Davide Moschini

Tracking Human Motion With Multiple Cameras Using Articulated ICP With Hard Constraints

Ph.D. Thesis

15th March 2009

Università degli Studi di Verona
Dipartimento di Informatica

Advisor:
prof. Andrea Fusiello

Università di Verona
Dipartimento di Informatica
Strada le Grazie 15, 37134 Verona
Italy

Contents

1 Introduction 1
1.1 Motion Capture History 1
1.2 Motion Capture Categories 3
1.2.1 Mechanical Systems 3
1.2.2 Electromagnetic Systems 3
1.2.3 Optical Systems 3
1.3 Motivations 4
1.4 System Overview 4
1.5 Original contributions 7
1.6 Organization of the thesis 9

2 Related Work 11
2.1 Classification by Initialization 12
2.2 Classification by Markers 13
2.2.1 Marked Systems 13
2.2.2 Marker-less Systems 14
2.3 Classification by Tracking Strategy 14
2.4 Classification by Model 14
2.5 Conclusion 15

3 Volume Reconstruction and Skeletonization 17
3.1 Shape from Silhouette 17
3.1.1 Introducing the reconstruction algorithm 17
3.1.2 Implementation 18
3.1.3 The algorithm 19
3.1.4 Reconstruction of the Volume 20
3.2 Skeletonization 22

4 Hierarchical Articulated ICP and Human Body Model 27
4.1 Human Body Model 27
4.1.1 H-Anim Body Model 27
4.1.2 Our Body Model 28
4.2 Weighted Extended Orthogonal Procrustes (WEOP) 29
4.3 Hierarchical Articulated ICP 31

5 Tracking 37
5.1 Direct and Inverse Body Kinematic 37
5.1.1 Twist rotation representation 37
5.1.2 Kinematic Chains 39
5.1.3 Kalman Filter 45
5.2 The Tracker 49

6 System Implementation 53
6.1 Camera Calibration 53
6.2 Frame Capture 53
6.3 Background subtraction 55
6.4 3D Volume Reconstruction, ICP and Tracking 56

7 Experimental Results 59
7.1 Experiments on the HumanEva dataset 59
7.2 Evaluation Metrics 63
7.3 Conclusion 64

8 Conclusions 77
8.1 Publications 78

Part II Articles

References 89

List of Figures

1.1 Two of the famous Muybridge's sequences: (a) Galloping Horse, 1878 - (b) Animal Locomotion Plate 99 , 1887 2
1.2 One of the interesting Marey's picture. Note the motion suite markers used to study the movements 2
1.3 The System Main Components. The first part uses multiple frames. Model fitting is done using the modified ICP algorithm. . 6
1.4 Mocap Phases. 8

2.1 Some of the body models you can find in literature. 15

3.1 The foreground/background subtraction phase using a virtual room created using Blender. 20
3.2 The Vitruvian or T-Pose of a body. 21
3.3 Rendering of 30 horizontal slices. 23
3.4 Principal axis of a blob. 23
3.5 a) Silhouettes; b) projection onto the sweeping plane; c) intersection (slice) d) final volumetric reconstruction 25
3.6 a) slices along Z; b) slices along X and Y; c) centerline points with the stick figure overlaied. 26

4.1 The H-Anim body model. 28
4.2 The stick figure body model. 29
4.3 A 2-D example of the computation of the closest point. 32
4.4 An example showing that the closet segment is not necessarily the one belonging to the closet line. 34

5.1 The stick figure body model. 39
5.2 The fifteen reference points. The numbers inside the circle specify the limbs number. 47
5.3 The working schema of a Kalman filter 49

6.1 The checkboard used for the intrinsic camera parameters. 54
6.2 Camera setup for the architecture. The camera number zero is pointing to the floor. All the camera are Basler model f602c 55
6.3 The calibration checkboard used for the extrinsic camera parameters computation. 56
6.4 The schema of the architecture. 57
6.5 Background subtraction process. 58

