) is a well-known dynamic programming technique that matches a test sequence with a reference sequence if their time scales are not linearly aligned but temporal ordering constraint holds [MRR80]. It divides the matching problem into smaller subproblems, and then solves the subproblems recursively. DTW can find the optimal sequence match by recursively finding the optimal matches of the subsequences using dynamic programming technique.

Recently, Rao et al. [RGSM03] used DTW to establish temporal correspondence between two videos. The videos may not capture the same dynamic scene, but they should capture similar motion such as two individuals doing the same motion. The authors assume that the point trajectories have been found and they only need to align the two videos temporally. They used the fundamental matrix between two cameras to approximate the spatial transformation. For each pair of frames in the two sequences, the difference between the pair is computed by the registration error between the corresponding points in the frames.

3.5.3 Summary

The linear temporal correspondence method can align two video sequences that capture the same dynamic scene by different types of sensors (e.g., light and infrared) or in slightly different view points or zooms. However, they often assume that each moving object is rigid and can be viewed as one motion point. Therefore, it cannot align complex motion sequences such as 3D human motion. Moreover, since it assumes that two video sequences have a linear time offset, it may fail for videos with a dynamic time shift.

Dynamic time warping (DTW) can determine the temporal correspondence between the video sequences that may not capture the same dynamic scene. But the video sequences should capture similar motion such as two individuals doing the same motion or one person doing the same motion at two different time instances. Two dynamic programming methods (Chapter 4) which are similar to DTW are developed in this thesis.

3.6 Conclusion

Compared to the proposed problem, human body tracking and posture estimation do not make use of a reference motion. So there is no temporal correspondence problem in human tracking and posture estimation. Video sequence alignment performs temporal correspondence between two sequences. It does not use any 3D human motion information and does not perform estimation. Our proposed problem involves both temporal sequence alignment and human posture estimation (Figure 3.7).

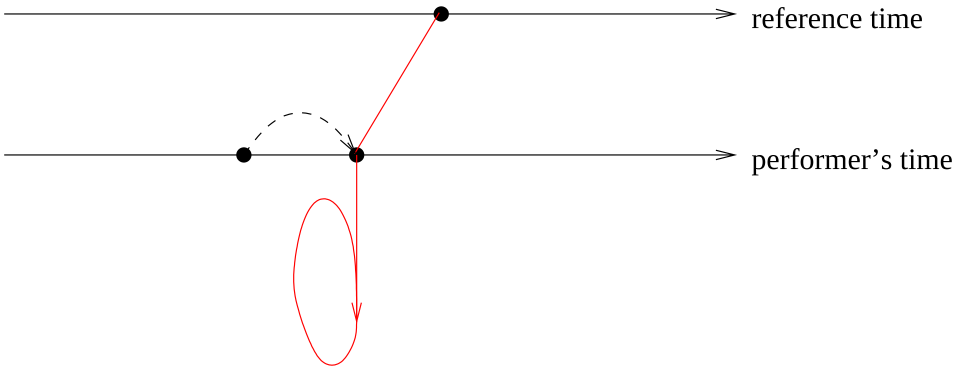


Figure 3.7: Schematic diagram for the proposed problem. The proposed problem involves both temporal sequence alignment and human posture estimation.

Chapter 4

Motion Analysis Algorithms

As discussed in Chapter 2, the motion analysis problem is formulated as a 3D-2D spatiotemporal motion registration problem. Due to the complexity of the problem, it is decomposed into four subproblems. The algorithms for solving the subproblems are described in Sections 4.2, 4.5, 4.6, and 4.7. In addition, in order to solve the subproblems, input body region $S'_{t'}$ needs to be extracted from each input image, 3D postures have to be projected and rendered to generate projected body regions, and difference measures between image regions and between 3D postures need to be defined. These algorithms are described in Sections 4.1, 4.3, and 4.4.

4.1 Extraction of Input Body Region

The input body region $S'_{t'}$ needs to be extracted from the input image $I'_{t'}$ for motion analysis. The main idea is to find the approximate position of the body region in $I'_{t'}$ by subtracting the current image $I'_{t'}$ from the next consecutive image $I'_{t'+1}$, and then use an efficient graph cut-based algorithm, i.e., GrabCut [RKB04], to accurately segment the foreground from the background in $I'_{t'}$. In addition, a skin detection algorithm is used to extract the arms from the input image $I'_{t'}$ (Figure 4.1).

4.1.1 GrabCut

GrabCut [RKB04] is an iterative image segmentation technique based on the Graph Cut algorithm [BJ01, Kol]. The GrabCut algorithm can be summarized as follows (refer to

[RKB04] for more details):

1. Place a rectangular window at the approximate position of the body region (Figure 4.1(c)). The size of the window is set at $2h/3 \times h$ where h is the height of the image. Pixels outside the rectangle are marked as known background, and pixels inside the rectangle are marked as unknown.

2. Create an initial image segmentation, where all unknown pixels are tentatively placed in the foreground class and all known background pixels are placed in the background class.

3. Create Gaussian Mixture Models (GMMs) for initial foreground and background classes.

4. Assign each pixel in the foreground class to the most likely Gaussian component in the foreground GMM. Similarly, assign each pixel in the background to the most likely background Gaussian component.

5. Update the GMMs using the assigned pixels.

6. Build a graph that consists of two types of links. N-links connect pixels in the 8-neighborhood, and the costs of the links are high in regions of low intensity gradient and low in regions of high intensity gradient. T-links connect each pixel to the foreground and background classes. The T-links describe the cost of assigning each pixel to the foreground or the background class.

7. Run graph cut to obtain two separated sub-graphs by minimizing the cost of cut links. A new tentative foreground and background classification of pixels is obtained from the graph cut result (see [RKB04] for more details).

8. Repeat steps 4-7 until the classification converges (Figure 4.1(d, e)).

GrabCut extends graph cut to color images and to incomplete initialization, i.e., only background information needs to be provided. It greatly improves the usefulness of graph cut. The inclusion of color information in the graph cut algorithm and the iterative graph cut procedure improve its robustness. GrabCut can accurately extract the input body region (Figure 4.1(e)) even if there is shadow in the image and unexpected motion in the background. When a shadow is cast into the background region, the color information in the shadow region is more similar to that in the background around the shadow region.

Such region similarity is used in GrabCut to separate the foreground from the shadow. In comparison, existing background removal methods, e.g., background subtraction by a statistical background model [SG98], often have difficulties in handling shadow and unexpected motion in the background.

4.1.2 Skin Detection

In the input image, the arms may fall inside the torso region. In this case, it would be very difficult to estimate the pose of the arms without knowing the arm region. Here, a skin detection technique [JR02] is used to detect the arms based on a statistical model of skin color. The edge and silhouette of the arms can be directly obtained once the skin region is detected. The skin detection algorithm is summarized as follows:

1. Create the Gaussian Mixture Models (GMMs) for the skin and the non-skin classes. The GMMs parameters are directly obtained from [JR02].

2. For each pixel in the input image $I'_{t'}$,

    - Compute the probability that the pixel is skin.
    - Compute the probability that the pixel is non-skin.
    - The pixel belongs to the skin class if the ratio of the skin probability and the non-skin probability is larger than a certain threshold (e.g., 0.2).

The skin detection algorithm described above will generate the initial skin region (Figure 4.1(f)) in which each pixel belongs to the skin class. However, since it is detected pixel-by-pixel, the detected skin region may not be connected and some pixels belonging to the background or other body parts may be classified into the skin class. The GrabCut algorithm is used again to extract the connected regions. Regarding the non-skin region as the background and the initial skin region as unknown, the GrabCut algorithm is applied to segment the connected skin region (Figure 4.1(g, h)) from the background in the input image. By combining foreground extraction and skin detection, the input body region $S'_{t'}$ (Figure 4.1(i)) is extracted from the input image $I'_{t'}$ (Figure 4.1(a)).

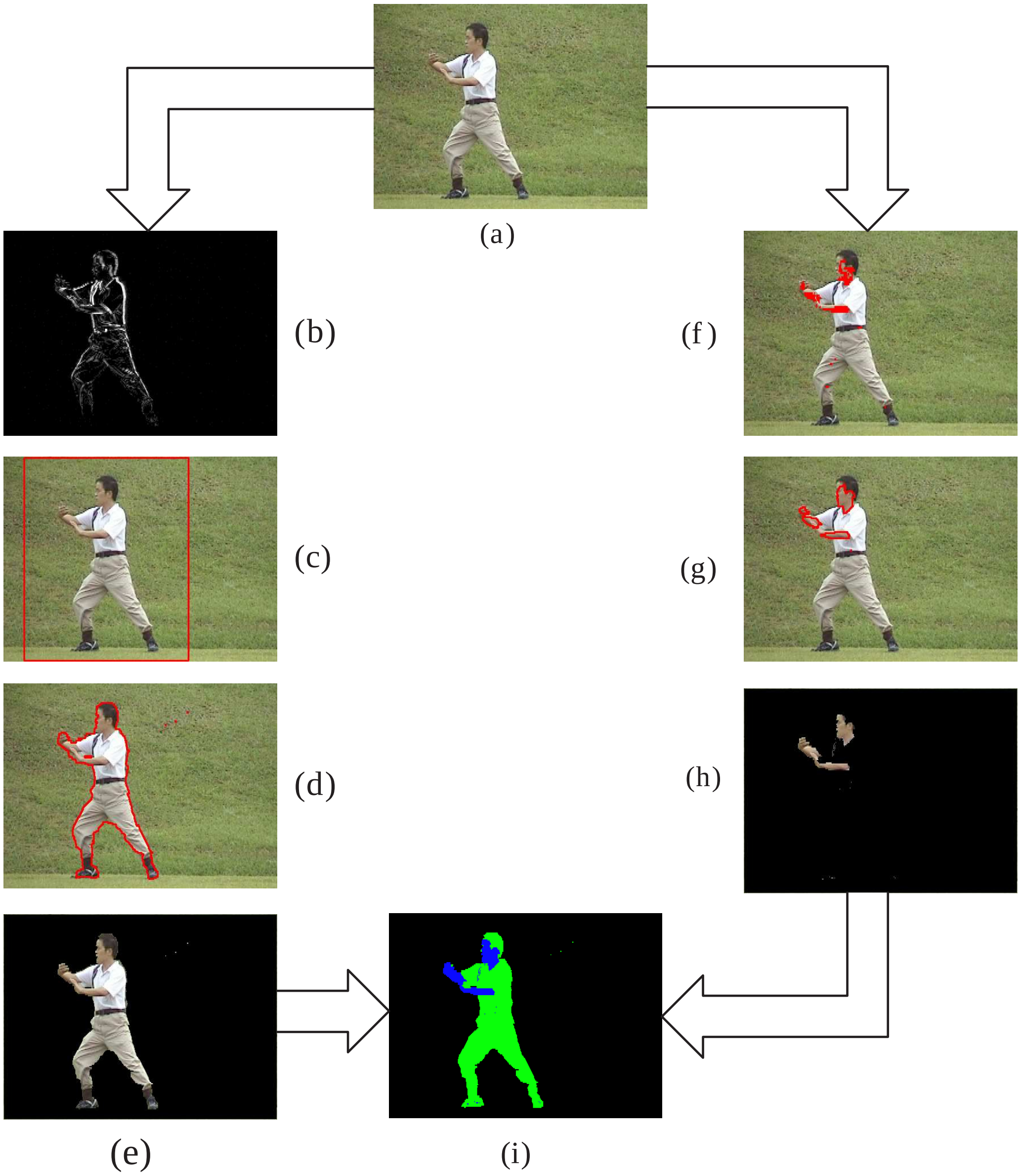


Figure 4.1: Foreground extraction. (a) Input image. (b) Difference image between two consecutive frames. (c) The initial foreground region (inside the rectangle). (d) The GrabCut segmentation result after the first iteration. The foreground boundary is represented by the curves. (e) The foreground region extracted by GrabCut. (f) The detected skin region (indicated by the red region) using GMM. (g) The GrabCut segmentation result after the first iteration based on detected skin region. (h) The skin regions extracted by GrabCut. (i) The final result of foreground extraction, which consists of the skin regions (blue) and the remaining body parts (green).

4.2 Camera Calibration

The objective of this stage (Stage 1) is to determine the camera projection P by minimizing the error E (Equation 2.6). The first image frame of the video is used as the calibration image, and the corresponding reference posture is used as the calibration posture. Here, the camera projection is assumed to be scaled orthographic because the body movement in depth is in general small compared to the distance from the body to the camera. In this case, the camera projection P contains camera scale s, camera orientation $\boldsymbol{\theta}_c = (\theta_x, \theta_y, \theta_z)^{\mathtt{T}}$, and camera position $\mathbf{c} = (c_x, c_y, c_z)^{\mathtt{T}}$ in the wold coordinate system. The camera parameters c_z, θ_x, and θ_y are omitted because of the following reasons:

1. c_z is absorbed into the camera scale s. The larger the c_z, the smaller the scale s.

2. The camera is pointing at the performer with optical axis parallel to the ground. Small change in input body region due to small rotation of θ_x and θ_y is difficult to compute accurately because the small change may be obscured by the error in foreground extraction. Also, small rotations are indistinguishable from small translations, so θ_x and θ_y are accounted for by c_x and c_y.

The remaining camera parameters are determined using the calibration posture and the calibration image as follows:

1. Set the camera parameters to default values: $\theta_z = 0$, $c_x = c_y = 0$, $s = 1$.

2. Project the calibration posture under the default camera parameters and render as a projected body region.

3. Compute the principal direction and the principal length h of the projected body region by applying principal component analysis (PCA) on the pixel positions in the projected body region. The principal direction is the first eigenvector computed by PCA, and the principal length is the maximum length of the body region along the principal direction.

4. Compute the principal direction and the principal length h' of the input body region in a similar way.

5. Compute the camera scale $s = h'/h$.

6. Compute the camera position (c_x, c_y).

- Compute the center $(p'_x,\, p'_y)$ of the input body region and the center $(p_x,\, p_y)$ of the projected body region.
- Compute the camera position as the difference between the centers, i.e., $c_x = (p_x - p'_x)/s$ and $c_y = (p'_y - p_y)/s$. The difference in sign in these two equations is due to the difference in the directions of the y-axes of the camera and world coordinate systems.

7. Compute the camera orientation θ_z about Z-axis as the angular difference between the principal directions of the input body region and the projected body region.

4.3 Projection of 3D Model

The reference posture B_t need to be projected and rendered to generate the projected regions to compare with the input body region. The projection process consists of the following steps:

1. Transform the skeleton of human model H.
   - Transform and articulate the skeleton by setting the global position of the root joint and joint rotation angles of each body joint according to the 3D posture data $\mathbf{p}_t$ and $\boldsymbol{\theta}_t$ to obtain the transformed skeleton (Figure 4.2(b)).
2. Transform the mesh of the human model accordingly to obtain the transformed 3D mesh model (Figure 4.2(d)).
3. Project the transformed 3D mesh model by a scaled orthographic projection and render the projected body region using OpenGL (Figure 4.2(e)).

4.4 Difference Measures for Motion Analysis

Two kinds of difference measures are defined in this section: difference measure between image regions and difference measure between 3D postures.

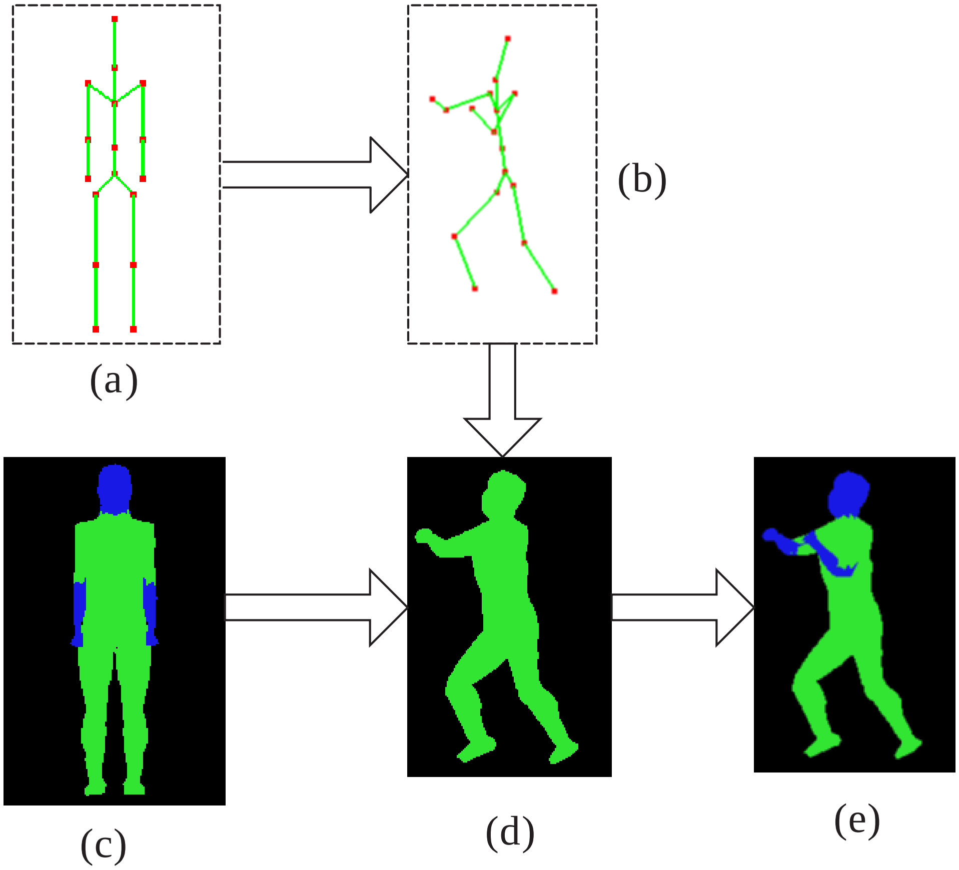


Figure 4.2: Projection of 3D model. The skeleton of the human model (a) is transformed and articulated according to the 3D posture data to obtain the transformed skeleton (b). The 3D mesh of the human model (c) is deformed accordingly to obtain the transformed 3D mesh (d), which is projected and rendered as the projected body region (e).

4.4.1 Difference Measure Between Image Regions

Edge and silhouette are used as the features for the difference measurement $d_S(S, S')$ between the image regions S and S'. The difference d_S consists of two parts: silhouette difference d_A and edge difference d_E.

Silhouette Difference d_A

Denote the set of pixels inside the silhouette of the projected body region as $\mathcal{A}$, and the set of pixels in the silhouette of the input body region as $\mathcal{A}'$. Let $\mathbf{u} = (u_x, u_y)$ denote the position of a pixel in $\mathcal{A}$ and $\mathbf{v} = (v_x, v_y)$ denote the position of a pixel in $\mathcal{A}'$. $|\mathcal{A}|$ is the size (number of pixels) of set $\mathcal{A}$. Then the silhouette difference d_A is defined as

$$d_A(\mathcal{A}, \mathcal{A}') \;=\; 1 - \frac{2|\mathcal{A} \cap \mathcal{A}'|}{|\mathcal{A}| + |\mathcal{A}'|} \tag{4.1}$$

where $\mathcal{A} \cap \mathcal{A}'$ is the amount of overlap between the two silhouettes. The larger the overlap, the smaller the d_S.

Edge Difference d_E

Denote the set of pixels on the edges of the projected body region as $\mathcal{E}$, and the set of pixels on the edges of the input body region as $\mathcal{E}'$. Let $\mathbf{u} = (u_x, u_y)$ denote a pixel in $\mathcal{E}$ and $\mathbf{v} = (v_x, v_y)$ denote a pixel in $\mathcal{E}'$. Then the edge difference d_E is defined as the chamfer distance from $\mathcal{E}$ to $\mathcal{E}'$, i.e.,

$$d_E(\mathcal{E}, \mathcal{E}') \;=\; \sum_{u \in \mathcal{E}} \left[\min_{\mathbf{v} \in \mathcal{E}'} d(\mathbf{u}, \mathbf{v}) \right] , \tag{4.2}$$

$$d(\mathbf{u}, \mathbf{v}) \;=\; \min \left(\frac{\|\mathbf{u} - \mathbf{v}\|^2}{\sigma^2}, \xi \right) , \tag{4.3}$$

where ξ is a threshold that limits the maximum difference between two pixels.

Now the difference $d_S(S, S')$ between body regions S and S' is defined as:

$$d_S(S, S') = \lambda_A d_A(\mathcal{A}, \mathcal{A}') + \lambda_E d_E(\mathcal{E}, \mathcal{E}') , \tag{4.4}$$

where λ_A and λ_E are weighting parameters.

4.4.2 Difference Measure Between 3D Postures

The difference $d_B(B_t, B'_{t'})$ between two 3D postures B_t and $B'_{t'}$ is computed from the difference $d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}'_{t'})$ between their joint rotation angles. Since the global orientation of human body is represented by the joint rotation angles of the root body part, the difference $d_{\boldsymbol{\theta}}$ includes difference in global orientations of postures. Note that the difference between the global positions is not included because for the Taichi and golf swing motion, the difference in global positions is not important.

Euclidean distance is not a good way to measure rotation angle difference because rotation angle is non-Euclidean in nature. For example, rotating an object by 360° is equivalent to rotating it by 0°. In the following, the orientations of the bones are used to measure the difference $d_{\boldsymbol{\theta}}$. The bones' orientations are computed from joint rotation angles $\boldsymbol{\theta}$ using the forward kinematic process.

Denote the 3D orientation of the j-th bone as $\mathbf{r}_{tj}$ and $\mathbf{r}'_{t'j}$ respectively for postures B_t and $B'_{t'}$. Then, the difference $d_B(B_t, B'_{t'})$, which is the same as $d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}'_{t'})$, is defined as the mean orientation difference of all the bones in the posture, i.e.,

$$d_B(B_t, B'_{t'}) = d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}'_{t'}) = \frac{1}{|\Upsilon|} \sum_{j \in \Upsilon} \cos^{-1} \left[\frac{\mathbf{r}_{tj} \cdot \mathbf{r}'_{t'j}}{\|\mathbf{r}_{tj}\| \|\mathbf{r}'_{t'j}\|} \right] \tag{4.5}$$

where Υ is the set of bones in the human body model. d_B is a general posture error for any B_t and $B'_{t'}$. In particular, $\varepsilon_{t'}$ is the posture error between the performer's posture $B'_{t'}$, which is unknown, and the corresponding reference posture $B_{C(t')}$, i.e.,

$$\varepsilon_{t'} = d_B(B_{C(t')}, B'_{t'})\,. \tag{4.6}$$

4.5 Estimation of Approximate Temporal Correspondence

The objective of this stage (Stage 2) is to determine the approximate temporal correspondence C and the global rigid transformation $T_{t'}$ for each t', given P, $B_t, t = 0, \ldots, L$, and $S'_{t'}, t' = 0, \ldots, L'$, that minimize E_C:

$$E_C = \frac{1}{L' + 1} \sum_{t'=0}^{L'} d_S(P(T_{t'}(B_{C(t')})), S'_{t'}) \tag{4.7}$$

subject to the temporal order constraint. The difference $d_S(\cdot, \cdot)$ is defined in Equation 4.4.

As discussed in Section 2.2, the main goal is to estimate the temporal correspondence C as accurate as possible given reasonable estimation of T (Figure 4.3). The computation of C is performed by applying dynamic programming (Section 4.5.2) to obtain the global optimal solution. On the other hand, the estimation of global rigid transformation does not require high precision because it will be refined in subsequent stages. Therefore, it is estimated using an efficient sampling technique instead of continuous optimization method.

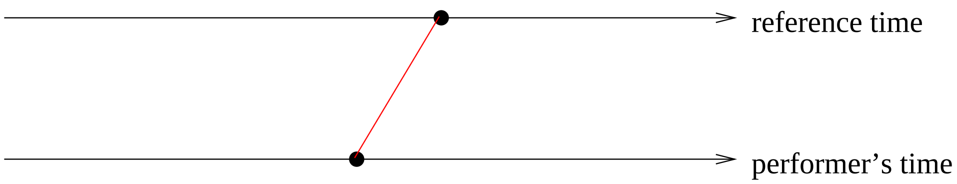


Figure 4.3: Schematic diagram for estimation of approximate temporal correspondence. A dynamic programming algorithm is developed to find the approximate temporal correspondence.

4.5.1 Estimation of Global Transformation

Given a particular C, each global transformation $T_{t'}$ can be determined by finding the best match between input body region $S'_{t'}$ and projected body region $P(T_{t'}(B_{C(t')}))$:

$$T_{t'} = \arg\min_{T} d_S(P(T(B_{C(t')})), S'_{t'}) \,, \tag{4.8}$$

In our implementation, sampling method is used to determine $T_{t'}$ due to its ease of implementation. The method is described as follows:

1. Generate a set of parameter samples $\{\mathbf{s}_i \,|\, i = 1, \ldots, N\}$ by regular sampling of the rotation parameters of $T_{t'}$. Translation parameters of $T_{t'}$ are omitted as discussed in Section 4.4.2. In this thesis, we focus on the human motion with at least one foot on the ground, e.g., Taichi and golf swing. So it is impossible to rotate the human body too much in the θ_x and θ_z directions without losing balance. Therefore, the sampling range of θ_x and θ_z can be set to a very small value. Therefore, only a small number of samples are required.

2. For each sample $\mathbf{s}_i$,

   - generate $P(T_{t'}(B_{C(t')}))$ by transforming, projecting, and rendering the reference posture using the method described in Section 4.3
   - compute the difference $d_S(P(T_{t'}(B_{C(t')})), S'_{t'})$

3. The transformation parameters of the sample with the smallest difference d_S is chosen as the approximate $T_{t'}$.

4.5.2 Dynamic Programming

Let $d(t', C(t'))$ denote the difference between $S'_{t'}$ and $P(T_{t'}(B_{C(t')}))$,

$$d(t', C(t')) = d_S(P(T_{t'}(B_{C(t')})), S'_{t'})\,. \tag{4.9}$$

The task is to determine C by minimizing E_C

$$E_C = \frac{1}{L'+1} \sum_{t'=0}^{L'} d(t', C(t')) \tag{4.10}$$

subject to the temporal constraint and the boundary constraint $C(0) = 0$ and $C(L') = L$. The main idea of determining C is to recursively decompose the optimization problem into smaller subproblems, which can be solved using dynamic programming (DP). We formulate the DP problem as follows.

Let $\mathbf{D}$ denote a $(L'+1) \times (L+1)$ correspondence matrix. Each matrix element at (t', t) corresponds to the possible frame correspondence between t' and t, and the correspondence cost is $d(t', t)$. A path in $\mathbf{D}$ is a sequence of frame correspondences for $t' = 0, \ldots, L'$ such that each t' has a unique corresponding $t = C(t')$, with $C(0) = 0$ and $C(L') = L$ (Figure 4.4). If the correspondence between t' and t can be expressed as a linear function, then the optimal path is a straight line along the diagonal of $\mathbf{D}$. The cost of a path is the sum of the correspondence costs over all t', and the average cost of the path is E_C. The problem is to find the least cost path on which E_C is minimized.

The least cost path can be efficiently found by making use of the temporal order constraint. Suppose the frame pair (t', t) is on the least cost path. Then, the possible previous frame pair should be one of $(t'-1, t-1-i)$ for $i = 0, \ldots, w$. The temporal window size w is defined as kL/L' for a small $k \geq 1$. k is small because the change of posture error between the pair of corresponding frames over time is small (as described in Section 2.1.3). The least cost path from the first frame pair $(0, 0)$ to the current pair (t', t) can be determined by recursively computing the least cost path from $(0, 0)$ to one of $(t'-1, t-1-i)$, $i = 0, \ldots, w$.

Let $D(t', t)$ denote the least cost from the frame pair $(0, 0)$ up to (t', t) on the least cost path, and $D(0, 0) = d(0, 0)$. Then $D(L', L)$ can be recursively computed as follows:

$$D(t', t) \;=\; d(t', t) + \min_{i=0}^{w} D(t'-1, t-1-i) \tag{4.11}$$

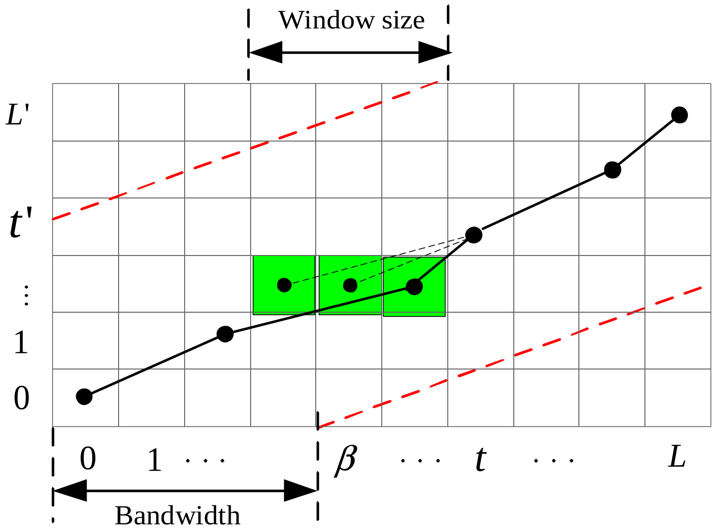


Figure 4.4: Correspondence matrix between t' and t. Each black dot denotes a correspondence between t' and a unique t, and the solid line connecting the black dots is a path in the correspondence matrix. The black dots in a small window (green elements) connected by the thin dashed lines denote the possible frame pairs preceding the pair (t', t). The thick dashed lines denote the band in which to search for possible correspondences.

Once $D(L', L)$ is computed, the least cost path is obtained by tracing back the path from $D(L', L)$ to $D(0, 0)$.

Our DP algorithm is similar to Dynamic Time Warping (DTW) [MRR80]. DTW permits one-to-many and many-to-one mappings between t' and t. In addition, two adjacent elements $(t', C(t'))$ and $(t'+1, C(t'+1))$ on the path have to satisfy $C(t'+1) - C(t') \leq 1$. On the other hand, in our DP formulation, each t' corresponds to a unique t (i.e., one-to-one mapping) and $C(t'+1) - C(t') \leq w$.

The computation complexity of DTW is $\mathrm{O}(L'L)$, and the complexity of our algorithm is $\mathrm{O}(wL'L)$. In the implementation, to improve the efficiency of the algorithm, the possible correspondence can be restricted within a narrow band (Figure 4.4, dashed lines) along the diagonal of the correspondence matrix because the change of posture error between the pair of corresponding frames over time is small. The bandwidth β of the band is defined as the horizontal distance from the straight diagonal line to the dashed line (Figure 4.4). Then the computation complexity of the algorithm is reduced from $\mathrm{O}(wL'L)$ to $\mathrm{O}(w\beta L')$.

4.6 Estimation of Posture Candidates

The objective of this stage (Stage 3) is to determine a set of posture candidates $\mathcal{B}'_{t'} = \{B'_{t'l'}\}$, where $B'_{t'l'} = A_{t'l'}(T_{t'l'}(B_{C(t')}))$ for each t' given P, C, $B_t, t = 0, \ldots, L$, and $S'_{t'}, t' = 0, \ldots, L'$, that minimize the error $E_{t'}$:

$$E_{t'} = d_S(P(A_{t'l'}(T_{t'l'}(B_{C(t')}))), S'_{t'}) \tag{4.12}$$

subject to the joint angle limits (Appendix A). This section first proposes a method to estimate the pose of each body part by a nonparametric implementation of Belief Propagation (BP) [SIFW03, Isa03, HW04, SMFW04], starting from initial pose estimates which can come from the corresponding reference posture and the estimated posture candidates in the previous frame. Then, this section develops a posture candidate estimation method to generate multiple posture candidates from the pose estimate of each body part (Figure 4.5). Compared to directly estimating posture candidates $\mathcal{B}'$ from S', estimating the pose of each body part is a lower dimensional optimization problem and therefore can be more easily solved. In particular, when the non-parametric BP is applied to solve the optimization problem, the number of lower dimensional pose samples required will be largely reduced compared to the number of high dimensional posture samples. In the following, the Belief Propagation is firstly introduced. Then, the non-parametric implementation of belief propagation and posture estimation algorithm are developed.

4.6.1 Belief Propagation

Let $p(B'|S')$ denote the probability that B' is a good posture candidate given S', i.e., $p(B'|S')$ is large when $d_S(P(B'), S')$ is small. Then, posture candidate estimation is to find B' with large $p(B'|S')$.

The human body consists of multiple body parts. Denote the pose of body part i as b_i, then $B' = \{b_i\}$. Obviously, the probability $p(B'|S')$ that the human body adopts posture B' is related to the probability that body part i adopts pose b_i. In addition, the body parts are connected by joints, and each joint connects two body parts. Therefore, $p(B'|S')$ is also related to the joint probability that body parts i adopts pose b_i and body part j adopts pose b_j. Based on the pairwise Markov Random Field formulation, $p(B'|S')$

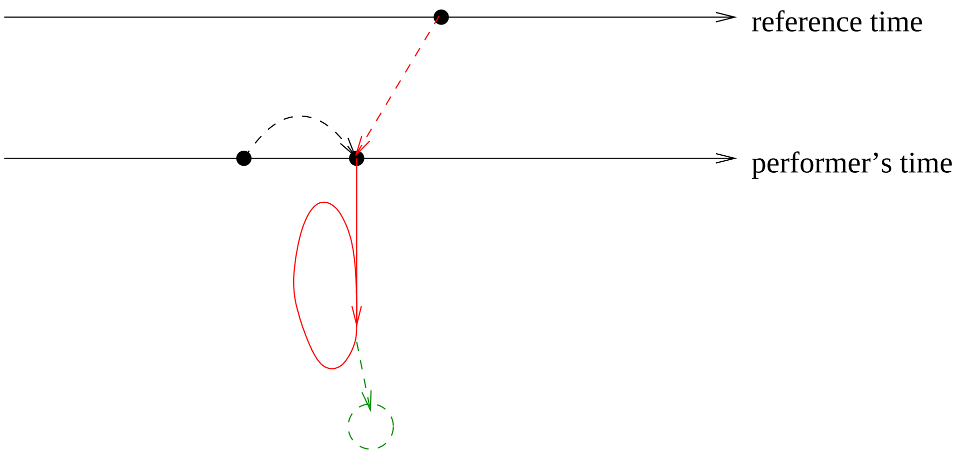


Figure 4.5: Schematic diagram for posture candidate estimation. The algorithm uses the correspondence reference posture and previous posture (dashed dark arrows) as the initial estimates to iteratively determine a set of posture candidates (dashed circle) that match the input image.

can be factorized as follows [Jor02]:

$$p(B'|S') \propto \prod_{i} \phi(b_i, S') \prod_{i,j} \psi(b_i, b_j) \tag{4.13}$$

where $j \in \Gamma(i)$ and $\Gamma(i)$ is the set of body parts connected to body part i. The function $\phi(b_i, S')$ is related to the probability $p(b_i|S')$ that body part i adopts pose b_i given S'. It measures the similarity between the projection of b_i and the input body region S'. The function $\psi(b_i, b_j)$ is related to the joint probability $p(b_i, b_j)$. $\psi(b_i, b_j)$ enforces the joint constraints between body parts i and j. When body part i adopts pose b_i and body part j adopts pose b_j, the two parts should be connected at the joint and the joint angle between the body parts should be within physical limit. When these conditions are satisfied, $\psi(b_i, b_j)$ is large; otherwise, it is small.

Instead of computing $p(B'|S')$ directly, BP algorithm computes $p(b_i|S')$ for each body part i. $p(b_i|S')$ is related to the similarity function $\phi(b_i, S')$ and is called the *belief* of body part i, i.e., the belief that body part i adopts pose b_i given S'. Compared to estimating posture B' from S', estimating the pose of each body part is a lower dimensional optimization problem and therefore can be more easily solved. In particular, when the non-parametric BP is applied to solve the optimization problem, the number of lower dimensional pose samples required will be largely reduced compared to the number of high dimensional posture (represented by joint rotation angles) samples. Therefore, a

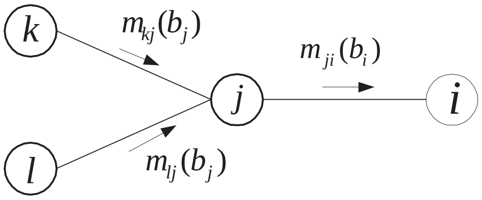


Figure 4.6: The contributions from connected body parts. Belief $p(b_i|S')$ is related to the contribution $m_{ji}(b_i)$ of body part j connected to body part i. $m_{ji}(b_i)$ is related to the contributions of the other body parts (i.e., k and l) that are connected to body part j.

set of poses $\{b_i\}$ instead of joint rotation angles are used to represent the posture B'.

Body part i is connected to other body parts, say body part j, at the joints. So, the poses of the connected body parts will affect the possible pose of body part i. Thus, the belief $p(b_i|S')$ is also related to the pose of the connected body parts. Denote the *contribution* of body part j to the pose b_i of body part i as $m_{ji}(b_i)$ (Figure 4.6). Then,

$$p(b_i|S') \propto \phi(b_i, S') \prod_{j \in \Gamma(i)} m_{ji}(b_i)\,. \tag{4.14}$$

The contribution $m_{ji}(b_i)$ is related to the pose b_j of body part j and the joint constraint between body part j and i. It is also related, in turn, to the contributions of the other body parts that are connected to body part j, except body part i because $m_{ji}(b_i)$ is directed from body part j to body part i (Figure 4.6). Thus, $m_{ji}(b_i)$ can be written as [Jor02]:

$$m_{ji}(b_i) \propto \int \phi(b_j, S')\ \psi(b_j, b_i) \prod_{k \in \Gamma(j) \backslash i} m_{kj}(b_j)\ db_j \tag{4.15}$$

Note that Equation 4.15 is integrated over all poses b_j of body part j to obtain the unconditioned contribution of body part j. Without the integration, the contribution of body part j would be conditioned on a specific pose b_j of body part j.

In the belief propagation algorithm, every contribution $m_{ji}(b_i)$ is updated according to formula 4.15 and every belief $p(b_i|S')$ is updated according to formula 4.14. The contributions $m_{ji}(b_i)$ are not accurate at the beginning because (1) each b_i comes from the initial posture estimate that may be very different from the actual posture, and (2) each $\phi(b_j, S')$ may not accurately measure the similarity between the projection of b_j and the S' because of self-occlusion and depth ambiguity in the input image. Similarly, the beliefs are also not accurate at the beginning because every contribution is updated according to the approximate contributions. Therefore, an iterative process is executed to gradually improve the accuracy of the beliefs and the contributions. Note that in classical belief propagation [Jor02], $m_{ji}(b_i)$ and $p(b_i|S')$ will converge after a specific number of iterations when every contribution $m_{ji}(b_i)$ has been propagated to all the other body parts. In this thesis, we modified the classical BP through the use of constraint-hardening schedule (Section 4.6.3 and Step 1 in Section 4.6.4).

To compute the contribution $m_{ji}(b_i)$ and the belief $p(b_i|S')$, the functions $\phi(b_i, S')$ and $\psi(b_i, b_j)$ are defined in the following sections.

4.6.2 Similarity Function

Similarity function $\phi(b_i, S')$ measures the similarity between S' and the projection of body part i at pose b_i. In order to measure the similarity, each body part at pose b_i computed in the current iteration is projected and rendered together with all the other body parts $j \neq i$ whose pose b_j are obtained from the previous iteration. Then, the similarity is computed between the projected body region S and the input body region S':

$$\phi(b_i, S') = \exp\left(-d_S(S, S')\right), \tag{4.16}$$

As discussed in Section 4.4.1, the difference $d_S(S, S')$ between projected body region S and input body region S' is defined in terms of their silhouette difference $d_A(\mathcal{A}, \mathcal{A}')$ and edge difference $d_E(\mathcal{E}, \mathcal{E}')$. The silhouette difference $d_A(\mathcal{A}, \mathcal{A}')$ measures the overall difference between S and S' in terms of total amount of overlap between them. Therefore, our implementation can handle partial self-occlusion of body parts. On the other hand, the original BP [SIFW03, Isa03] measures the similarity of a projected body part with the entire input body region, without using edge information of the other body parts. Therefore, it cannot handle partial self-occlusion of body parts.

When the image of a body part (e.g., arm) falls inside the body region, the silhouette of body region cannot provide any information to help infer the pose of this body part. In such a case, the edge of the body part can be used to help search for the poses by computing the edge difference $d_E(\mathcal{E}, \mathcal{E}')$ between the two regions (Equation 4.2).

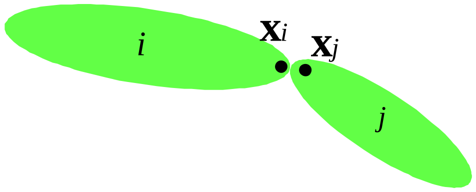


Figure 4.7: Joint constraint. The two ends of the connected body parts should be at the same 3D position, i.e., $\mathbf{x}_i = \mathbf{x}_j$.

4.6.3 Joint Constraint Function

Joint constraint function $\psi(b_i, b_j)$ enforces the constraints between two connected body parts i and j. It is defined in terms of two constraints: joint constraint and joint angle constraint.

The joint constraint states that two neighboring body parts should be connected at the joint (Figure 4.7). Let ω_i and ω_j denote the points on body parts i and j that connect to form a joint in the human body model H. Let $\mathbf{x}_i$ and $\mathbf{x}_j$ denote the 3D positions of ω_i and ω_j. When ω_i and ω_j are connected, as in the default pose, $\mathbf{x}_i = \mathbf{x}_j$. In general, when body parts i and j adopt poses b_i and b_j independently, the joint may become detached if b_i and b_j violate joint constraint. So, the separation $\|\mathbf{x}_i - \mathbf{x}_j\|$ can indicate how well the joint constraint is satisfied. The smaller the separation, the better the joint constraint is satisfied. The degree of satisfaction of joint constraint is measured by $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$, where σ is a positive parameter.

The joint angle constraint ensures that the angle α_{ij} between two connected bones in body parts i and j falls within the physical limit A_{ij} (Appendix A). The degree of satisfaction of joint angle constraint is measured by the function $J(b_i, b_j)$:

$$J(b_i, b_j) \quad = \quad \begin{cases} 1 & \text{if } \alpha_{ij} \in A_{ij} \\ a & \text{otherwise} \end{cases} \tag{4.17}$$

where $a < 1$ is a positive constant. The joint constraint function $\psi(b_i, b_j)$ combines the joint constraint and the joint angle constraint:

$$\psi(b_i, b_j) = J(b_i, b_j) \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2) \,. \tag{4.18}$$

The parameters σ and a decrease over iteration. At the first few iterations, the pose estimate of each body part may be far from the actual pose. So the constraints are loosely enforced at the first few iterations to ensure that the correct poses can be included. At latter iterations, the pose estimate of each body part is expected to become more similar to the actual pose, and therefore the constraints should become more strict.

4.6.4 Nonparametric Implementation of Belief Propagation

In practice, the evaluation of the BP integral in Equation 4.15 is often intractable with continuous state variable b_i. Several implementations of BP using nonparametric sampling approach have been proposed [SIFW03, Isa03, HW04]. In this thesis, an algorithm similar to Belief Propagation Monte Carlo [HW04] is adopted.

In the algorithm, the possible pose b_i of body part i is represented by a discrete set of samples s_{ilk} where l denotes the l-th sample of the set and k denotes the iteration number of the algorithm. The contribution $m_{ji}(b_i)$ in the k-th iteration is represented by the set $\{(s_{ilk}, \omega_{jilk})\}$ where ω_{jilk} is the weight of the contribution from body part j to body part i. The belief $p(b_i|S')$ in the k-th iteration is represented by the set $\{(s_{ilk}, \pi_{ilk})\}$ where π_{ilk} is the weight of the belief of body part i. The algorithm iteratively updates the pose of each body part to match the input body region S' in four steps:

1. Decrease parameters σ and a.
2. Generate new samples s_{ilk} according to the beliefs.
3. Compute the weights ω_{jilk} of the contributions.
4. Compute the weights π_{ilk} of the beliefs.

The details for each step is described as follows.

Step 1: Decrease parameters

σ and a are gradually decreased over iterations by $\sigma_k = \lambda\sigma_{k-1}$ and $a_k = \{a_{k-1}\}^{1/\lambda}$, where λ is a decreasing factor from 1 to 0. The parameters a is set to decrease at exponential rate of $1/\lambda$ so as to match the influence of σ which exists in the exponent of the exponential function. A larger λ (e.g., 0.95) can make the beliefs converge to the global optimal estimates with a higher probability. On the other hand, a lower λ (e.g., 0.60) can make the beliefs converge faster, but the beliefs may converge to local optimal estimates with a higher probability. This step is used to harden the joint constraint

(Section 4.6.3) to make the beliefs and contributions gradually converge over iterations. Note that there is no such constraint-hardening schedule in [HW04].

Step 2: Generate new samples

Generate samples s_{ilk} for body part i according to its belief in iteration k and the beliefs of its connected body parts in iteration $k-1$ (Figure 4.8(c)). The sampling technique consists of three steps:

1. Sample from the nonparametric distributions of belief of body part i and the beliefs of the connected body parts in iteration $k-1$.
2. For each selected sample, construct a Gaussian function to generate a sample for body part i.
3. Draw a sample from each Gaussian function randomly.

Step 3: Compute the weights of contribution

For each new sample s_{ilk}, compute the weight ω_{jilk} by the nonparametric version of Equation 4.15:

$$\omega_{jilk} = \sum_{l'=1} \phi(s_{jl',k-1}, S') \ \psi(s_{jl',k-1}, s_{ilk}) \prod_{h \in \Gamma(j) \setminus i} \omega_{hjl',k-1} \,. \tag{4.19}$$

Step 4: Compute the weights of belief

For each new sample s_{ilk}, compute the weight π_{ilk} by the nonparametric version of Equation 4.14 (Figure 4.8(d-f)):

$$\pi_{ilk} = \phi(s_{ilk}, S') \prod_{j \in \Gamma(i)} \omega_{jilk} \,. \tag{4.20}$$

The weights π_{ilk} are then normalized such that the sum of them for each body part i is 1:

$$\sum_{l} \pi_{ilk} = 1 \,. \tag{4.21}$$

The iteration process (Figure 4.8(g, h)) stops when $p(b_i|S')$ for all i converge or after a fixed number of iterations. The samples with larger weights are the pose estimates of body part i.

In the algorithm, most time is spent on computing the similarity functions. Suppose the number of body parts is N_i, the number of samples for each body part is N_l, and the number of iterations is N_k. Then, the computation complexity of the algorithm is $O(N_i N_l N_k)$.

Figure 4.9 illustrates sample results of the algorithm. Figure 4.9(a) is the input image and Figure 4.9(b) is the extracted input body region. Given the initial posture (Figure 4.9(c)), the pose samples of each body part are more close to the initial posture instead of the performer's posture after the first BP iteration (Figure 4.9(d)). However, the pose samples converge more and more to the performer's posture with respect to the BP iterations (Figure 4.9(e, f)). It shows that after a small number of iterations, the pose of each body part converges. The convergence is guaranteed by the constraint-hardening schedule of Step 1 in the algorithm, which has been shown by extensive experimental results (Section 5.2).

Note that if a body part is totally occluded, the belief of the part will be influenced by several factors: the beliefs of connected body parts, the corresponding reference posture, and the posture candidate estimated in the previous frame. If the neighboring body parts are close to the ground truth, the joint constraints between the neighboring parts and the current part will generate pose samples of the current body part that are close to the ground truth. Similarly, if the corresponding reference posture and the posture candidates in the previous frame are similar to the performer's posture in the current frame, the pose samples of the current body part will be close to its true belief. Otherwise, the pose estimate may be quite different from the ground truth.

4.6.5 Posture Candidate Estimation Algorithm

In principle, the set of estimated beliefs $\{b_i\}$ can represent the probability distribution of posture B' given enough pose samples for each b_i. In practice, however, a very large number of pose samples is required to capture all possible body postures. It makes the algorithm very inefficiency. Therefore, in our algorithm, only a small number of N posture candidates are generated from the pose samples of each body part and the corresponding reference posture as follows:

1. Run the nonparametric BP algorithm to generate pose samples for each body part.

2. Determine a best matching pose for each body part.

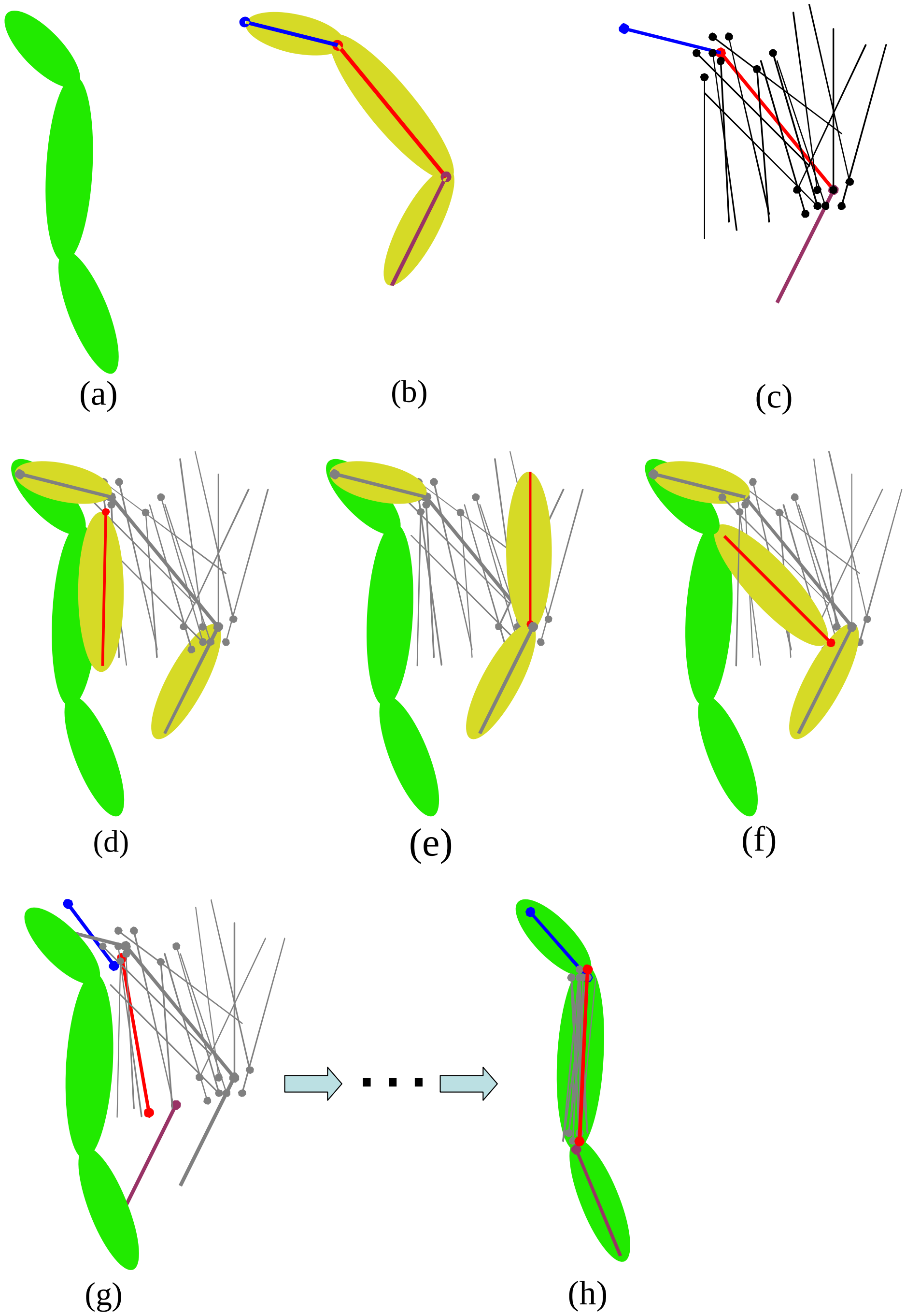


Figure 4.8: An illustrative example of nonparametric implementation of belief propagation. (a) Input body region. (b) Projected body region. (c) Generate samples for the red body part from the distributions of the body part and its connected boy parts. (d)–(f) Compute the weight of each sample: (d) high similarity and low connectivity, (e) low similarity and low connectivity, (f) medium similarity and high connectivity. (g) Updated pose of the red body part. (h) Iterate until each body part converges.

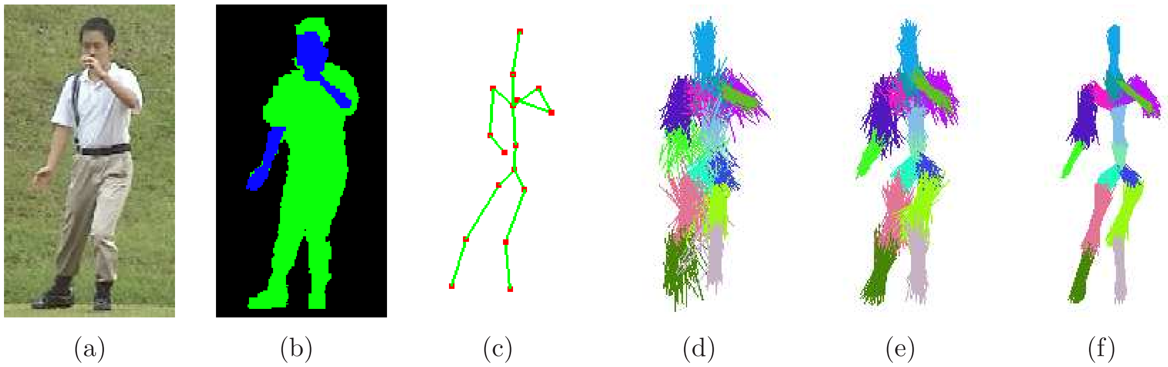


Figure 4.9: Estimation of pose candidates from a real input image. (a) Input image. (b) Input body region. (c) The initial posture. (d)–(f) The projections of the samples of each body part after the 1-th, 10-th, and 30-th iterations respectively.

- If the pose samples of each body part converge to a single state, choose any pose sample as the best pose for this body part.
- If the pose samples of each body part do not converge to a single state, project each body part at each pose sample to compute the mean image positions of its joints. Then starting from the root body part, generate a pose sample for each body part such that the body part at the pose sample is connected to its parent body part, and the projected image positions of its joints match the computed mean positions of its joints.

3. Generate the first posture candidate. For each body part starting from the root body part, modify the depth orientation of the best pose sample such that it has the same depth orientation as that in the corresponding reference posture. All the pose samples are combined into a posture candidate by translating the depth coordinate in each sample, if necessary, such that the neighboring body parts are connected.

4. Generate new posture candidates. Starting from the first posture candidate, flip the depth orientation of n body parts about their parent joints (Figure 4.10), starting with $n = 1$, while keeping the body parts connected at the joints.

5. The above step is repeated for $n = 1, 2, \ldots,$ until N posture candidates are generated.

Figure 4.11 illustrates sample results of posture candidate generation. All the posture candidates have the same projection in the frontal view. However, they are different from

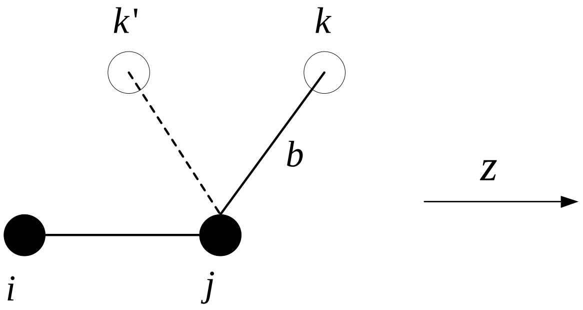


Figure 4.10: Flipping the depth orientation of body parts. The body part b is flipped in depth orientation (z-axis direction) to the new orientation (dashed line).

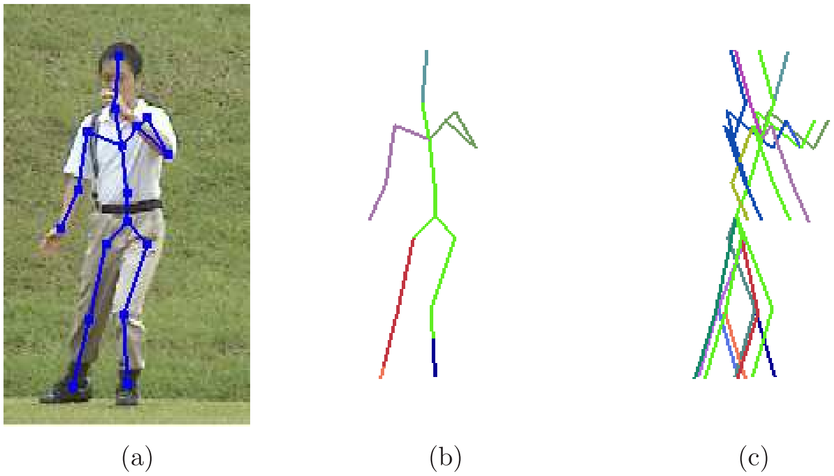


Figure 4.11: Estimation of posture candidates. (a) An input image with a posture candidate (skeleton) projected. (b) The skeleton of all posture candidates viewed from the front. Every candidate has a unique color. (c) The skeleton of all posture candidates viewed from the left side.

each other in the depth orientations of some body parts, which can be observed from the side view.

4.7 Candidate Selection and Refinement of Estimates

The objective of this stage (Stage 4) is to select the best $B'_{t'}$ from set $\mathcal{B}'_{t'}$ and determine the C that minimize the posture errors $\varepsilon_{t'}$

$$E_D = \frac{1}{L'} \sum_{t'} \varepsilon_{t'} \tag{4.22}$$

subject to the constraints B, C, and D. To satisfy the segment boundary constraint (Constraint C), $\sum_{t'} \varepsilon_{t'}$ needs to be minimized within each motion segment. Therefore, it is necessary to identify the performer's segment boundaries in the performer's motion given the reference segment boundaries in the reference motion.

4.7.1 Determination of Performer's Segment Boundaries

Given the set $\mathcal{T}_b$ of reference segment boundaries that are known in advance, the approximate temporal correspondence C obtained in stage 2 (Section 4.5), and the posture candidates $B'_{t'l'}$ at each t', the objective is to to determine the performer's segment boundaries in the performer's motion.

For each reference segment boundary $t \in \mathcal{T}_b$, the corresponding performer's segment boundary is determined by the following steps:

1. Determine initial estimate of the performer's segment boundary t' by $C(t') = t$.

2. Obtain a temporal window $[t' - \omega, t' + \omega]$, where ω is the window size.

3. Find one or more smooth sequences of posture candidates in the temporal window.

    - Correct posture candidates should change smoothly over time. Suppose $B'_{\tau l'}$ and $B'_{\tau+1,k'}$ are correct posture candidates, then $d_B(B'_{\tau l'}, B'_{\tau+1,k'})$ is small for any $\tau \in [t' - \omega, t' + \omega]$.
    - Choose a posture candidate for each $\tau \in [t' - \omega, t' + \omega]$ to obtain a sequence of posture candidates that satisfy the condition that $d_B(B'_{\tau l'}, B'_{\tau+1,k'})$ is small for each τ.

4. Find candidate segment boundaries.

- For each smooth sequence of posture candidates, find the candidate segment boundary $\tau \in [t' - \omega, t' + \omega]$ and the corresponding posture candidate at τ that satisfies the segment boundary condition (Section 2.1.6).
- Denote a candidate segment boundary found above as τ_i and the corresponding posture candidate as B'_i.

5. Identify the optimal segment boundary τ^*.

   The posture candidate at the optimal segment boundary τ^* should be the most similar to the corresponding reference posture B_t. Therefore, τ^* can be determined as follows,

$$\begin{aligned} k &= \arg\min_i d_B(B_t, B'_i)\,, \\ \tau^* &= \tau_k\,. \end{aligned} \tag{4.23}$$

4.7.2 Refinement of Estimates within Each Motion Segment

After determining the performer's segment boundaries, a posture candidate has to be selected at each t' within each motion segment to determine the optimal C and compute the posture errors. Let $[t_b,\, t_e]$ denote a reference motion segment and $[t'_b,\, t'_e]$ denote the corresponding performer's motion segment. Let $\ell(t')$ denote the index of the selected posture candidate $B'_{t'\ell(t')}$ at t' within the motion segment $[t'_b,\, t'_e]$. Then, the problem is to determine ℓ and C that minimize the error E_D (Equation 2.9) subject to the constraints B and D. Constraint D, i.e., small rate of change of posture errors, can be incorporated into the error E_D to obtain E_F:

$$E_F = \frac{1}{t'_e - t'_b + 1} \sum_{t'=t'_b}^{t'_e} \{d_c(t', C(t'), \ell(t')) + d_s(t', C(t'), C(t'-1), \ell(t'), \ell(t'-1))\}\,. \tag{4.24}$$

The difference d_c is obtained from E_D:

$$d_c(t', t, l') = d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}'_{t'l'})\,, \tag{4.25}$$

where $d_{\boldsymbol{\theta}}$ is the posture error between the posture candidate $B'_{t'l'}$ and the reference posture B_t (Section 4.4.2). The difference $d_s(t', t, s, l', k')$ measures the change of posture

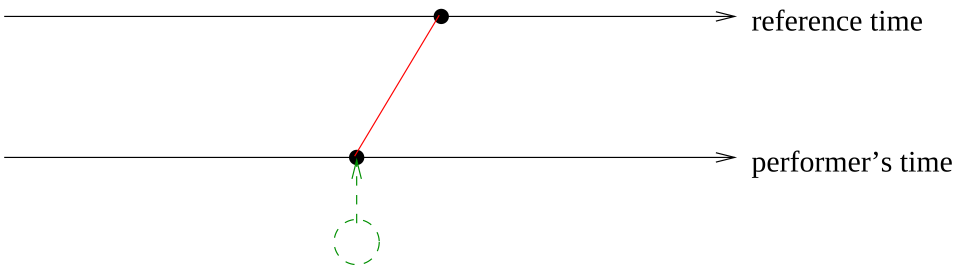


Figure 4.12: Schematic diagram for posture selection and refinement of estimates. Best matching posture candidate is selected (dashed arrows).

errors between two pairs of corresponding postures $(B'_{t'l'}, B_t)$ and $(B'_{t'-1,k'}, B_s)$:

$$d_s(t', t, s, l', k') \;=\; [d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}'_{t'l'}) - d_{\boldsymbol{\theta}}(\boldsymbol{\theta}_s, \boldsymbol{\theta}'_{t'-1,k'})]^2 \,. \tag{4.26}$$

It is minimized to satisfy constraint D.

In each motion segment, the best posture candidates $\ell(t')$ and temporal correspondence C are determined (Figure 4.12) by dynamic programming (similar to that in Section 4.5.2) as follows. Let $\mathbf{D}$ denote a correspondence matrix of size $(t'_e - t'_b) \times (t_e - t_b) \times N_B$, where N_B is the maximum number of posture candidates for $t' \in [t'_b, t'_e]$, i.e.,

$$N_B = \max_{t' \in [t'_b, t'_e]} |\mathcal{B}'_{t'}| \,. \tag{4.27}$$

Each matrix element at (t', t, l') corresponds to the possible correspondence between posture candidate $B_{t'l'}$ and reference posture B_t, and the correspondence cost consists of two terms $d_c(t', t, l')$ and $d_s(t', t, s, l', k')$. A path in $\mathbf{D}$ is a sequence of correspondences for $t' = t'_b, \ldots, t'_e$ such that each t' has a unique corresponding $t = C(t')$ and a unique corresponding $l' = \ell(t')$, and $C(t'_b) = t_b$ and $C(t'_e) = t_e$. The cost of a path is the sum of the correspondence costs over all t', and the average cost of the path is E_F. Now, the problem is to find the least cost path on which E_F is minimized.

The least cost path can be efficiently found by making use of the temporal order constraint (Constraint B). Suppose the triplet (t', t, l') is on the least cost path. Then, the possible previous triplet should be one of $(t' - 1, t - 1 - i, k')$, for $i = 0, \ldots, w$ and candidate k' at $t' - 1$. The temporal window size w is defined as kL/L' for a small $k \geq 1$. Therefore, the least cost path from any triplet (t'_b, t_b, l'_b) at the beginning segment boundary t'_b to the current triplet (t', t, l') can be determined by recursively computing the least cost path from (t'_b, t_b, l'_b) to one of $(t' - 1, t - 1 - i, k')$. Similar to Section 4.5.2, this problem is also solved efficiently using dynamic programming.

Let $D(t', t, l')$ denote the least cost from the triplet (t'_b, t_b, l'_b) up to (t', t, l') on the least cost path, and $D(t'_b, t_b, l'_b) = d_c(t'_b, t_b, l'_b)$. Then, $D(t'_e, t_e, \ell(t'_e))$ can be computed by the following recursive formulae:

$$D(t', t, \ell(t')) = \min_{l'} D(t', t, l') \tag{4.28}$$

$$\ell(t') = \arg\min_{l'} D(t', t, l') \tag{4.29}$$

where

$$D(t', t, l') \;\; = \;\; d_c(t', t, l') + \min_{i,k'}\{D(t'-1, t-1-i, k') + d_s(t', t, t-1-i, l', k')\} \tag{4.30}$$

for each $t' \in [t'_b, t'_e]$. Once $D(t'_e, t_e, \ell(t'_e))$ is computed, the least cost path is obtained by tracing back the path from $D(t'_e, t_e, \ell(t'_e))$ to $D(t'_b, t_b, \ell(t'_b))$.

The complexity of this DP algorithm can be derived in a similar manner as in Section 4.5.2. In particular, let β denote the bandwidth parameter as defined in Section 4.5.2, and L_B denote the length of a motion segment, then the complexity of the DP algorithm is $\mathrm{O}(w\beta L'_B N_B^2)$.

4.8 Summary

This chapter describes the algorithms for solving the subproblems in the four stages in detail. Table 4.1 summarizes the complexity of the algorithm in each stage. Stage 1 uses a simple algorithm for calibrating a scaled orthographic camera. Stage 2 uses dynamic programming to determine the approximate temporal correspondence C between the single video and the 3D reference motion. The DP algorithm is efficient and accurate because it can find the optimal solution in a narrow band along the diagonal of the correspondence matrix (Figure 4.4). The computation complexity of the algorithm is $\mathrm{O}(w\beta L')$, where w is the window size, β is the bandwidth, and L' is the length of the input video.

Stage 3 uses a nonparametric implementation of Belief Propagation and posture candidate generation to estimate posture candidates $B'_{t'l'}$ at each time t'. The BP algorithm decomposes the high-dimensional posture estimation problem into a set of low-dimensional

Table 4.1: Summary of the main algorithms.

Stage	Algorithm	Input	Output	Complexity
Stage 2	DP	P, M, $S'_{t'}, t' = 0, \ldots, L'$	C, $T_{t'}$	$\mathrm{O}(w\beta L')$
Stage 3	BP	P, C, $T_{t'}$, $S'_{t'}$	$\{B'_{t'l}\}$	$\mathrm{O}(N_i N_l N_k)$
Stage 4	DP	M, $\{B'_{t'l}\}$, $t' = 0, \ldots, L'$	C, $B'_{t'\ell(t')}$, $\varepsilon_{t'}$, $t' = 0, \ldots, L'$	$\mathrm{O}(w\beta L'_B N_B^2)$

pose estimation problems for the body parts. The computation complexity of the algorithm is $\mathrm{O}(N_i N_l N_k)$, where N_i, N_l, and N_k are respectively the number of body parts, pose samples for each body part, and BP iterations. After the pose of each body part is estimated by the BP algorithm, a small number of posture candidates are generated by the posture candidate generation algorithm based on the pose estimate of each body part and the corresponding reference posture. The nonparametric implementation of BP can handle partial self-occlusion of body parts because the difference measure (Section 4.4.1) measures the overall difference between the input body region and the projected body region. In comparison, the original BP measures the similarity of a projected body part with the entire input body region, without using edge information of the other body parts. Therefore, it cannot handle partial self-occlusion of body parts.

Stage 4 selects the best candidate for each t', refines temporal correspondence C, and computes the posture error $\varepsilon_{t'}$ at each t'. In this stage, the performer's segment boundaries in the performer's motion are first determined by the boundary estimation algorithm using the segment boundary property. Then, for each motion segment, an efficient dynamic programming algorithm is developed to simultaneously select the best posture candidate for each t' and determine the optimal temporal correspondence between the reference motion segment and the selected candidate sequence in the performer's motion segment. The complexity of the DP algorithm is $\mathrm{O}(w\beta L'_B N_B^2)$, where N_B is the number of posture candidates for each t', L'_B is the length of a motion segment. After selecting the posture candidate for each t' and determining the optimal temporal correspondence, the performer's posture error $\varepsilon_{t'}$ at each t' is directly computed by the difference measurement (Section 4.4.2) between the selected posture candidate and the corresponding reference posture.

Chapter 5

Experiments and Discussions

This chapter presents experiments that assess the performance of the main algorithms described in Chapter 4. The algorithms that are evaluated include estimation of approximate temporal correspondence (Stage 2, Section 5.1), estimation of posture candidates (Stage 3, Sections 5.2, 5.3), estimation of performer's segment boundaries (Stage 4, Section 5.4), and final estimation of posture errors (Stage 4, Section 5.5). All the experiments were performed in an Intel Pentium 3.0 GHz PC with 1GB RAM.

5.1 Estimation of Approximate Temporal Correspondence

5.1.1 Test Overview

The objective of this set of experiments is to evaluate the performance of the algorithm at Stage 2 for estimating the approximate temporal correspondence C. As discussed in Section 4.5, the approximate temporal correspondence C between the reference motion and the input video is determined by finding the least cost path in the correspondence matrix (Figure 4.4). The matrix element at (t', t) is the difference $d(t', t)$ that measures the difference between the input body region $S'_{t'}$ in the input video and the projected body region $P(T_{t'}(B_t))$. P is the camera projection and rendering function and $T_{t'}$ is the rigid transformation of the reference posture B_t.

The estimation of approximate temporal correspondence C is influenced by the window size $w = kL/L'$, where L and L' are the lengths of the reference motion and the input video respectively, and k is a small constant. The window size limits the number of reference frame between $C(t'-1)$ and $C(t')$, i.e., $C(t')-C(t'-1) \leq w$. In the experiment, the effect of window size on the optimality of the approximate temporal correspondence C is evaluated.

The estimation of approximate temporal correspondence C is also influenced by the bandwidth β which constrains the least cost path to lie within a narrow region along the diagonal of the correspondence matrix (Figure 4.4). The smaller the bandwidth, the more efficiency is the algorithm. In the experiments, the effects of bandwidth on the optimality of the approximate temporal correspondence C and the efficiency of the algorithm are evaluated.

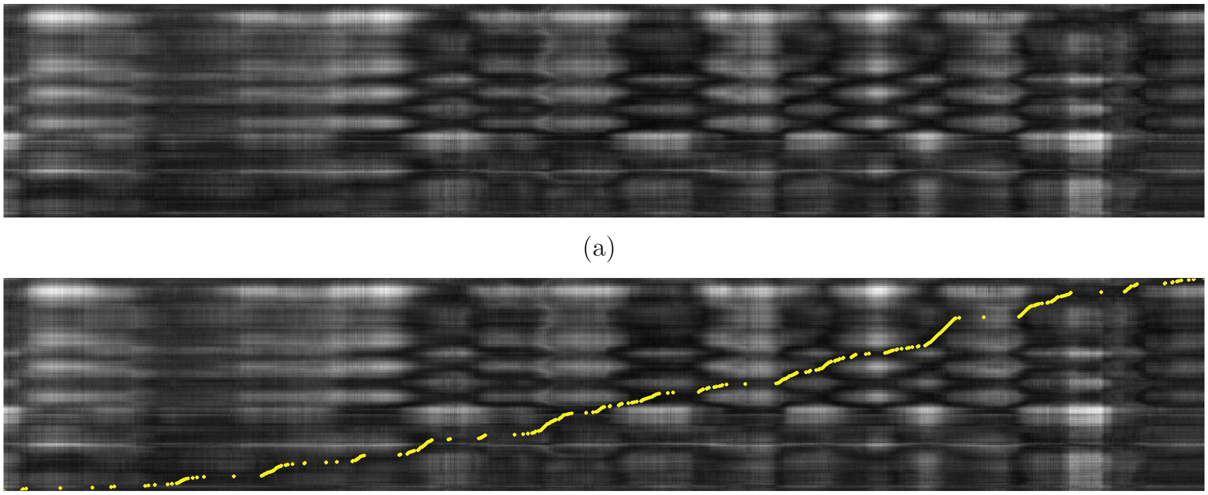
Note that the efficiency of the algorithm is also influenced by the window size. However, since the window size is much smaller compared to the bandwidth, the effect of the window size on the efficiency of the algorithm is not tested in the experiment.

For the tests, the 3D reference motion comprised a Taichi sequence of 2250 reference postures captured by a commercial motion capture system, and the input video consisted of 399 images of size 320×240. The input video was captured by the Sony DCR-TRV16E digital video recorder with frame size 720×576. A subimage including the performer's body image was cropped out from each video frame and then resized to an input image of size 320×240. In computing $d(t', t)$, the near-optimal rigid transformation $T_{t'}$ was determined by sampling method (Section 4.5.1). The number of samples was set to 10 in the tests.

The error E_C (Equation 4.10) along the least cost path in the correspondence matrix was used to measure the optimality of the approximate temporal correspondence C. Note that E_C has been normalized by the length of the input video.

5.1.2 Determination of Optimal Solution

The first test determines the optimal solution of the estimation algorithm. The experimental procedure is as follows. First, the camera calibration algorithm (Section 4.2) was run to obtain the camera parameters. Then, the optimal solution of approximate temporal correspondence C was determined by running the dynamic programming algorithm (Section 4.5.2) with the normalized bandwidth $\beta/L = 100\%$ and $w = L$. That is, the whole correspondence matrix was searched for the least cost path. In the algorithm, most



(a)

(b)

Figure 5.1: Optimal solution of approximate temporal correspondence. (a) Visualization of correspondence matrix. (b) The approximate C shown as the yellow dots along the diagonal.

time was spent in computing the correspondence cost $d(t', t)$. About 0.38 millisecond was spend in computing each $d(t', t)$.

The optimal solution of the test is shown in Figure 5.1. Figure 5.1(a) illustrates a visualization of the correspondence matrix. The intensity of pixel (t, t') represents the difference $d(t', t)$. So, a darker pixel represents a smaller difference value. Note that the pixel $(0, 0)$ is located at the bottom-left, and the original $d(t', t)$ value is scaled to intensity value between 0 and 255.

The optimal C is shown as the yellow dots along the diagonal in Figure 5.1(b). Estimation of C is performed by dynamic programming, which is guaranteed to find the least cost path in the correspondence matrix.

In the least cost path, each input frame t' corresponds to one unique reference frame t. Figure 5.2 illustrates several pairs of input images and their corresponding reference postures obtained based on the approximate temporal correspondence. We can see that the corresponding reference postures are indeed similar to the performer's postures in the input images.

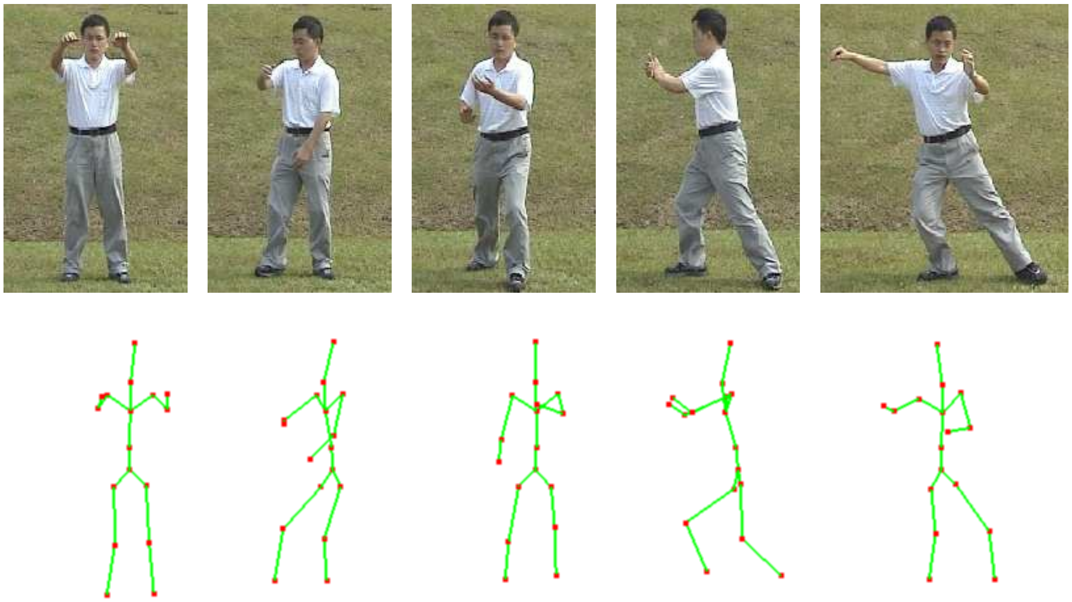


Figure 5.2: Examples of input images (first row) and their corresponding reference postures (second row).

5.1.3 Effect of Window Size

In this test, the effect of window size w on the optimality of the approximate temporal correspondence C was evaluated. The window size $w = kL/L'$ was varied by varying k from 0 to 40. The bandwidth β/L was set to 50%, which was large enough to remove the effect of bandwidth on the optimality of the approximate temporal correspondence.

Figure 5.3 illustrates the test result. From the decreasing trend of error E_C with respect to k, we can see that a small k (e.g., 10) is enough for finding the optimal C. Figure 5.4 shows that the least cost path varies according to the window size for $k \leq 7$. The least cost paths at $k = 4$ and $k = 7$ are almost the same as the optimal solution in Figure 5.1. The least cost path converges to the optimal solution for window size $\omega > 7L/L'$ (i.e., $k = 7$).

5.1.4 Effect of Bandwidth

Two tests were performed in this experiment. In the first test, the effect of bandwidth β/L on the optimality of the approximate temporal correspondence C were evaluated by varying β/L from 0% to 50%. The window size w was set at $L/2$, which is large

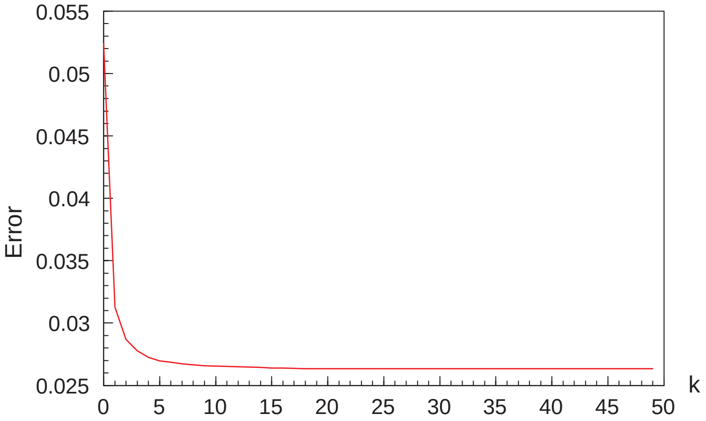


Figure 5.3: The effect of window size w on the optimality of approximate temporal correspondence.

enough to remove the effect of window size on the optimality of the approximate temporal correspondence.

In the second test, the effect of bandwidth β/L on the computation time for determining the approximate temporal correspondence C was evaluated by varying β/L from 0% to 100%. For each β/L, the algorithm was run 30 times and the computation time was obtained by averaging the running time over the 30 trials.

Figure 5.5 illustrates the effect of bandwidth β/L on the optimality of the approximate temporal correspondence. The error E_C along the least cost path decreases with increasing bandwidth, and it decreases to the minimum when $\beta/L = 16\%$. This means that a small bandwidth (e.g., 20%) relative to L is enough to obtain the optimal solution.

Figure 5.6 shows that the least cost path varies for $\beta/L < 20\%$. When the bandwidth is small ($\leq 11\%$), the least cost path is constrained within a small region along the diagonal. The least cost path for $\beta/L = 16\%$ is almost the same as the optimal solution in Figure 5.1. The least cost path converges to the optimal solution when $\beta/L \geq 20\%$.

Figure 5.7 shows the effect of bandwidth on the computation time in determining the approximate temporal correspondence. This test result is obtained with window size w set at $k = 10$ such that it does not affect the optimality of the temporal correspondence. From the figure, we can see that the computation time increases with increasing bandwidth. This observation is consistent with the computational complexity $\mathrm{O}(w\beta L')$ analyzed in Section 4.5.2.

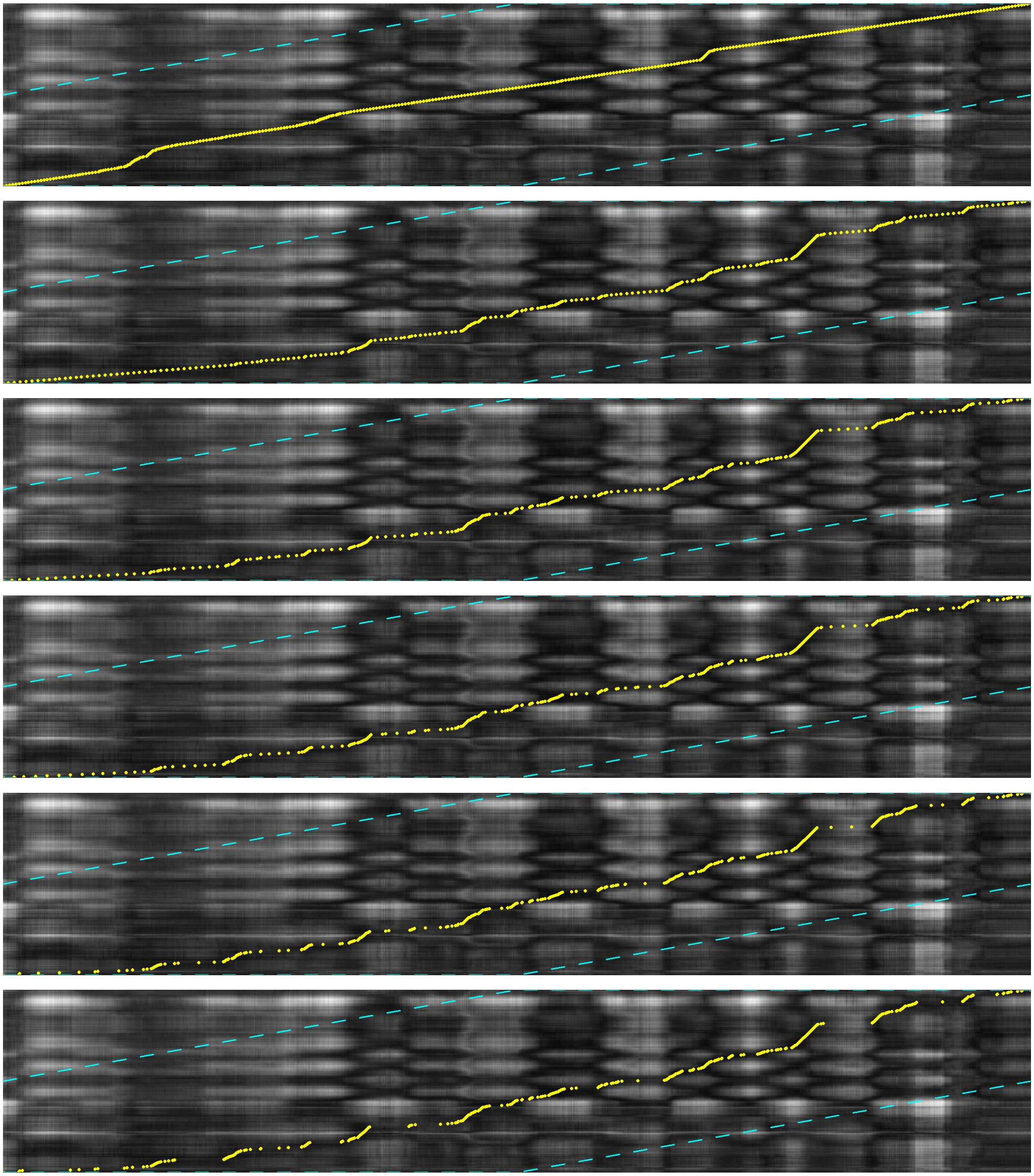


Figure 5.4: The estimated approximate temporal correspondences at different window sizes. The yellow dots along the diagonal in each image represent the approximate temporal correspondence. The two dashed (cyan) lines represent the boundary of the band $b = 50\%$. The images from top to bottom correspond to k values of 1, 2, 3, 4, 7, and 30.

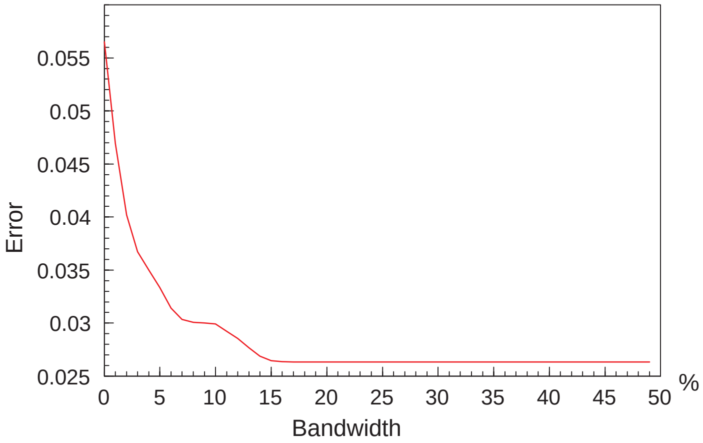


Figure 5.5: The effect of bandwidth on the optimality of approximate temporal correspondence.

5.1.5 Optimal Solution with Small Window Size and Bandwidth

Based on the previous test results, it is observed that the optimal solution of the approximate temporal correspondence can be obtained for window size parameter $k \in [10, L']$ and bandwidth $\beta/L \in [20\%, 100\%]$ (Figure 5.8). A test was performed with window size $k = 10$ and bandwidth $\beta/L = 20\%$. The least cost path found in this test (Figure 5.9) is the same as the optimal solution (Figure 5.1). Therefore, we can conclude that a small window size and bandwidth are enough for finding the optimal solution. At these settings, 60% of the computation time is saved compared to searching the correspondence matrix with window size $k = 10$ and $\beta/L = 100\%$ (Figure 5.7).

5.2 Estimation of Posture Candidates

5.2.1 Test Overview

This section evaluates the accuracy of the algorithm at Stage 3 that estimates the posture candidates that match the input video (Section 4.6). After estimating the approximate temporal correspondence C, the 3D reference postures that approximately correspond to the input images are obtained. Then, the posture candidate estimation algorithm generated posture candidates as follows. Using the corresponding 3D reference posture as

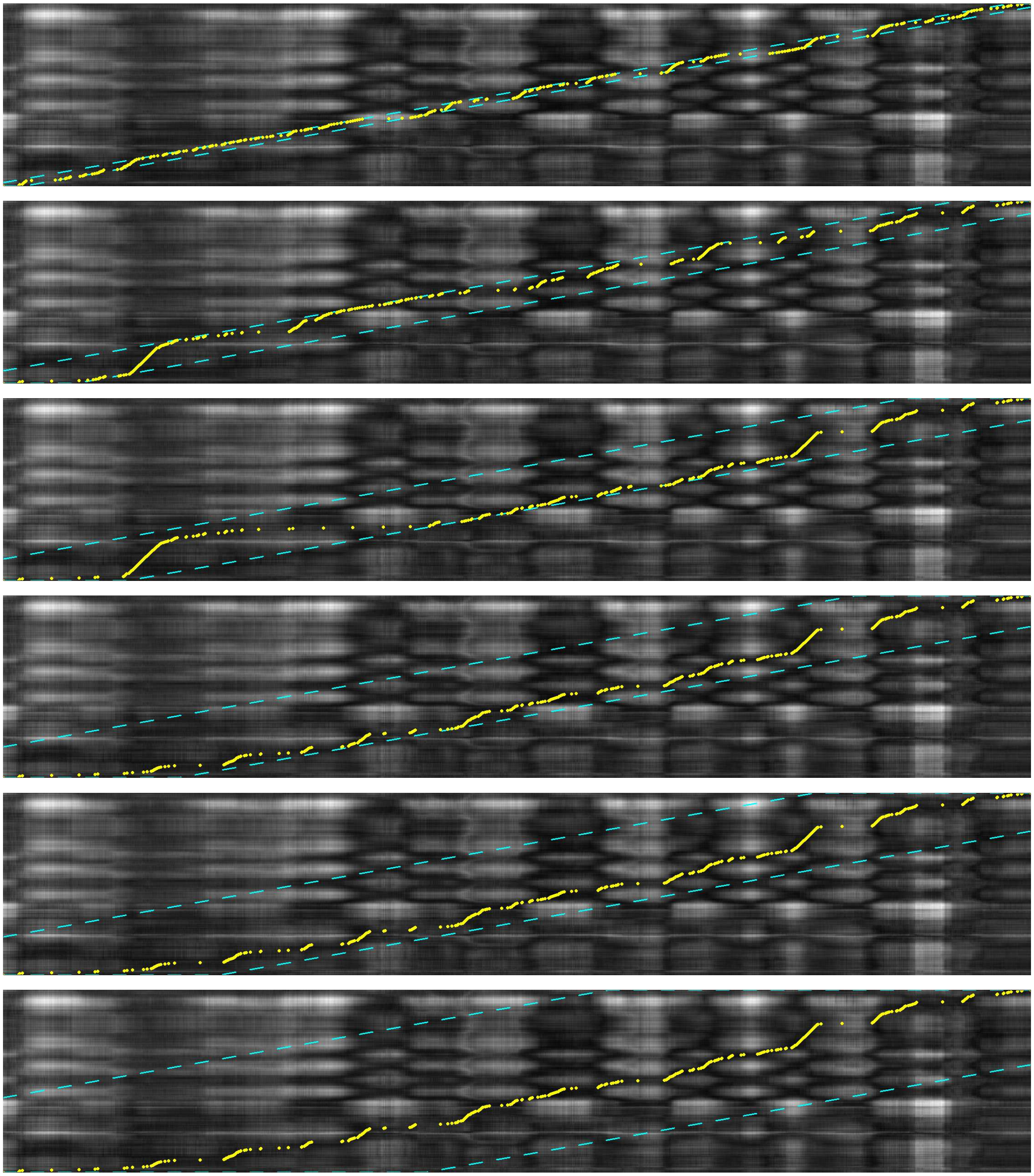


Figure 5.6: The estimated approximate temporal correspondences at different bandwidths. The two dashed (cyan) lines represent the boundary of the band around the diagonal, and the yellow dots represent the approximate temporal correspondence. The images from top to bottom correspond to bandwidth values β/L of 1%, 6%, 11%, 16%, 20%, and 40%.

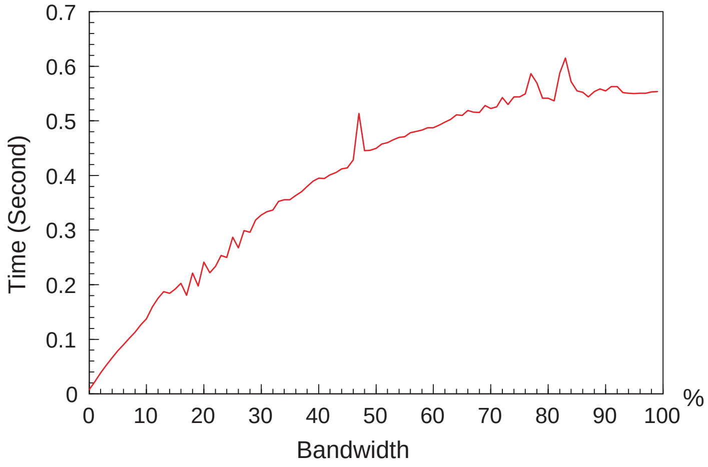


Figure 5.7: Computation time increases with bandwidth.

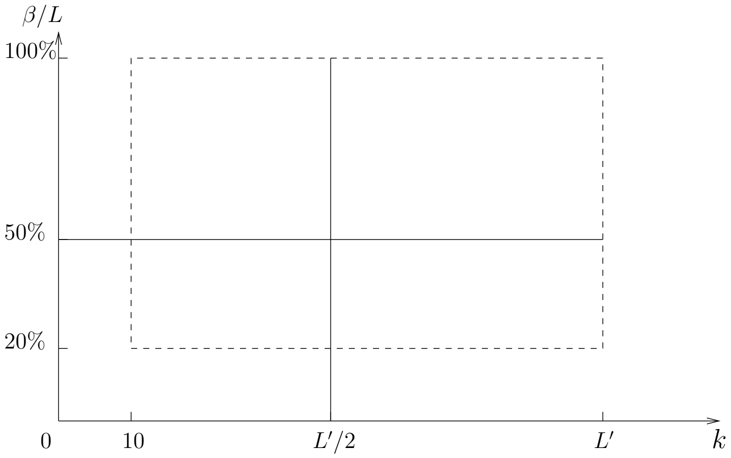


Figure 5.8: The optimal operation region. The range of window size and bandwidth parameters tested in the experiments are shown as the solid vertical and horizontal lines. The optimal solution of approximate temporal correspondence can be obtained by setting the two parameters to any values in the dashed rectangular region.

an initial posture estimate, the algorithm transformed and articulated the initial posture to obtain a new posture whose projection best matched the input body region in the input image. Due to possible depth ambiguity, multiple posture candidates that all match the input image were also generated.

In order to quantitatively evaluate the algorithm, the ground-truth 3D postures of input images should be known. However, it is very difficult to obtain ground-truth 3D postures from real input images. Therefore, we generated the test data as follows: 110

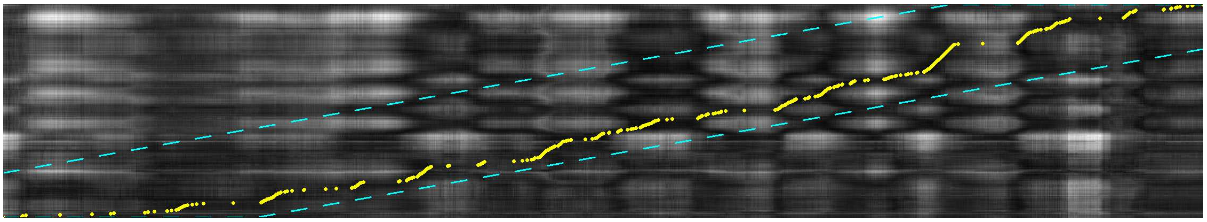


Figure 5.9: Approximate temporal correspondence with small window size and bandwidth. Here $k = 10$ for window size w and $b = 20\%$ for bandwidth. The approximate temporal correspondence (by yellow dots along the diagonal) is almost the same as the optimal in Figure 5.1(b).

reference postures were selected at regular intervals from the 3D Taichi motion. Each selected 3D posture (Figure 5.10(a)) was then mapped to a 3D human model. The articulated 3D human model was then projected by scaled orthographic projection and rendered using OpenGL to obtain a synthetic input image (Figure 5.10(b)). The 3D reference posture that was used to generate the synthetic input image served as the ground-truth. Next, the ground-truth posture was randomly articulated to generate a new posture to serve as the initial posture. This method was adopted to simulate the condition that in real applications, the actual performer's posture may be different from the corresponding reference posture. Note that some of the synthetic input images generated contained partial self-occlusion, total self-occlusion, and depth ambiguity.

In the experiments, multiple posture candidates were generated by the posture candidate estimation algorithm for each synthetic input image. All the posture candidates yield the same projections of the human body joints. The 2D joint position error between the projections of posture candidates and the projections of the ground truth was used to measure the accuracy of the algorithm. Let $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$ respectively denote the image positions of the i-th body joint projected by the posture candidates and the ground truth. Then, the mean error of 2D joint positions is defined by

$$E_{2P} = \frac{1}{K} \sum_{i=1}^{K} \|\hat{\mathbf{p}}_i - \mathbf{p}_i\| \,, \tag{5.1}$$

where K is the number of body joints, and the maximum error of 2D joint positions is defined by

$$E_{2M} = \max_i \{ \|\hat{\mathbf{p}}_i - \mathbf{p}_i\| \} \,, \tag{5.2}$$

Among these posture candidates, there is one best candidate that is most similar to the ground truth. The posture difference E_{3B} between the best posture candidate and the ground truth was also used to measure the accuracy of the algorithm:

$$E_{3B} = \min_{l'} d_B(B'_{t'l'}, B_{C(t')}) \tag{5.3}$$

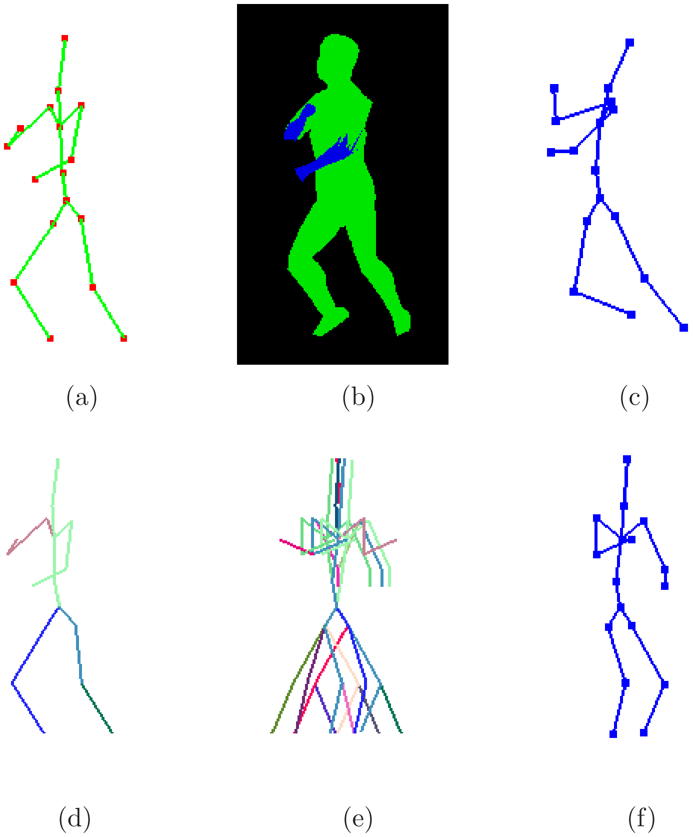
5.2.2 Accuracy of Posture Candidate Estimation

For assessing the accuracy of the posture candidate estimation algorithm, the initial postures were obtained by rotating the joint angles of the selected 3D reference postures by random values in the range $[-20°, 20°]$. In the experiments, the decreasing factor λ was set to 0.9, and the number of iterations was set to 40. In each iteration of the algorithm, 300 pose samples were generated for each body part. About 8 seconds were spent for each iteration, most of which were used to compute the similarity function. 50 posture candidates were estimated by executing the posture estimation algorithm.

Figure 5.10 illustrates a sample test result. Given the input image (Figure 5.10(b)) and the initial posture (Figure 5.10(c)) generated by articulating the ground-truth 3D reference posture (Figure 5.10(a)), multiple posture candidates were estimated by the posture estimation algorithm. All the posture candidates had the same projections from the frontal view (Figure 5.10(d)), but they differed in depth orientations for some body parts, as revealed in the side views (Figure 5.10(e)). Figures 5.10(f) illustrates the side view of the best posture candidate in the candidate set. From Figures 5.10(f) and (d), we can see that the best posture candidate is very similar to the ground truth (Figures 5.10(a)).

For the input image (Figure 5.10(b)) and the initial posture (Figure 5.10(c)), Figure 5.11 shows the decreasing trend of 2D joint position error E_{2P} with respect to the iteration number of the algorithm. The error decreases to a small value of about 1 pixel after 35 iterations, which indicates that the posture estimation algorithm converges after 35 iterations. A 2D joint position error of 1 pixel is the best that can be achieved without using sub-pixel algorithm.

Figure 5.12 illustrates the 2D joint position errors for the 110 input images. The solid curve represents the mean joint position error E_{2P} and the dashed curve represents the maximum joint position error E_{2M}. From the figure, we can see that for most input images, the mean errors are about 1 pixel and the maximum errors are about 2 to 3 pixels. For several input images, the maximum errors are relatively large, i.e., about 10


(a) (b) (c)

(d) (e) (f)

Figure 5.10: Posture candidate estimation from a synthetic input image. (a) Ground-truth 3D posture. (b) Synthetic input image. (c) Initial posture. (d, e) Posture candidates viewed from the front and the side. (f) Best posture candidate viewed from the side.

to 50 pixels. Figure 5.13 illustrates two examples (frames 66 and 84) with the largest 2D joint position errors. The left lower arm is totally occluded in the first example, and the right arm is totally occluded in the second example. In these cases, the pose estimates of the occluded body parts are very different from the ground truth, resulting in the large 2D joint position errors.

Figure 5.14 shows the posture errors E_{3B} of the posture candidates that best match the ground-truth postures for all the input images. At Stage 3 of the algorithm, it cannot determine the optimal postures as yet (Section 4.6.5). Nevertheless, the optimal postures should match the ground-truth postures well. Therefore, the errors of the best matching posture candidates are used here to evaluate the accuracy of the algorithm.

For this test, posture error ranges from 2° to 15°, with a mean of 7° and a standard deviation of 2.6°. Similar to Figure 5.12, the larger errors happen in the input images

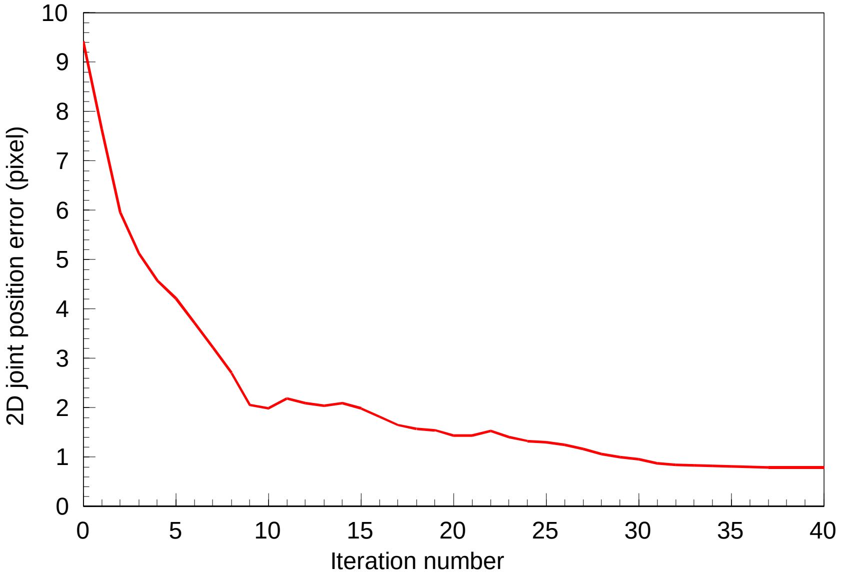


Figure 5.11: 2D joint position error E_{2P} with respect to iteration number.

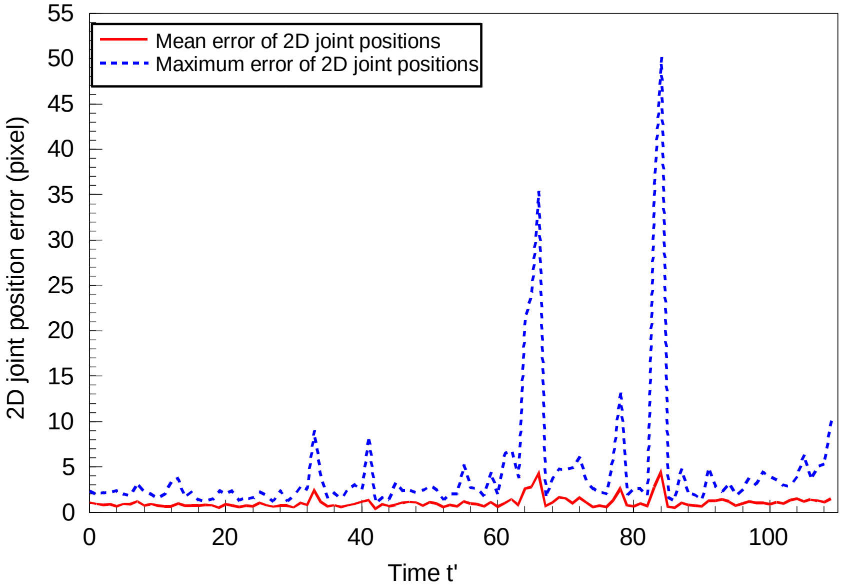
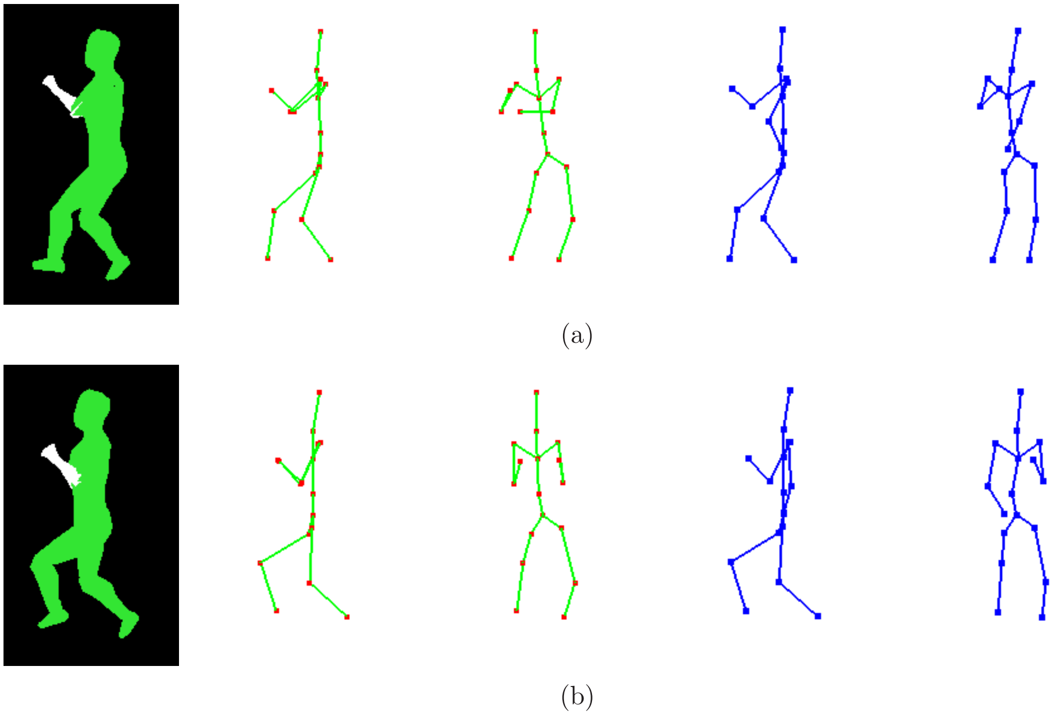


Figure 5.12: 2D joint position errors for the synthetic input images. Solid curve: mean joint position error E_{2P}. Dashed curve: maximum joint position error E_{2M}.

with total occlusion for some body parts. For the other input images, the posture error are mainly due to rotation of body parts in depth. For a body part of length 30cm parallel to the image plane with camera scale $s = 1.2$ pixels/cm, the length of the body part in the image changes by only one pixel when the body part is rotated by 14° in



(a)

(b)

Figure 5.13: Examples of input images with totally occluded body parts. The left lower am is occluded in the first input image, and the the right arm is occluded in the second input image. For each image, the occluded body parts are wrongly estimated, which can be observed by comparing the frontal views and side views of the ground-truth posture (the first two skeletons in green) and the best posture candidate (the last two skeletons in red).

depth. That is, there is almost no change in the input body region when rotating the body part by, e.g., 10°. Therefore, a mean error of 7° is reasonable and acceptable for a posture estimation algorithm using a single camera view. The accuracy can be further improved by implementing sub-pixel algorithm, which takes more time to execute.

Figure 5.15 shows sample test results for the synthetic images. The first row illustrates the input images. The second and third rows show the frontal and side views of all the posture candidates estimated for each input image. The fourth row highlights the side views of the posture candidates that best match the actual postures in the input images. From the second row, we an see that all the estimated posture candidates are aligned when viewed from the front. That is, their projections match the input image equally well. However, due to depth ambiguity, these posture candidates are not the same, as revealed in their side views (third row). From the fourth row, we can see that the best posture candidate for each input image corresponds to the ground-truth posture in the

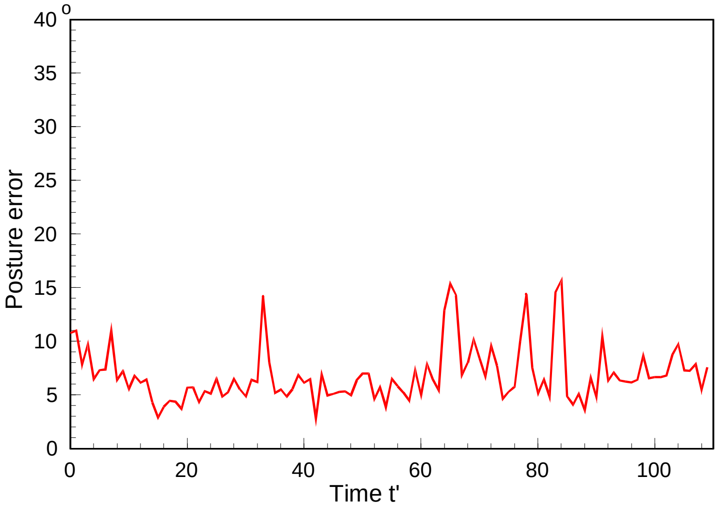


Figure 5.14: Posture errors E_{3B} of the synthetic input images.

sense that the depth orientation of each body part in the best candidate is the same as that in the ground truth.

5.3 Estimation of Posture Candidates from Real Input Images

5.3.1 Test Overview

The objective of the tests is to qualitatively evaluate the correctness of the algorithm for posture candidate estimation. Two sets of motion sequences were used for the tests:

1. 3D Taichi motion. The 3D reference motion comprised a sequence of 2250 reference postures, and the real input video consisted of 339 input images of size 320 × 240.
2. 3D golf swing motion. The 3D reference motion comprised a sequence of 250 reference postures, and the real input video consisted of 51 input images of size 320 × 240.

The following test procedure was performed for each motion reference. First, camera calibration algorithm was executed. Then, approximate temporal correspondence was estimated by the dynamic programming algorithm with bandwidth and window size

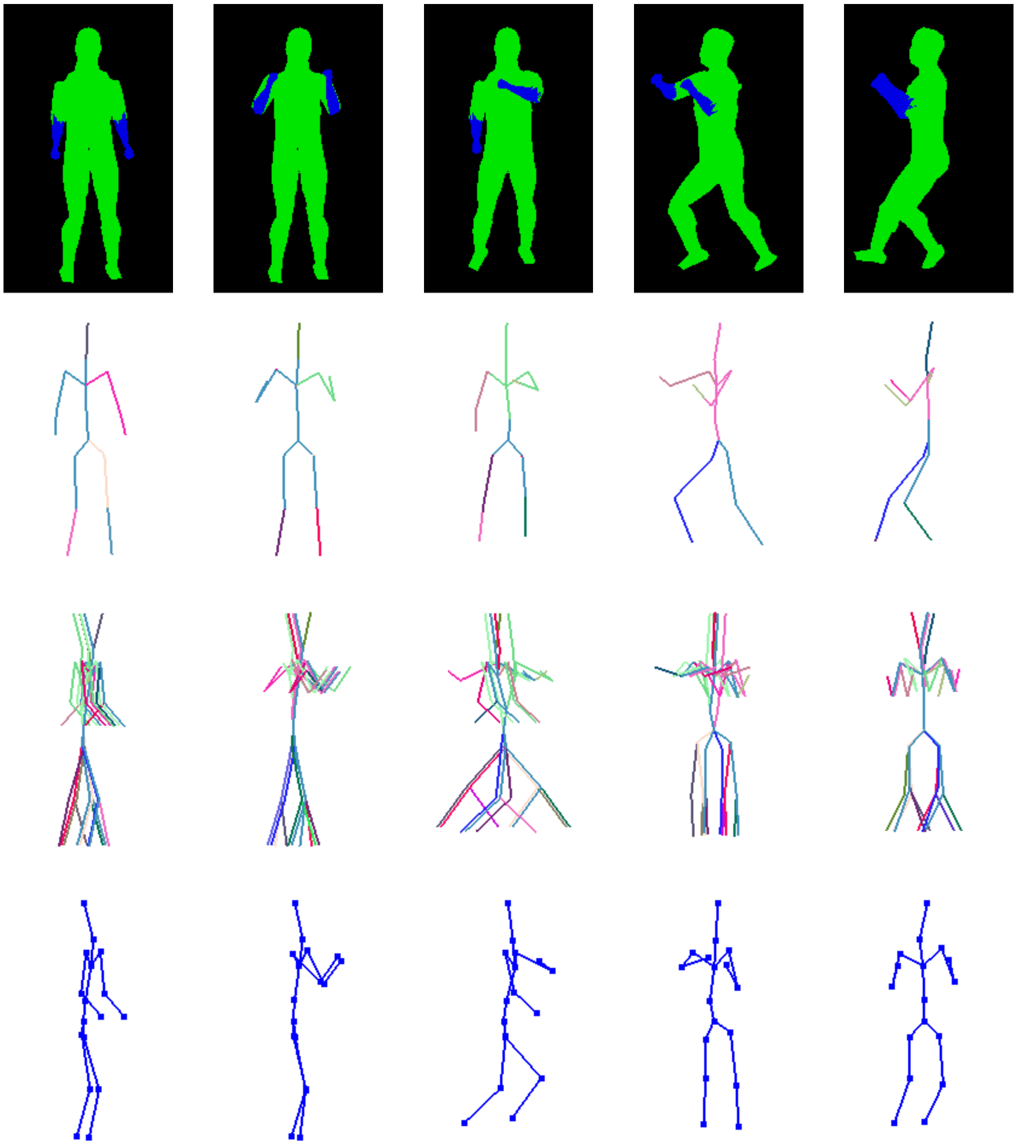


Figure 5.15: Estimation of posture candidates from synthetic images. First row: input images. Second to third rows: the frontal and side views of all the posture candidates for each input image. Every candidate has a unique color. Fourth row: the best posture candidates viewed from the side.

parameters set at $\beta/L = 30\%$ and $k = 10$ respectively. Next, the posture candidate estimation algorithm was executed with the parameters as listed in Section 5.2.1.

5.3.2 Test Results and Discussions

Figure 5.16 shows sample test results for the Taichi sequence. The first row illustrates sample input images. The second and third rows show the frontal and side views of all the posture candidates estimated for each input image. The fourth row illustrates the side views of the posture candidates that best match the actual postures in the input images. These best matching candidates are identified manually for the purpose of assessing the accuracy of the algorithm. From the second row, we an see that all the estimated posture candidates are aligned when viewed from the front. That is, their projections match the input image equally well. However, due to depth ambiguity, these posture candidates differ in the depth orientations of some body parts, as revealed in their side views (third row). From the fourth row, we can see that the best posture are quite similar to the actual performer's postures. The depth orientations of the body parts in the best candidates are the same as those in the performer's actual posture.

Figure 5.17 illustrates sample test results for the golf swing sequence. Similar to the results for the Taichi sequence (Figure 5.16), multiple posture candidates were estimated for each input image (third row), and all the candidates have the same frontal view and match the input image equally well (second row). Among the candidates for each input image, there exists a best posture candidate (fourth row) that matches the performer's actual posture.

5.4 Estimation of Performer's Segment Boundaries

5.4.1 Test Overview

This section assesses the performance of the segment boundary estimation algorithm. In the tests, the 3D reference motion comprised a sequence of 2250 Taichi reference postures, and the performer's input video consisted of 339 input images of size 320×240. For each input image, 25 posture candidates were estimated by the posture estimation algorithm.

The reference motion contained 7 segment boundaries, including the first and the last segment boundaries which were the first and the last frames. The other 5 segment boundaries were determined manually by domain knowledge. The corresponding performer's segment boundaries in the performer's motion were also determined manually to serve as the ground truth.

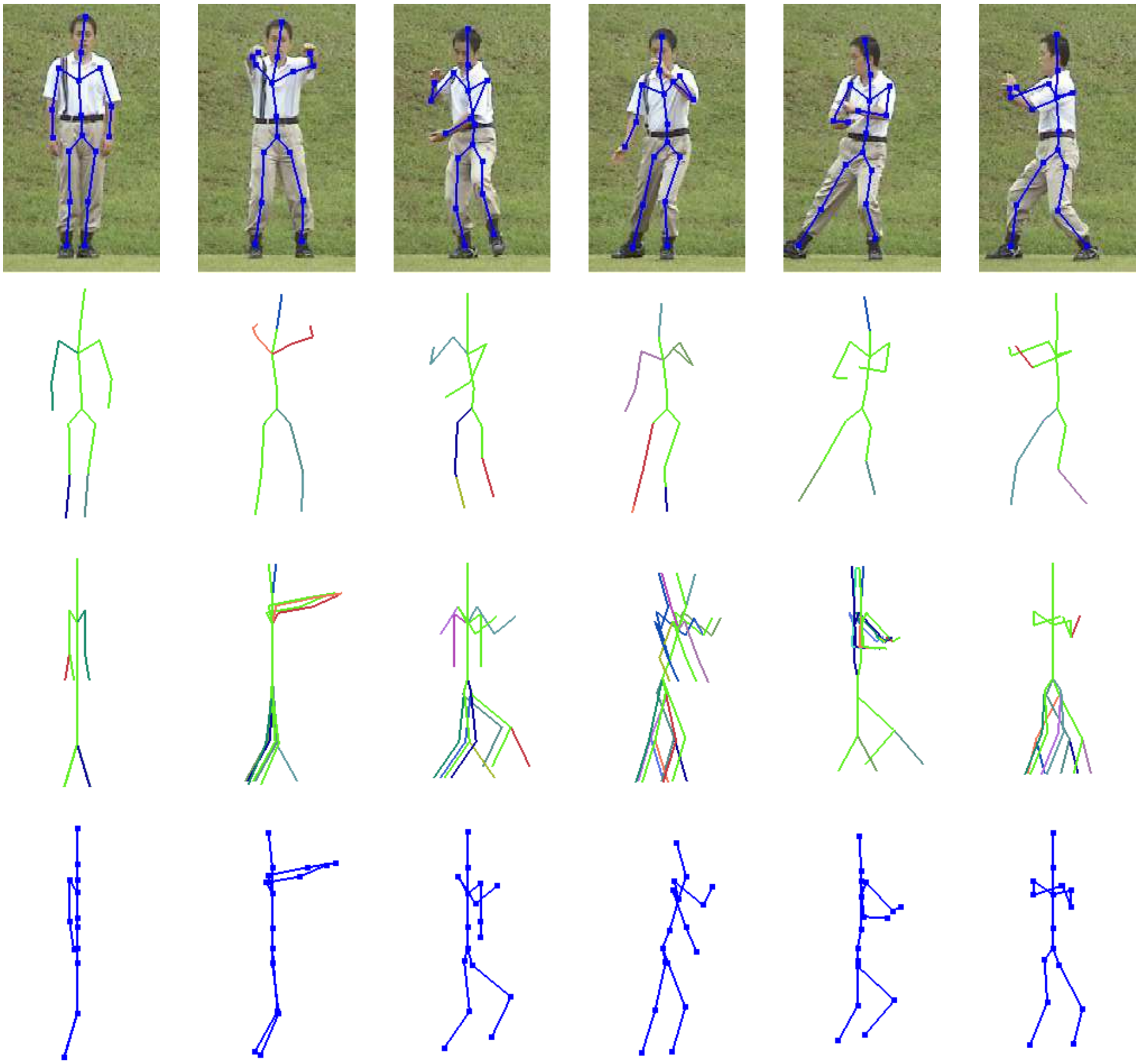
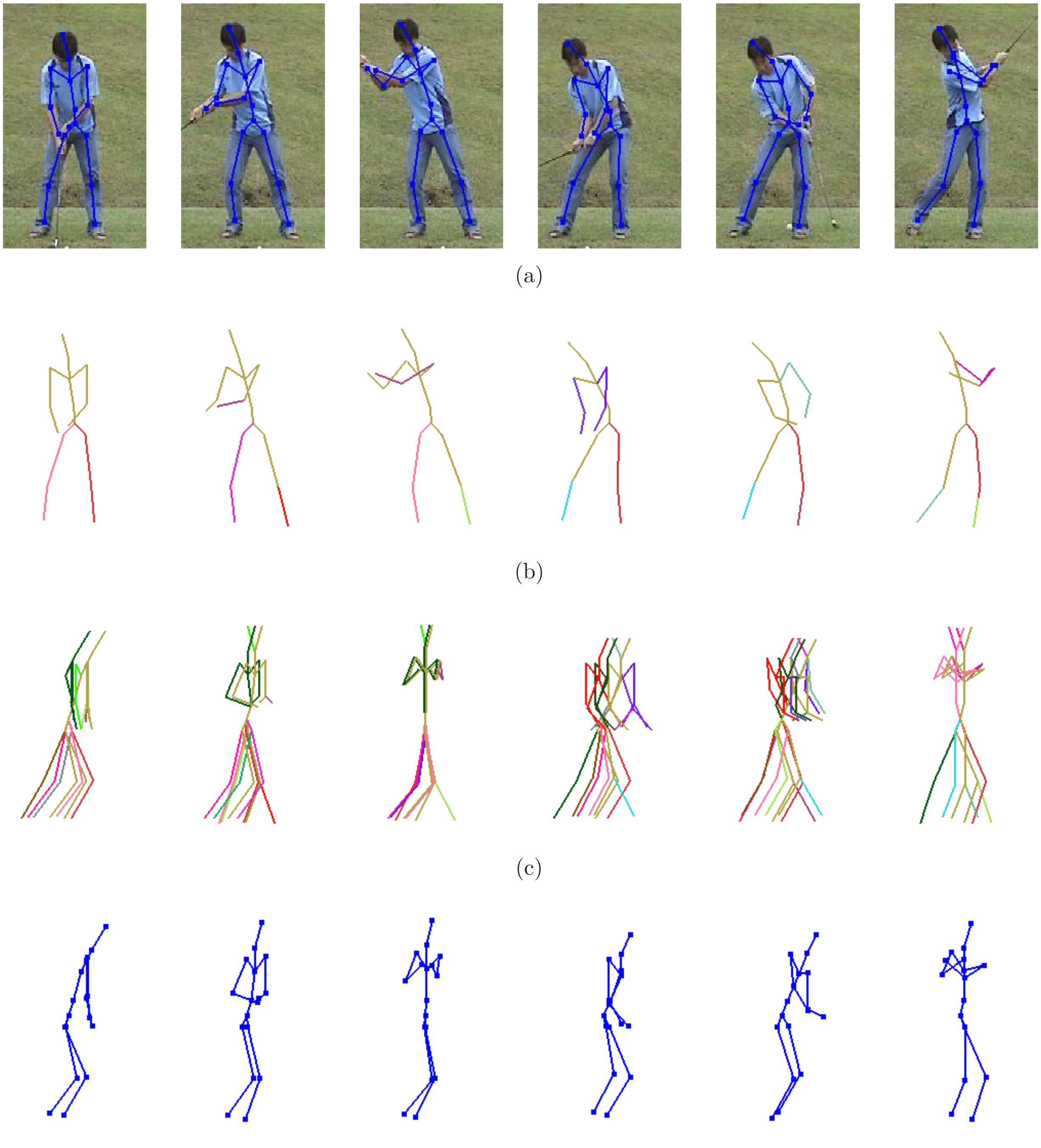


Figure 5.16: Estimation of posture candidates from real Taichi images. First row: input images with a posture candidate (skeleton) projected. Second and third rows: the frontal and side views of all the posture candidates for each input image. Each candidate has a unique color. Fourth row: the best posture candidates viewed from the side.

5.4.2 Determination of Segment Boundary Parameter

This test determines the segment boundary parameter for the Taichi motion, i.e., the minimum direction change of the body joint at the segment boundaries (Section 2.1.1). The segment boundary parameter was used to estimate the performer's segment boundaries in the next test.



(a)

(b)

(c)

(d)

Figure 5.17: Estimation of posture candidates from real golf swing images. First row: input images with a posture candidate (skeleton) projected. Second and third rows: the frontal and side views of all the posture candidates for each input image. Each candidate has a unique color. Fourth row: the best posture candidates viewed from the side.

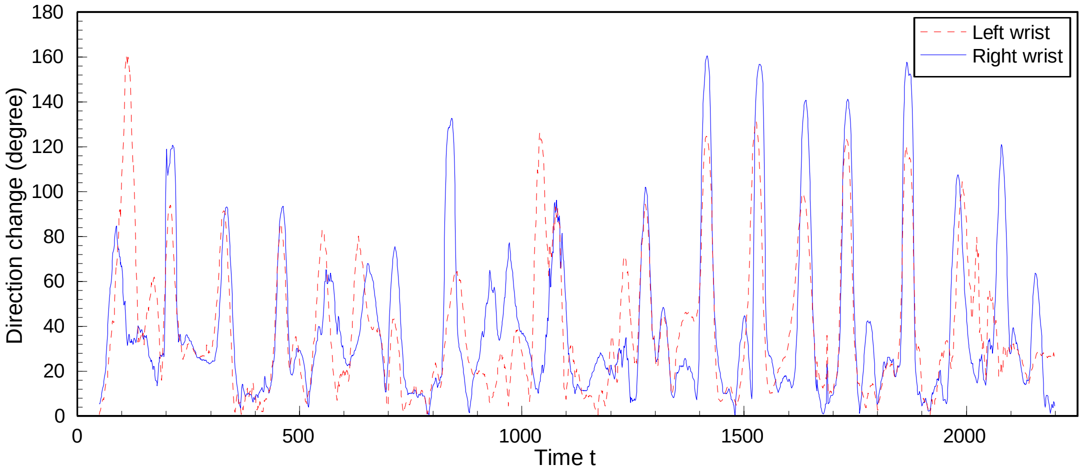


Figure 5.18: The change of motion direction for the left wrist and the right wrist joints.

From the 3D Taichi reference motion, the 3D velocity direction $\mathbf{v}_t$ of each body joint at each time instance t was first computed. Then, the direction change of the velocity was calculated as $\cos^{-1}\{\mathbf{v}_t \cdot \mathbf{v}_{t+\Delta t}/(\|\mathbf{v}_t\|\|\mathbf{v}_{t+\Delta t}\|)\}$, where Δt was set to 10 by considering that the 3D Taichi reference motion is slow and captured at a high frame rate (120fps).

Figure 5.18 illustrates the direction change of the left wrist's and the right wrist's. From the figure, we can see that there are about 20 reference frames with large direction changes for both the left wrist and the right wrist. Large change in direction for left wrist and right wrist may not occur in the same frame. This indicates that the two hands sometimes do not simultaneously change their motion directions. Not all the frames with large direction change correspond to the reference boundaries. It is reasonable because the hands can change motion directions more than once in a motion segment.

Figure 5.19 illustrates the motion direction change for the left knee and the right ankle joints. It shows that there are too many frames with large direction changes, which indicates that it would be very difficult to determine the segment boundaries using the properties of these joints.

At the reference segment boundaries (second row in Table 5.1), there is always a large direction change for the right wrist by at least 60°. Therefore, the right wrist is used as the joint that indicates segment boundary and the direction change threshold is set at 60° for the Taichi motion. For golf swing motion, the threshold is found to be 120°

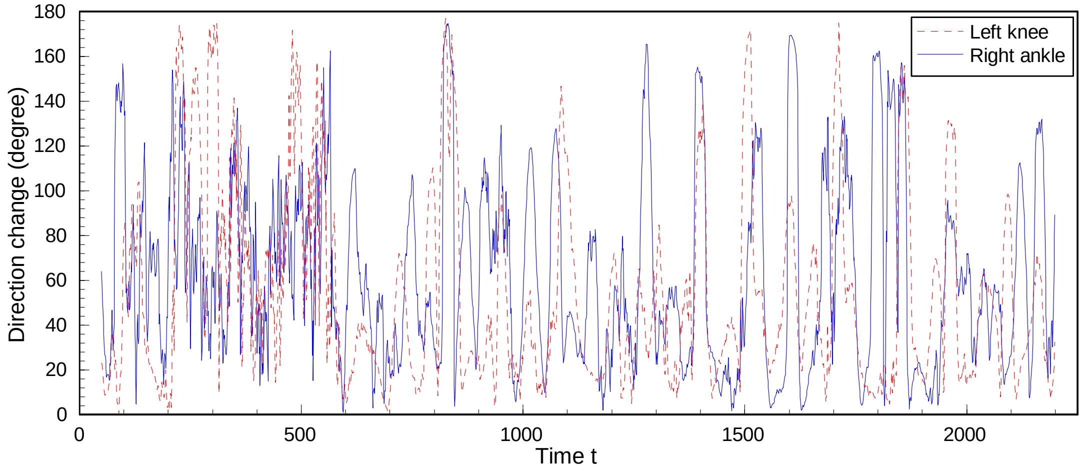


Figure 5.19: The change of motion direction for the left knee and the right ankle joints.

5.4.3 Estimation of Performer's Segment Boundaries

In this test, the performance of the algorithm for estimating the performer's segment boundaries was evaluated. First, 25 posture candidates were estimated by running the algorithms in the previous stages with the parameters listed in Section 5.3. Then, the performer's segment boundaries were estimated by the segment boundary estimation algorithm with the parameters determined in Section 5.4.2.

Table 5.1 illustrates the test results. The first row lists the 7 reference segment boundaries. The second and third rows list the frame numbers of the reference segment boundaries and the ground-truth performer's segment boundaries. The fourth row lists the initial estimates of the performer's segment boundaries obtained from the approximate correspondence obtained from Stage 2. The last three rows list the final estimates of the performer's segment boundaries for input video frame rates of 25fps and 12.5fps respectively. From the fourth row, we can see that the initial estimates are very different from the ground-truth performer's segment boundaries because the temporal correspondence determined at Stage 2 is only an approximate that does not take into account articulation of body parts. In comparison, the final estimates of the performer's segment boundaries (fifth row) differ from the ground truth (third row) by at most two frames, which is reasonably small in an input video of 339 frames. When the input video has a lower frame rate, the estimates of the performer's segment boundaries (last row) are also very close to the ground truth, which indicates that the algorithm is robust.

Figure 5.20 visually shows the boundary estimation results. The first row displays

Table 5.1: Segment boundaries of Taichi sequence. The first segment boundary is the first frame, and the last segment boundary is the last frame. The frame numbers of the performer's segment boundaries for 12.5fps input video are up-scaled for ease of comparison with the ground truth.

Segment boundaries		B1	B2	B3	B4	B5	B6	B7
Reference segment boundaries		0	215	570	972	1535	1733	2249
Performer's segment boundaries (Ground truth)		0	60	91	145	227	262	338
Performer's segment boundaries (Initial estimate)		0	16	85	134	233	265	338
Performer's segment boundaries (Final estimate: 25fps)		0	60	90	143	228	261	338
Performer's segment boundaries (Final estimate: 12.5fps)	actual	0	29	46	70	114	131	169
	up-scaled	0	58	92	140	228	262	338

the reference postures at the reference segment boundaries. The second to the fourth rows display the input images for the ground truth, the initial estimates, and the final estimates of the performer's segment boundaries. From the results, we can see that the performer's postures in all the displayed input images are similar to the corresponding reference postures. It indicates that the performer's segment boundaries cannot be accurately estimated only by the similarity between the input body regions and the projected body region at the reference segment boundaries. This also explains why the initial estimates are not accurate because the approximate temporal correspondence is determined without taking into account articulation of body parts. In contrast, the segment boundary estimation algorithm accurately estimates the performer's segment boundaries by using the segment boundary properties (Section 2.1.1).

5.5 Posture Candidate Selection and Estimation of Posture Errors

5.5.1 Test Overview

This section evaluates the performance of the algorithm for posture candidate selection and the refinement of temporal correspondence. Then, the posture errors between the selected best posture candidates and the corresponding reference postures are computed to measure the errors of the performer's postures.

Two sets of motion sequences were used to evaluate the algorithm: (1) 3D Taichi reference motion with 2250 reference postures and real input video with 339 input images, and (2) 3D golf swing motion with 250 reference postures and input video with 51 input images. For each input image, 25 posture candidates were generated by running the algorithms in all the stages with the parameters described in Section 5.4.2. The performer's segment boundaries in the performer's motion were determined by the boundary estimation algorithm with the parameters described in Section 5.4.3.

5.5.2 Refinement of Temporal Correspondence

The refinement of the temporal correspondence C is influenced by two factors: the window size $w = kL_i/L'_i$ and the normalized bandwidth β_i/L_i, where L_i and L'_i are the lengths of the i-th reference motion segment and the performer's segment respectively, and k is a small constant. Three tests were performed. In the first test, the optimal solution of refined temporal correspondence C was first computed by setting $\beta_i/L_i = 100\%$ and $w = L_i$ for the i-th motion segment. For each motion segment, a posture candidate was selected for each input frame and the refined temporal correspondence was efficiently determined by the dynamic programming algorithm (Section 4.7.2).

In the second test, the effect of window size on the refined temporal correspondence was evaluated by varying k ($w = kL_i/L'_i$) 0 to 30. The bandwidth was set to 50% to remove the effect of bandwidth on C.

In the third test, the effect of bandwidth β/L on C was evaluated by varying β/L from 0% to 30%. The window size w was set at $L_i/2$ to remove the effect of window size on C. All the tests were performed on the Taichi motion sequences. The error

E_F (Equation 4.24) was computed to measure the optimality of the refined temporal correspondence C. Note that E_F has been normalized by the length of the input segment.

Figure 5.21 illustrates the optimal solution of the refined temporal correspondence C for the whole sequence. Compared to the approximate C obtained in Stage 2, the refined C is more accurate because it satisfies the segment boundary constraint.

The effects of the window size and the bandwidth on the refined temporal correspondence are illustrated for the 4th. Similar results have been obtained for other motion segments.

Figures 5.22 and 5.23 illustrate the effect of window size $w = kL_i/L'_i$ on the optimality of the refined temporal correspondence. From the decreasing trend of error E_F with respect to k, we can see that a small k (e.g., 10) is enough for determining the optimal solution of temporal correspondence. Figure 5.23 shows the refined temporal correspondences at two different window sizes. We can see that the least cost path at $k = 5$ is already very similar to the optimal solution ($k = 25$). The least cost path at $k = 10$ is almost the same as the optimal solution and therefore not displayed.

Figures 5.24 and 5.25 illustrate the effect of bandwidth on the optimality of the refined temporal correspondence. It shows that the error E_F along the least cost path decreases with increasing bandwidth, and it decreases to the minimum when $\beta_i/L_i = 20\%$. This means that a small bandwidth (e.g., 20%) relative to L_i is enough to determine the optimal least cost path. Figure 5.25 shows that the temporal correspondence at $\beta_i/L_i = 5\%$ is already very close to the optimal solution (at $\beta_i/L_i = 30\%$). The temporal correspondence at $\beta_i/L_i = 20\%$ is the same as the optimal solution of the refined temporal correspondence and so is not displayed.

5.5.3 Final Estimation of Posture Errors

In this test, the performance of the algorithm in posture selection was evaluated, and the posture errors between the selected posture candidates $B'_{t'\ell(t')}$ and the corresponding reference postures $B_{C(t')}$ were evaluated. Posture candidates and segment boundary were estimated by running the algorithms in the previous stages with the parameters listed in Section 5.4.3. Then, the refined temporal correspondence was determined by setting window size factor $k = 10$ and bandwidth $\beta/L = 25\%$ for each motion segment. The posture errors were measured by $d_B(B_{C(t')}, B'_{t'\ell(t')})$ (Equation 4.5).

Figure 5.26 illustrates the posture errors $d_B(B_{C(t')}, B'_{t'\ell(t')})$ computed for the selected posture candidates for the Taichi motion. In the ideal case, the computed posture error $d_B(B_{C(t')}, B'_{t'\ell(t')})$ is equal to the performer's posture error $\varepsilon_{t'} = d_B(B_{C(t')}, B'_{t'})$ (Equation 4.6), where $B'_{t'}$ is unknown. However, in practice, the algorithm is not perfect. As shown in Section 5.2.2, the algorithm has a mean error of 7° (solid line in Figure 5.26) in estimating the postures in the synthetic input images. For real input images, this algorithmic error is expected to be larger, say the mean error plus one standard deviation (dashed line in Figure 5.26). The computed error $d_B(B_{C(t')}, B'_{t'\ell(t')})$ would, in general, include both the algorithmic error and the performer's posture error $\varepsilon_{t'}$. If the algorithmic error is small compared to the computed error, as is the case for Figure 5.26, then there is high confidence that the computed error $d_B(B_{C(t')}, B'_{t'\ell(t')})$ indeed reflects the performer's error $\varepsilon_{t'}$.

Figure 5.26 shows that the computed posture errors are relatively small in most of the first 100 frames compared to the later frames. This is reasonable because the performer starts from a standard standing posture which is easy to maintain a correct posture. As the performer moves on to the more difficult postures, more error are made.

Figure 5.27 shows sample results of the selected posture candidates having small posture errors. The first row illustrates the sample input images with the selected posture candidates projected. The second and third rows show the frontal and oblique views of the selected candidates overlapped with the corresponding reference postures. We can see that the selected posture candidates are similar to the corresponding reference postures. The depth orientations of the body parts in the selected posture candidates are the same as those in the performer's postures in the input images. These results qualitatively verifies that the algorithm can select the best posture candidates.

Figure 5.28 shows sample results of selected posture candidates with larger posture errors. From the frontal and oblique views of the best posture candidates that overlap with the corresponding reference postures, we can see that there are indeed large errors in the arms' rotations in the performer's postures. The performer raised his arms higher compared to the reference postures.

Figure 5.29 illustrates posture error results for the golf swing motion. Similar to the Taichi case, the performer made less error at the beginning of the swing and larger error later on in the swing. This is visually confirmed by the sample results illustrated in Figure 5.30. There is a small posture error in the first frame. For subsequent frames, there are indeed larger errors in the torso orientations. The depth orientations of body parts in the selected posture candidates are the same as those in the performer's posture in the input images. This again qualitatively verifies that the algorithm can select the best posture candidates.

5.5.4 Posture Estimation under Ambiguous Conditions

Figure 5.31 shows sample test results of posture estimation under ambiguous conditions with small algorithmic errors. Due to the lack of ground truth, these algorithmic errors in posture candidate estimation are assessed qualitatively. The first row illustrates sample input images. The second row shows the frontal views of the selected posture candidates for each input image. In the third row, the first three images show the side views of the selected posture candidates corresponding to those in the second row, and the last three show the frontal oblique views of the corresponding selected posture candidates. Each bone has a unique color for identifying the corresponding bones in the views. Left-right ambiguity of the legs (Section 2.1.2) and depth ambiguity exist in all the images. Partial and total self-occlusions of the right arms exist in the last three images. These test results show that the algorithm can select the correct posture candidate even under left-right ambiguity and partial self-occlusion. In the case of total self-occlusion, the algorithm can still often infer the pose of the occluded body part if the performer's postures do not differ greatly from the reference postures.

Figure 5.32 shows sample test results for the posture estimation under ambiguous conditions with larger algorithmic errors. For the first two images, the orientations of the legs are incorrectly estimated. A person cannot stand in the manner depicted by the estimated postures without falling over. The correct posture estimates should be similar to those in the first two images in Figure 5.31. For the third image, the left elbow in the estimated posture bends too much compared to the actual performer's posture. For the fourth image, the two arms are not stretched out by the same amount in the estimated posture. For the last image, the right shoulder joint rotates backward too much in the estimated posture. The correct postures for the last three images should be similar to the estimated postures for the last images in Figure 5.31. The large error in the first three estimated postures are due to depth ambiguity. The large errors in the last two images are due to the occlusions between body parts.

5.6 Summary

This chapter has presented and discussed the experimental results for assessing the performance of the main algorithms described in Chapter 4. The experimental results on estimating the approximate temporal correspondence (Stage 2, Section 5.1) show that a small window size and bandwidth are enough for finding the optimal solution. With small window size and bandwidth for the optimal solution, the computation time can be significantly reduced compared to searching the whole correspondence matrix (Figure 5.7).

In Stage 3, the experiments based on synthetic data show that the mean error of projected joint positions is about 1 pixel for most images, and the maximum error is about 2 to 3 pixels. For the input images with total self-occlusions, the maximum errors are relatively large, i.e., about 10 to 50 pixels. The experiments also show that the posture error varies from 2° to 15°, and the mean error is about 7°. Larger errors happen in the input images with total occlusion for some body parts. For the other input images, the posture error are mainly due to rotation of body parts in depth. A mean error of 7° is reasonable and acceptable for a posture estimation algorithm using a single camera view. The accuracy can be further improved by implementing sub-pixel algorithm, which takes more time to execute. The experiments on real Taichi sequence and golf swing motion qualitatively show that the best posture candidate in each candidate set is quite similar to the actual performer's posture. The depth orientations of the body parts in the best candidate are the same as those in the performer's actual posture.

In stage 4, the estimated segment boundaries differ from the ground truth by at most two frames, which is reasonably small in an input video of 339 frames. After determining segment boundaries, the optimal refined approximate temporal correspondence can be determined by setting a small window size and bandwidth.

Based on the refined temporal correspondence, the posture errors between selected posture candidates and the corresponding reference postures are computed. Test results show that the computed errors are significantly larger than the expected algorithmic error. This indicates that there is high confidence that the computed posture errors indeed reflect the performer's error. The computed posture errors can be used for the coach or the performer to adjust the performer postures in sports training.

Test results on posture candidate selection show that the algorithm can select the correct posture candidate even under left-right ambiguity and partial self-occlusion. In the case of total self-occlusion, the algorithm can still infer the pose of the occluded body part if the performer's postures do not differ greatly from the reference postures.

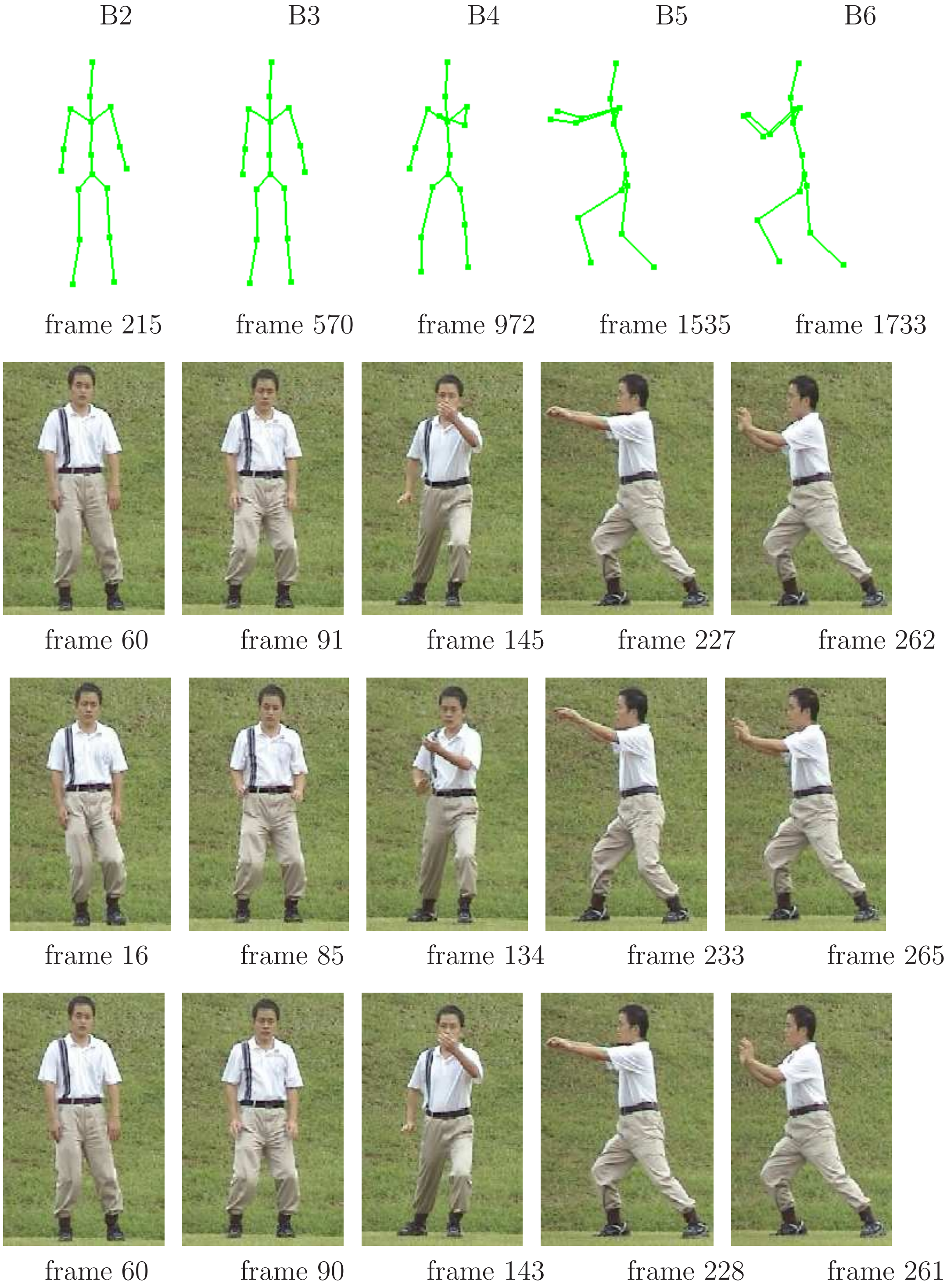


Figure 5.20: Visual illustrations of segment boundaries. First row: reference postures at the reference segment boundaries. Second to fourth rows: the input images of the ground truth, the initial estimates, and the final estimates at the performer's segment boundaries.

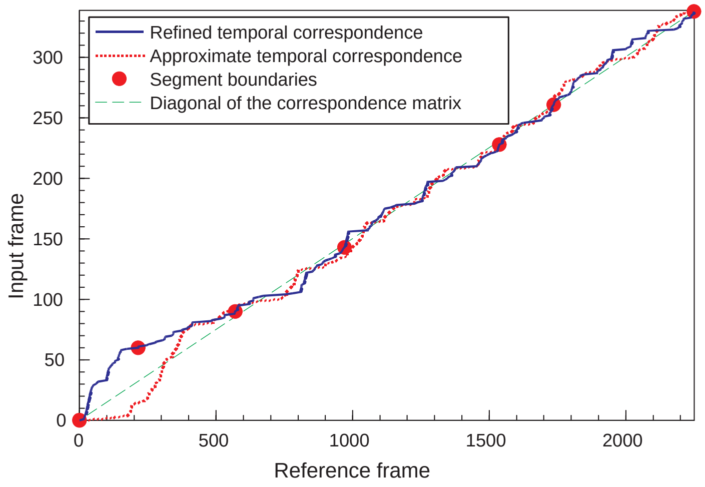


Figure 5.21: The optimal refined temporal correspondence.

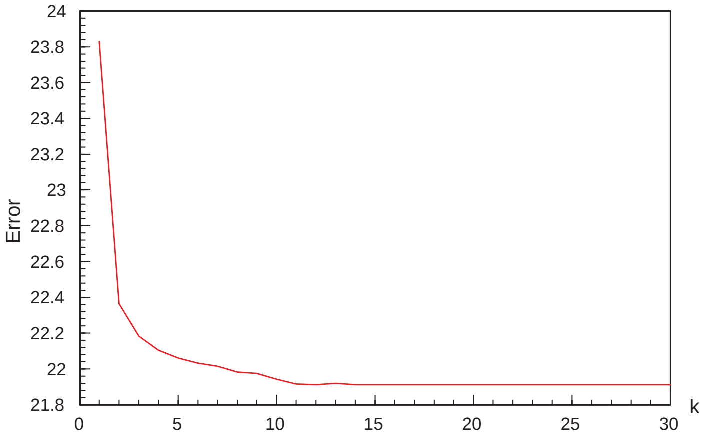


Figure 5.22: The effect of window size on the optimality of the refined temporal correspondence.

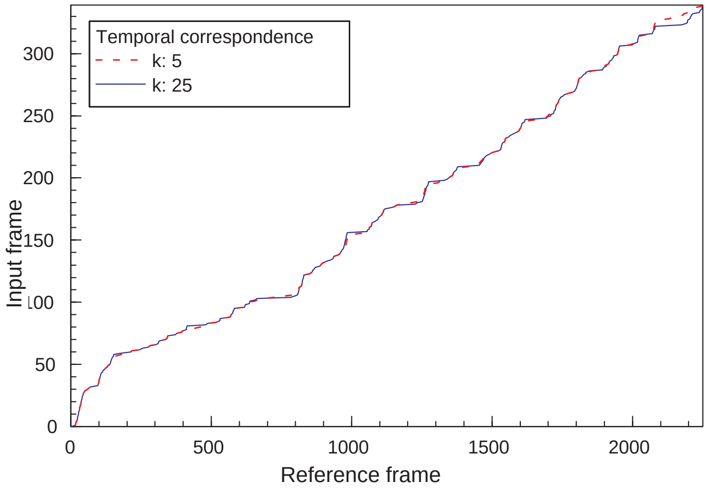


Figure 5.23: Temporal correspondence at three different window sizes.

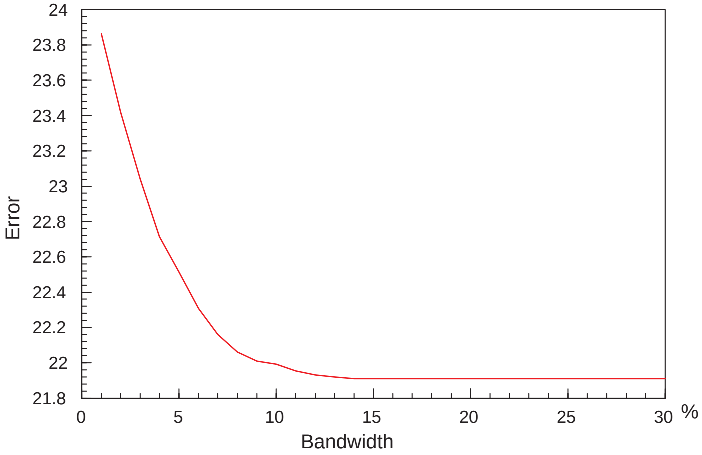


Figure 5.24: The effect of bandwidth on the optimality of the refined temporal correspondence.

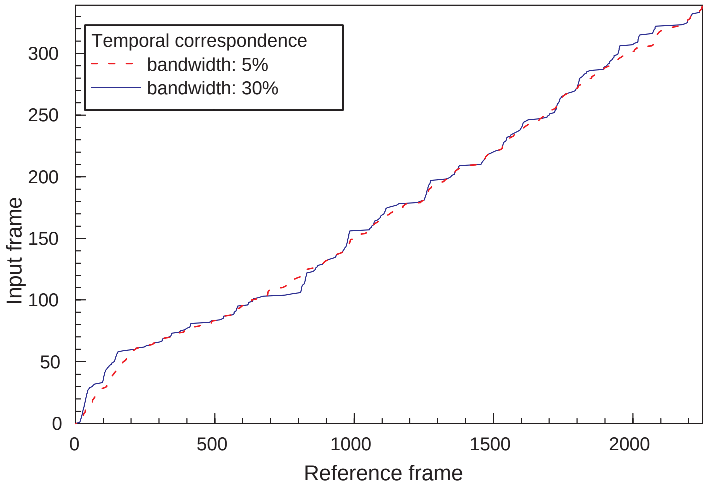


Figure 5.25: Temporal correspondence at three different bandwidths.

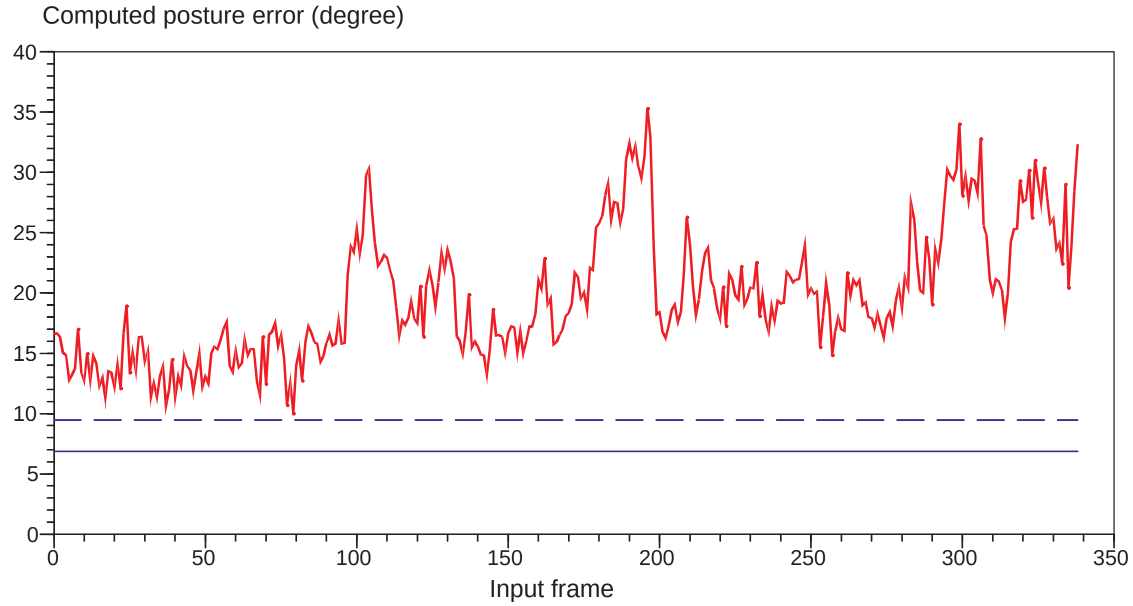
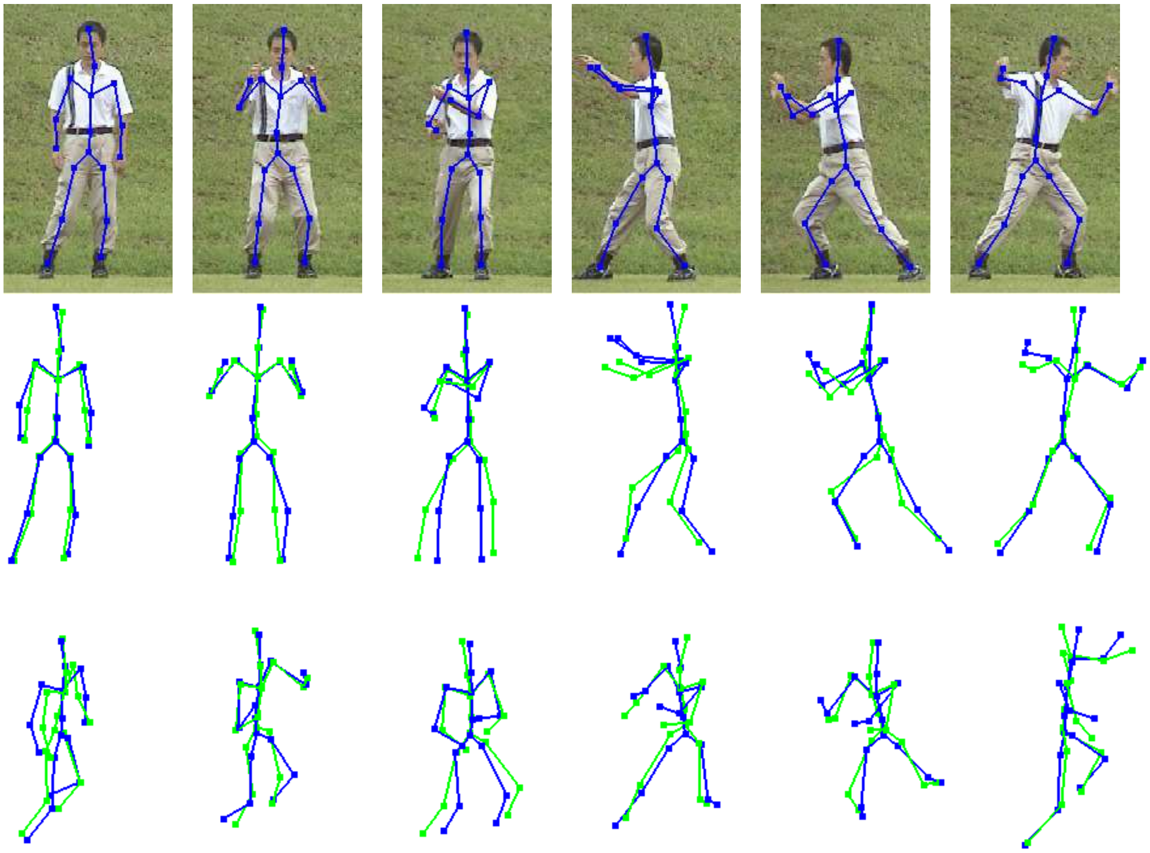
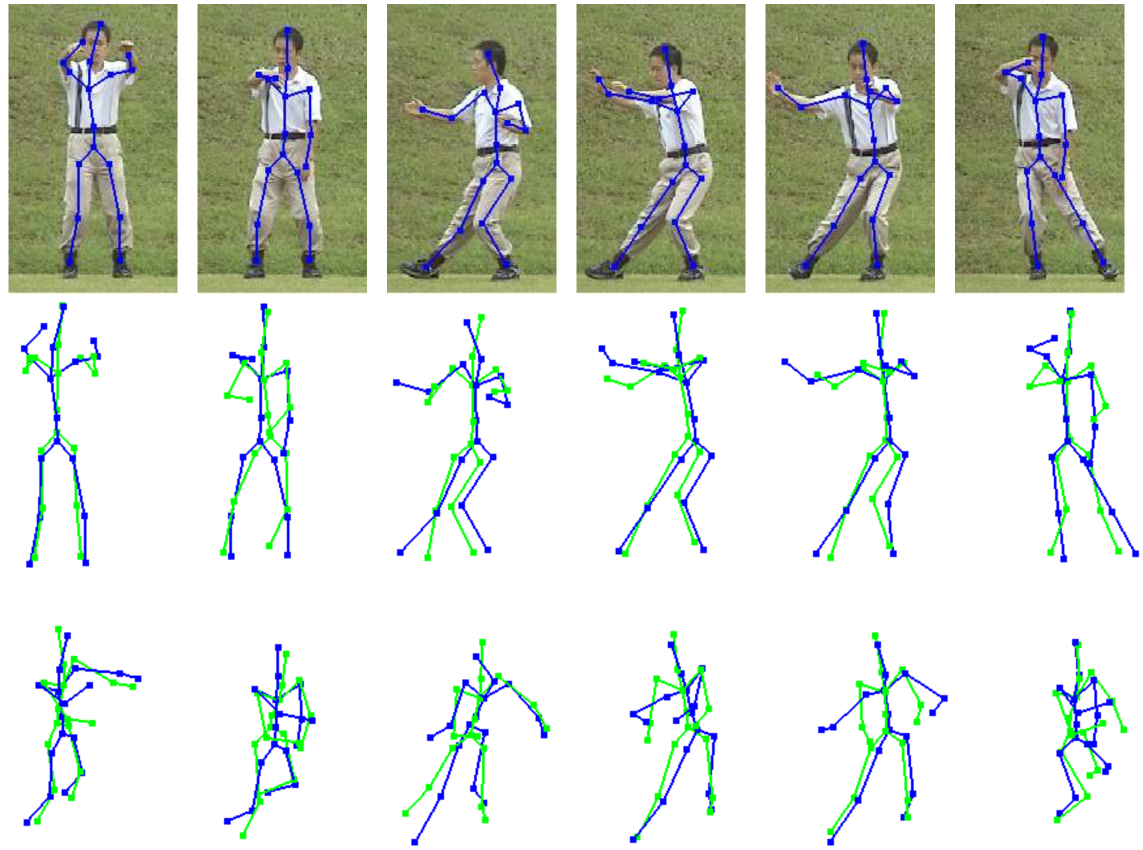


Figure 5.26: Posture error of Taichi motion. Solid curve: computed posture error $d_B(B_{C(t')}, B'_{t'\ell(t')})$. Solid line: algorithmic error in estimating postures in synthetic data. Dashed line: expected algorithmic error.



frame 49 frame 78 frame 133 frame 235 frame 260 frame 335

Figure 5.27: Selected posture candidates with small errors in the Taichi motion. First row: input images with the selected posture candidates projected. Second and third rows: selected posture candidates (blue skeleton) overlapped with the corresponding reference postures (green skeleton) in the frontal and oblique views.



frame 71 frame 103 frame 209 frame 275 frame 282 frame 303

Figure 5.28: Selected posture candidates with larger errors in the Taichi motion. First row: input images with the selected posture candidates projected. Second and third rows: selected posture candidates (blue skeleton) overlapped with the corresponding reference postures (green skeleton) in the frontal and oblique views.

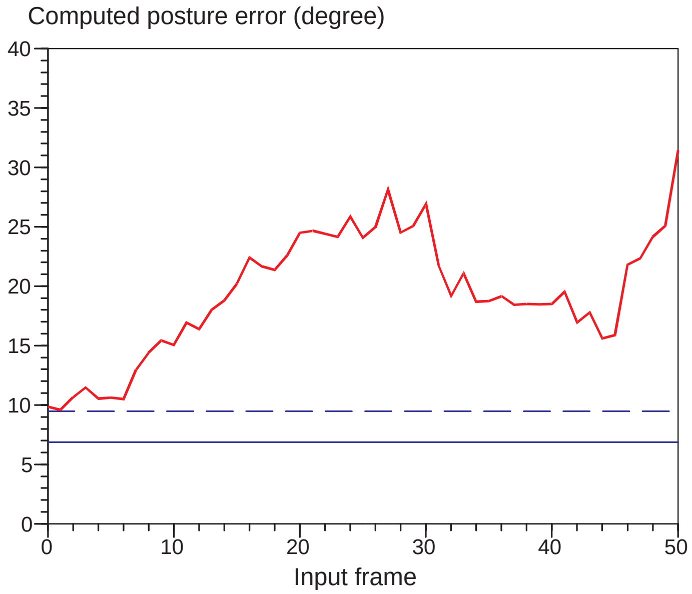


Figure 5.29: Posture error of golf swing motion. Solid curve: computed posture error $d_B(B_{C(t')}, B'_{t'\ell(t')})$. Solid line: algorithmic error in estimating postures in synthetic data. Dashed line: expected algorithmic error.

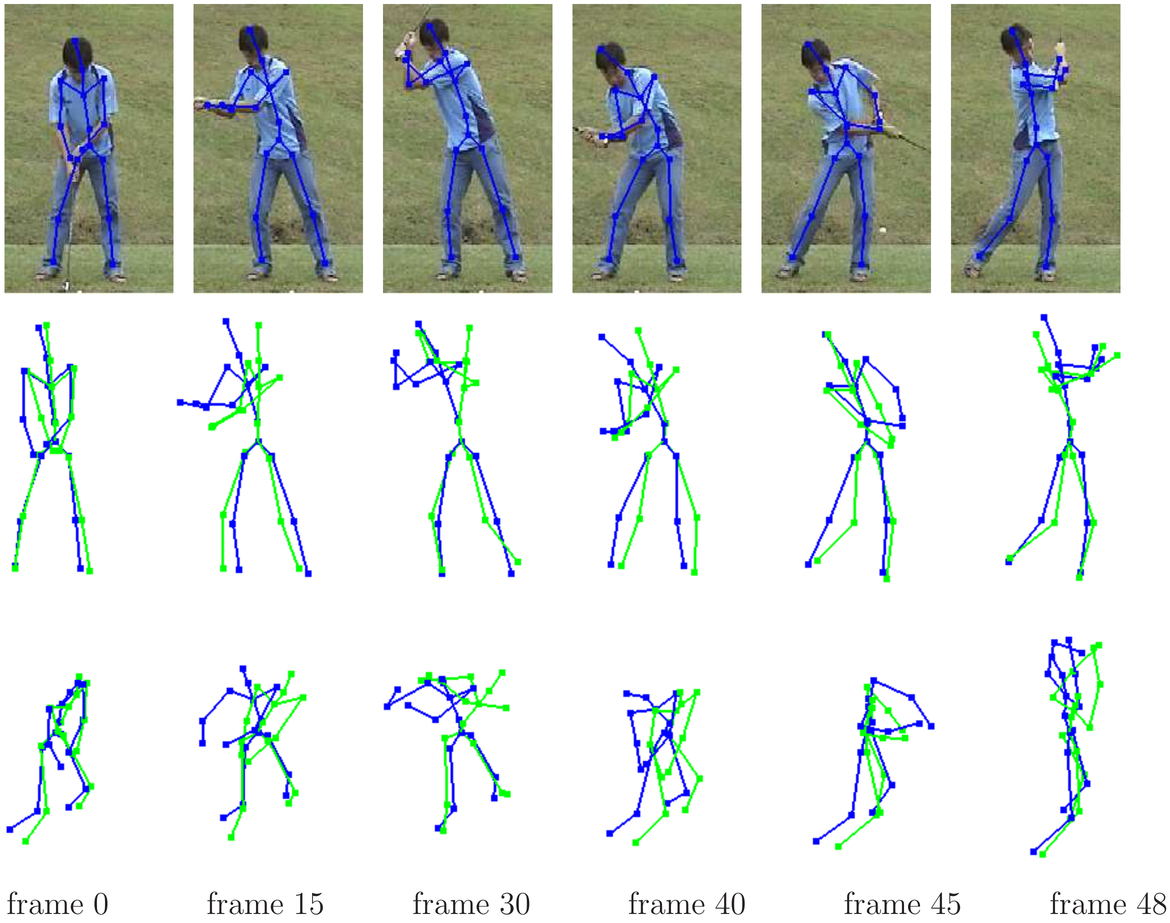


Figure 5.30: Selected posture candidates for the golf swing. First row: input images with selected posture candidates projected. Second and third rows: selected posture candidates (blue skeleton) overlapped with the corresponding reference postures (green skeleton) in the frontal and oblique views.

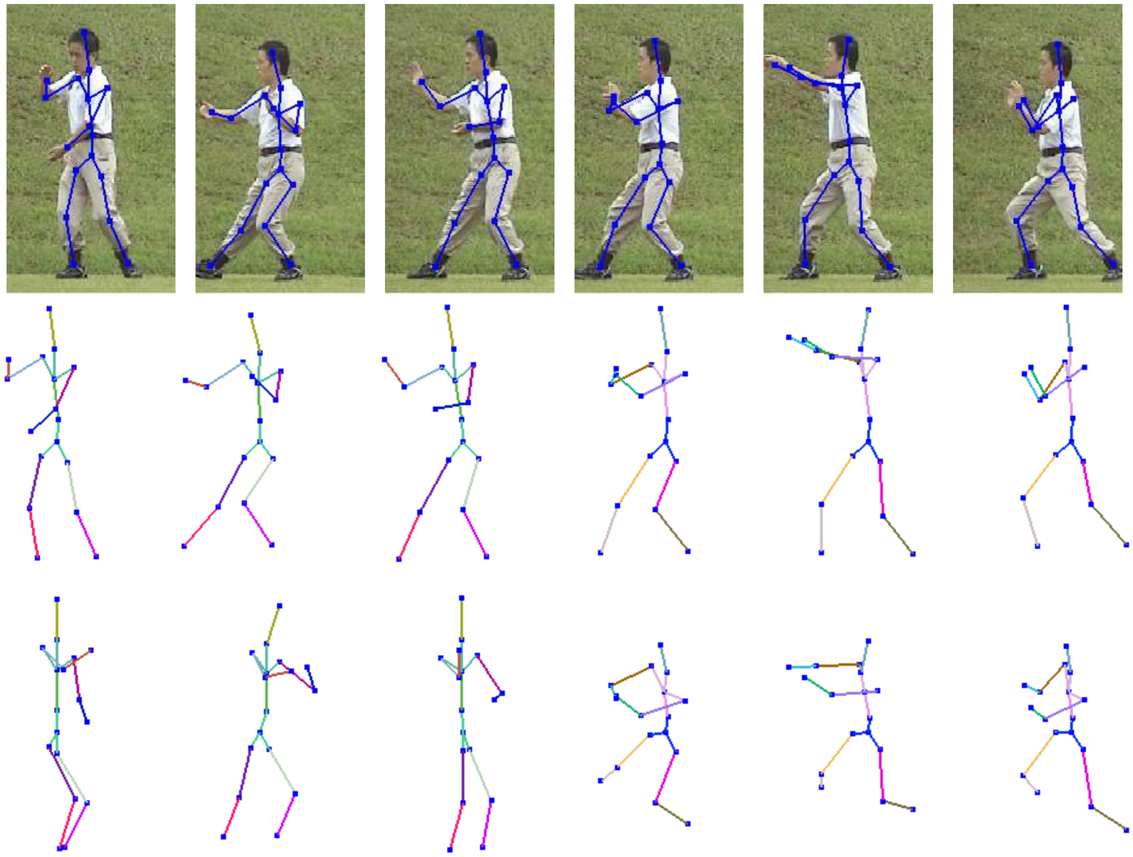


Figure 5.31: Posture estimation under ambiguous conditions with small algorithmic error. First row: input images with posture candidates projected. Second row: frontal views of the selected posture candidates. First three image in the third row: side views of the selected posture candidates. Last three images in the third row: frontal oblique views of the selected posture candidates. Each bone is in displayed in a unique color. Depth ambiguity and left-right ambiguity of legs occur in all the images. Partial and total self-occlusion of the right arm exist in the last three images.

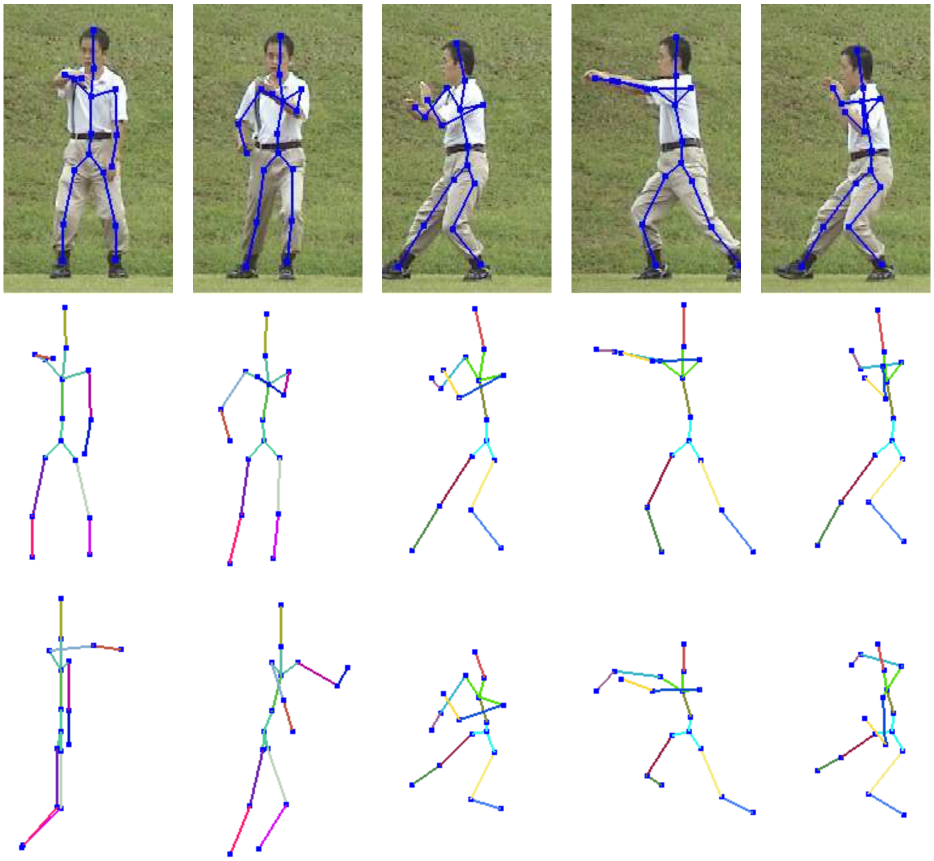


Figure 5.32: Posture estimation under ambiguous conditions with larger algorithmic error. First row: input images with posture candidates projected. Second row: frontal views of the selected posture candidates. First two images in the third row: side views of the selected posture candidates. Last three images in the third row: frontal oblique views of the selected posture candidates.

Chapter 6

Future Work

The motion analysis framework described in this thesis can be extended in several ways to adapt to a wider range of applications.

6.1 Perspective Camera Model

Perspective camera model can be used to improve the accuracy of camera projection. More accurate projection of human model should improve the accuracy of the algorithms that generate pose samples of body parts and posture candidates to match the input body regions. Well-known methods like Zhang's method [Zha00] or LaRose method [LaR98] can be used for camera calibration using a calibration object (e.g., chess board).

When perspective camera model is used, more accurate human body model for the performer's body should be constructed. Otherwise, the accuracy of the algorithm will be limited by the approximate human model.

6.2 Multiple Cameras

Multiple cameras can be used to efficiently resolve depth ambiguity by recording the performer's motion from multiple views. It can reduce the error in posture estimation and therefore improve the accuracy of proposed framework.

There are two methods of using multiple views to resolve the depth ambiguity during posture estimation. The first method is to project each pose sample to the multiple camera views, and measure the difference between the projected body regions and the input body regions in multiple views. Then the correct pose sample is the one that minimizes the differences in all the views. If enough cameras are used, this method will be able to determine a unique pose sample for each body part. Therefore, a unique posture candidate can be determined and posture candidate selection is not required. The shortcoming of this method is that each body part needs to be projected to all the views at each iteration of the BP algorithm. So it is not possible to process the different views in parallel, resulting in much longer total computation time.

The second method is to first estimate pose samples of body parts in each camera view independently, and generate a unique 3D posture from the estimated pose samples of all the views. Then the processing of the different views can be performed in parallel, shortening the total computation time. However, some body parts may be occluded in one or more views. In this case, robust algorithm needs to be used to find the body parts that are not occluded and use them for generating the 3D posture of the human body.

Note that multiple camera calibration has to be done in advance before recording the performer's motion. In general, the calibration can be accurately performed using a calibration object.

6.3 Uncertain Beginning and End of Input Video

The current implementation of the framework assumes that the input video and the reference motion begin and end at the corresponding frames, i.e. $C(0) = 0$ and $C(L') = L$. In general, this condition may not be satisfied. In this case, dynamic programming can be applied in Stage 2 of the framework to determine the best correspondence within a window for the beginning and the end of the input video.

6.4 Sub-pixel Algorithm

The current implementation of the posture estimation algorithm incurs a mean error of 1 pixel in 2D projected joint position. This error can be reduced by using sub-pixel algorithm. That is, when the BP algorithm is converging, switch to gradient descent algorithm to determine the pose samples. In this way, the 2D projected joint positions can be localized to sub-pixel accuracy.

6.5 Total Occlusion of Body Parts

When some body parts are totally occluded in the input images, their poses are unknown. In the current algorithm, their poses are determined based on the corresponding reference posture and the estimated posture candidates in the previous frame. When the algorithm converges, the estimated pose sample may be quite far from the ground truth. This problem can be controlled in the following manner. The algorithm can check whether a body part is occluded when it is projected to 2D image plane. If it is occluded, then the pose sample can be replaced by the one in the reference posture. In this case, the projected 2D joint position error will not be arbitrarily large.

6.6 Missing and Extraneous Motion Segments

In practise, the user may forget to perform a motion segment or repeat some motion segments incorrectly. In this case, to obtain an optimal temporal correspondence between the performer's motion and the reference motion, the missing or extraneous motion segments should be determined. Possible way for determining such segments is to find all the segment boundary candidates in the performer's motion, and then use dynamic programming to determine the correct correspondence between the performer's segment boundary candidates and the reference segment boundaries.

6.7 Very Large Performer's Error

When the performer's posture is very different from the corresponding reference posture, the reference posture cannot provide a good initial estimate for posture estimation. In this case, the estimated posture candidates in the previous frame can be used as the initial estimates.

If the performer's posture is found to be very different from the corresponding reference posture, the detailed posture error need not be measured. Instead, an overall large error feedback can be provided to the performer to indicate that his posture is very different from the reference posture.

6.8 Domain-specific Posture Error

The objective is to map the computed posture error to domain-specific error based on the domain-specific knowledge so that the feedback to the performance is more useful and direct in improving his motion. For example, the torso should be upright in most Taichi postures. So, a small error in torso orientation is considered as a major error by the domain-specific criteria. On the other hand, some posture errors are not important for computing the domain-specific error. For example, in Taichi, the knee's joint angle is allowed to vary according to whether the performer is practicing "high stance" or "low stance".

6.9 Hardware Acceleration

To reduce the the algorithms' computation time, hardware acceleration can be used to speed up the process. For example, the projection and rendering of human model, which is the most time-consuming part in the posture estimation algorithm, can be performed in GPU instead of CPU. In addition, the images in the video can be allocated to different CPUs to be processed in parallel, thereby reducing the overall throughput time.

6.10 Intuitive Visualization of Results

The current algorithms compute detailed errors between the orientations of the performer's body parts and the reference body parts. These errors need to be visualized in an intuitive manner for general users to easily understand the errors. For example, color code can be used to denote different amount of errors of different body parts. Animations of body parts can be used to illustrate to the performer how to adjust his posture to match the reference posture.

Chapter 7

Conclusion

The goal of this thesis is to develop an affordable and intelligent sports coaching system for general users that can automatically compare the performer's motion in a single video with an expert's 3D reference motion. To our best knowledge, this is the first attempt at automatic intelligent computer analysis of sports motion. In this thesis, we propose a new and fundamental problem for sports motion analysis: 3D-2D spatiotemporal motion registration. The proposed problem is by nature very complex due to the characteristics of the inputs and the outputs. All the complexities of the inputs and the outputs have been captured in the problem formulation.

Since it is infeasible to directly solve such a complex problem, this thesis presents a framework that decomposes the problem into four subproblems. The fist subproblem is to determine the camera projection parameters using the first reference posture and the first input image. This is a low-dimensional problem and the camera projection is determined only once at the beginning. The second subproblem is to determine the approximate temporal correspondence between the 3D reference motion and the performer's motion in the single video. This is a low-dimensional problem with long time sequence, which can be more easily solved compared to the proposed problem.

The third subproblem is to estimate the posture candidates for each input image. Given a single camera, there can be occlusion between body parts and depth ambiguity in the input image. Therefore, there are potentially multiple posture candidates that match the same input body region in the image. As a result, a set of posture candidates are estimated for each input image. Posture candidate estimation is a high-dimensional problem, but it is formulated for each image frame independently. The last subproblem is to select the best posture candidate for each input image and refine the temporal correspondence between the selected candidate sequence and the reference motion. Since the posture error between each posture candidate and each reference posture can be directly computed, this is a low-dimensional problem with long time sequence. It can be further decomposed into several low-dimensional problems with short time sequence using the segment boundary property. After posture candidate selection and temporal correspondence refinement, the posture error of each performer's posture can then be directly computed between the selected posture candidate and the corresponding reference posture.

For each subproblem, an algorithm is developed to solve it accurately. A simple algorithm is used for calibrating a scaled orthographic camera. A dynamic programming algorithm is developed to determine the approximate temporal correspondence between the 3D reference motion and the single video. The DP algorithm is efficient and accurate because it can find the optimal solution in a narrow band along the diagonal of the correspondence matrix.

According to the approximate temporal correspondence, the corresponding reference posture can be used as an initial posture estimate for posture candidate estimation from each input image. A nonparametric implementation of Belief Propagation is developed to estimate the pose of each body part. The BP algorithm decomposes the high-dimensional posture estimation problem into a set of low-dimensional pose estimation problems for the body parts. After the pose of each body part is estimated by the BP algorithm, a small number of posture candidates is generated by the posture candidate generation algorithm based on the pose estimate of each body part and the corresponding reference posture. The nonparametric implementation of BP can handle partial self-occlusion of body parts.

After multiple posture candidates are generated for each input body region in the input image, the performer's segment boundaries in the performer's motion are determined by the segment boundary estimation algorithm based on the segment boundary property. Then, for each motion segment, an efficient dynamic programming algorithm is developed to simultaneously select the best posture candidate for each t' and determine the optimal temporal correspondence between the reference motion and the performer's motion. After selecting the posture candidate for each t' and determining the optimal temporal correspondence, the performer's posture error at each t' is directly computed by the difference measurement between the selected posture candidate and the corresponding reference posture.

A comprehensive set of experiments is performed to evaluate the performance of the main algorithms. Test results for the estimation of approximate temporal correspondence show that a small window size and bandwidth are enough for finding the optimal solution. With these settings, significant amount of computation time is saved compared to the whole correspondence matrix. Experiments on posture estimation from synthetic images indicate that for most input images, the mean error of projected joint positions is about 1 pixel, and the maximum error is about 2 to 3 pixels. For the input images with total self-occlusions, the maximum errors are relatively large, i.e., about 10 to 50 pixels. The experiments also reveal that the mean posture error 7°. Larger posture errors happen in the input images with total occlusion for some body parts. For the other input images, the posture error are mainly due to rotation of body parts in depth. A mean error of 7° is reasonable and acceptable for a posture estimation algorithm using a single camera view. The experiments on real Taichi sequence and golf swing motion again show that the best posture candidate in each candidate set are very similar to the actual performer's posture. The depth orientations of the body parts in the best candidate are the same as those in the performer's actual posture.

From the experiments on estimating performer's segment boundaries, we find that the estimates differ from the ground truth by at most two frames, which is reasonably small in an input video of 339 frames. Accurate segment boundary estimation make the temporal correspondence between the two motion more precise. Test results show that the computed errors are significantly larger than the expected algorithmic error. This indicates that there is high confidence that the computed errors indeed reflect the performer's error. Therefore, the computed posture errors can be used for the coach or the performer to adjust the performer postures in sports training. In addition, the algorithm can select the correct posture candidates even under left-right ambiguity and partial self-occlusion of body parts. In the case of total self-occlusion, the algorithm can often infer the pose of the occluded body part if the performer's postures do not differ greatly from the reference postures.

Some enhancements to the basic framework are discussed in the Future Work section. It includes extensions to cater to more general application scenarios, enhancements of the algorithms' accuracies, and shortening of computation time by hardware acceleration. The enhancements would make the system feasible for real practical applications.

Appendix A

Joint Angle Limits

A human body consists of a set of body parts, and each part can be rotated around its parent joint in a local coordinate system (Figure 2.1(d)). Based on the local coordinate systems, the degree of freedom (DOF) of each joint and the valid range of joint angles between connecting body parts are listed in Table A.1. Joint angle limit is measured in terms of the possible difference between 3D orientation of the body parts connected at a joint.

Number	Name	Type	DOF	Range of joint angle (degrees)
0	Hip	Root	3	N/A
1	Left Hip	Joint	3	[20, 180]
2	Left Knee	Joint	1	[35, 180]
3	Left Ankle	End Effector	0	NA
4	Right Hip	Joint	3	[20, 180]
5	Right Knee	Joint	1	[35, 180]
6	Right Ankle	End Effector	0	NA
7	Lower Chest	Joint	2	[120, 180]
8	Upper Chest	Joint	3	[135, 180]
9	Left Shoulder	Joint	3	[0, 180]
10	Left Elbow	Joint	1	[40, 180]
11	Left Wrist	End Effector	0	NA
12	Right Shoulder	Joint	3	[0, 180]
13	Right Elbow	Joint	1	[40, 180]
14	Right Wrist	End Effector	0	NA
15	Head	Joint	3	[100, 180]
16	Head Tip	End Effector	0	NA

Table A.1: The DOF of each body joint and the valid range of joint angles between connecting body parts [RB02, NAS95].

Bibliography

[AASK04] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. Boostmap: A method for efficient approximate similarity rankings. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 268–275, 2004.

[aF98] M. Leventon andW. Freeman. Bayesian estimation of 3-d humanmotion from an image sequence. Technical Report TR-98-06, MERL, 1998.

[AS00] V. Athitsos and S. Sclaroff. Inferring body pose without tracking body parts. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 721–727, 2000.

[AS03] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[AT04] A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 882–888, 2004.

[BJ01] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *Proceedings of IEEE International Conference on Computer Vision*, pages 105–112, 2001.

[BM98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 8–15, 1998.

[Bra99] M. Brand. Shadow puppetry. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1237–1244, 1999.

[Bro98] E. Brookner. *Tracking and Kalman Filtering Made Easy.* John Viley & Sons, 1998.

[CI00] Y. Caspi and M. Irani. A step towards sequence-to-sequence alignment. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 682–689, 2000.

[CI01] Y. Capsi and M. Irani. Alignment of non-overlapping sequences. In *Proceedings of IEEE International Conference on Computer Vision*, pages 76–83, 2001.

[CI02] Y. Capsi and M. Irani. Spatio-temporal alignment of sequences. *IEEE Trans. on Pattern Analysis and Medical Intelligence*, 24(11):1409–1424, 2002.

[CR99] T.J. Cham and J.M. Rehg. A multiple hypothesis approach to figure tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[CSI02] Y. Caspi, D. Simakov, and M. Irani. Feature-based sequence-to-sequence matching. In *VAMODS workshop with ECCV*, 2002.

[DB98] J. W. Davis and A. F. Bobick. Virtual pat: A virtual personal aerobics trainer. In *Proceedings of Workshop on Perceptual User Interfaces (PUI'98)*, pages 13–18, 1998.

[DBR00] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 126–133, 2000.

[DCR01] D.E. Difranco, T.J. Cham, and J.M. Rehg. Recovery of 3-D figure motion from 2-D correspondences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[EBMM03] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proceedings of IEEE International Conference on Computer Vision*, pages 726–733, 2003.

[EL04] A. Elgammal and C.S. Lee. Inferring 3D body pose from silhouettes using activity manifold learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 681–688, 2004.

[FH05] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int. Journal of Computer Vision*, 61(1):55–79, 2005.

[FL95] C. Faloutsos and K.I. Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *ACM SIGMOD*, pages 163–174, 1995.

[FP03] D.A. Forsyth and J. Ponce. Tracking with non-linear dynamic models. One chapter excluded from "Computer Vision: A Modern Approach", 2003.

[Gle98] M. Gleicher. Retargeting motion to new characters. In *ACM SIGGRAPH*, pages 33–42, 1998.

[GP99] M. Giese and T. Poggio. Synthesis and recognition of biological motion patterns based on linear superposition of prototypical motion sequences. In *Proceedings of IEEE Workshop on Multi-view Modeling and Analysis of Visual Science*, 1999.

[HLF99] N.R. Howe, M.E. Leventon, and W.T. Freeman. Bayesian reconstruction of 3D human motion from single-camera video. In *Neural Information Processing Systems*, 1999.

[How04] N.R. Howe. Silhouette lookup for automatic pose tracking. In *Computer Vision and Pattern Recognition Workshop, 2004 Conference on*, pages 15–22, 27-02 June 2004.

[HS03] G.R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):530–549, 2003.

[HW04] G. Hua and Y. Wu. Multi-scale visual tracking by sequential belief propagation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 826–833, 2004.

[HYW05] G. Hua, M. H. Yang, and Y. Wu. Learning to estimate human pose with data driven belief propagation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 747–754, 2005.

[IB96] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of European Conference on Computer Vision*, pages 343–356, 1996.

[IF99] S. Ioffe and D. Forsyth. Finding people by sampling. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1092–1097, 1999.

[Isa03] M. Isard. Pampas: Real-valued graphical models for computer vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 613–620, 2003.

[JBY96] S. Ju, M. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated motion. In *Proceedings of IEEE Conference on Automatic Face and Gesture Recognition*, pages 38–44, 1996.

[Jor02] M.I. Jordan. *An Introduction to Probabilistic Graphical Models*. In preparation, 2002.

[JR02] M.J. Jones and J.M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46:81–96, 2002.

[KHM00] I.A. Karaulova, P.M. Hall, and A.D. Marshall. A hierarchical models of dynamics for tracking people with a single video camera. In *British Machine Vision Conference*, 2000.

[Kol] Vladimir Kolmogorov. Source code of graph cut. http://www.cs.cornell.edu/ rdz/grachcuts.html.

[LaR98] D. LaRose. A fast, affordable system for augmented reality. Technical report, CMU-RI-TR-98-21, 1998.

[LC04] M. W. Lee and I. Cohen. Proposal maps driven MCMC for estimating human body pose in static images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 334–341, 2004.

[LRS00] L. Lee, R. Romano, and G. Stein. Monitoring activities from multiple video streams: Establishing a common coordinate frame. *IEEE Trans. on Pattern Analysis and Machine Intelligenece*, 22:758–767, August 2000.

[LYST06] R. Li, M.H. Yang, S. Sclaroff, and T.P. Tian. Monocular tracking of 3d human motion with a coordinated mixture of factor analyzers. In *Proceedings of European Conference on Computer Vision*, pages 137–150, 2006.

[May] Autodesk Maya. Integrated 3d modeling, animation, effects and rendering. http://usa.autodesk.com.

[MG01] T.B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[MHK06] T.B. Moeslund, A. Hilton, and V. Kruger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, 2006.

[MM02] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *Proceedings of European Conference on Computer Vision*, pages 666–680, 2002.

[MOB05] A. Micilotta, E. Ong, and R. Bowden. Detection and tracking of humans by probabilistic body part assembly. In *British Machine Vision Conference*, 2005.

[Mor05] G. Mori. Guiding model search using segmentation. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1417–1423, 2005.

[Mota] Sports Motion. 2D video-based motion analysis system. http://www.sports-motion.com.

[Motb] MotionCoach. Golf swing analysis. http://www.motioncoach.com.

[MRR80] C. Myers, L. Rabinier, and A. Rosenberg. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 28(6):623–635, 1980.

[MSZ04] K. Mikolajczyk, D. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *European Conference on Computer Vision*, 2004.

[NAS95] NASA. Man-systems integration standards. Technical Report NASA-STD-3000, NASA Johnson Space Center, Houston, Texas, 1995.

[Pro] V1 Pro. Golf swing analysis software. http://www.ifrontiers.com.

[RAS01] R. Rosales, V. Athitsos, and S. Sclaroff. 3D hand pose reconstruction using specialized mappings. In *Proceedings of IEEE International Conference on Computer Vision*, pages 378–385, 2001.

[RB02] Nancy Berryman Reese and William D. Bandy. *Joint range of motion and muscle length testing*. Philadelphia : Saunders, 2002.

[RBM05] X. Ren, A.C. Berg, and J. Malik. Recovering human body configurations using pairwise constraints between parts. In *IEEE International Conference on Computer Vision*, volume 1, pages 824–831, 2005.

[RFZ05] D. Ramanan, D. A. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 271–278, 2005.

[RGSM03] C. Rao, A. Gritai, M. Shah, and T.S. Mahmood. View-invariant alignment and matching of video sequences. In *Proceedings of IEEE International Conference on Computer Vision*, pages 939–945, 2003.

[RK95] J. Rehg and T. Kanade. Model-based tracking of self occluding articulated objects. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 612–617, 1995.

[RKB04] C. Rother, V. Kolmogorov, and A. Blake. Grabcut - interactive foreground extraction using iterated graph cuts. In *Proc. ACM Siggraph*, 2004.

[RMR04] T.J. Roberts, S.J. McKenna, and I.W. Ricketts. Human pose estimation using learnt probabilistic region similarities and partial configurations. In *European Conference on Computer Vision*, 2004.

[RS00a] R. Rosales and S. Sclaroff. Specialized mappings and the estimation of human body pose from a single image. In *Workshop on Human Motion*, pages 19–24, 2000.

[RS00b] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[RS06] R. Rosales and S. Sclaroff. Combining generative and discriminative models in a framework for articulated pose estimation. *International Journal of Computer Vision*, 67(3):251–276, 2006.

[RST02] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *European Conference on Computer Vision*, 2002.

[SB01] H. Sidenbladh and M.J. Black. Learning image statistics for bayesian tracking. In *Proceedings of IEEE International Conference on Computer Vision*, pages 709–716, 2001.

[SB03] H. Sidenbladh and M.J. Black. Learning the statistics of people in images and video. *International Journal of Computer Vision*, 54:183–209, 2003.

[SBF00a] H. Sidenbladh, M. Black, and D. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *Proceedings of European Conference on Computer Vision*, pages 702–718, 2000.

[SBF00b] H. Sidenbladh, M.J. Black, and D.J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conference on Computer Vision*, 2000.

[SBR$^+$04] L. Sigal, S. Bhatia, S. Roth, M.J. Black, and M. Isard. Tracking loose-limbed people. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 421–428, 2004.

[SBS02] H. Sidenbladh, M.J. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In *European Conference on Computer Vision*, 2002.

[SG98] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[SIFW03] E.B. Sudderth, A.T. Ihler, W.T. Freeman, and A.S. Willsky. Nonparametric belief propagation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 605–612, 2003.

[Sima] Simi. 3d motion tracking system. http://www.simi.com.

[Simb] Simi. Video based motion analysis. http://www.simi.com.

[SMFW04] E.B. Sudderth, M.I. Mandel, W.T. Freeman, and A.S. Willsky. Visual hand tracking using nonparametric belief propagation. In *IEEE CVPR Workshop on Generative Model based Vision*, 2004.

[ST01] C. Sminchisescu and B. Triggs. Covariance scaled sampling for monocular 3D body tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 447–454, 2001.

[ST03] C. Sminchisescu and B. Triggs. Kinematic jump processes for monocular 3D human tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 69–76, 2003.

[Ste98] G.P. Stein. Tracking from multiple view points: Self-calibration of space and time. *In DARPA IU Workshop*, pages 521–527, 1998.

[SVD03] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proceedings of IEEE International Conference on Computer Vision*, pages 750–757, 2003.

[TdSL00] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[Tip00] M. Tipping. The relevance vector machine. In *Neural Information Processing Systems*, 2000.

[TNS$^+$06] A. Thayananthan, R. Navaratnam, B. Stenger, P. H. S. Torr, and R. Cipolla. Multivariate relevance vector machines for tracking. In *Proceedings of European Conference on Computer Vision*, 2006.

[UFF06] R. Urtasun, D. J. Fleet, and P. Fua. 3D people tracking with gaussian process dynamical models. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 238–245, 2006.

[UFHF05] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *Proceedings of IEEE International Conference on Computer Vision*, pages 403–410, 2005.

[Vic] Vicon. Optical motion capture system. http://www.vicon.com.

[WL05] R. Wang and W. K. Leow. Human body posture refinement by nonparametric belief propagation. In *Proceedings of IEEE International Conference on Image Processing*, 2005.

[WN99] S. Wachter and H. Nagel. Tracking persons in monocular image sequences. *Computer Vision and Image Understanding*, 74(3):174–192, 1999.

[YFW02] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. Technical report, MERL, 2002.

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, November 2000.