7.1 A graphical summary of the algorithm pipeline. 60
7.2 Sample frames of *"S2 Jog"* and silhouettes. 61
7.3 Sample frames of *"S2 Walk"* and silhouettes. 62
7.4 Plots comparing ground truth and estimated joint angles of the torso, knee and elbow in two of the sequences used in the experiments. 65
7.5 Plots comparing ground truth and estimated joint angles of the Right Elbow, Right Knee and the Neck used in the experiments. . 66
7.6 Plots comparing ground truth and estimated joint angles of the Left and Right Shoulder and Left Hip 67
7.7 Plots comparing ground truth and estimated joint angles of the Right Hip. 68
7.8 Frames taken from the sequences with the stick skeleton model on them. 70
7.9 Plots of the position error on two of the sequences used in the experiments. 71
7.10 Plots of the position error on two of the sequences used in the experiments. 72
7.11 Plots of the position error on two of the sequences used in the experiments. 73
7.12 Plots of the position error on two of the sequences used in the experiments. 74
7.13 Plots of the position error on two of the sequences used in the experiments. 75

List of Tables

5.1 Range of the valid values for each body model joint angles. The third column of the table specify the axis of rotation of the angles. 41
5.2 Direct Kinematic. 45
5.3 **t** vector values 46
5.4 The Inverse Kinematics formulae used in the experiments. 48
5.5 The prediction formulae used in the kalman filter first step. 48
5.6 The correction formulae used in the kalman filter second step. 49
5.7 The correction formulae used in the Extended Kalman filter second step. 50
5.8 The Jacobian structure. 51

7.1 RMSE of the angular value expressed in radians. 68
7.2 RMSE of the position value expressed in mm. 69

1 Introduction

The work presented in this thesis is about marker-less human motion capture. Motion capture is the recording of human body movement (or other movement) for immediate or delayed analysis and playback. The information captured can be as general as the simple position of the body in space or as complex as the deformations of the face and muscle masses. Motion capture is also called motion tracking, or using an abbreviation, *mocap*.

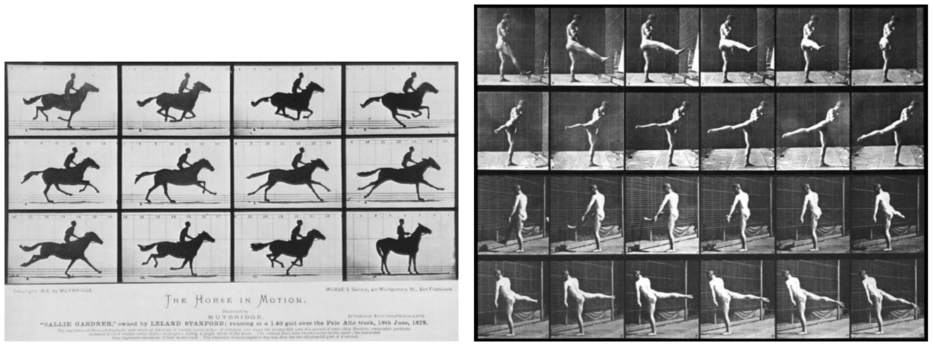
1.1 Motion Capture History

The first attempt to capture the motion of human bodies and animals was done by Eadweard Muybridge. He was an English photographer interested in studying people and animals movements. The technique used by Muybridge to capture the famous sequence of the running horse was to use cameras activated by wires hit by horse's hoofs (see Figure 1.1).

Contemporary of Muybridge was Étienne-Jules Marey a French scientist. He developed a chronophotographic gun in 1882 capable of taking twelve consecutive frames per second. All the frames were recorded in one single picture. He used his technique to study animals and in particular bird motion. He was the first person to analyze human and animal motion using video with markers (see Figure 1.2).

One of the first application of motion capture technique in animation date back to 1914. Max Fleisher in that year created the rotoscope technique for use in animated films. This technique consist of tracing a drawing over live action film movement, frame by frame, to achieve realistic animation. The technique was used in numerous Disney animated films as *Snow White* and *Cinderella*.

In the eighties Tom Calvert, a professor of kinesiology and computer science at Simon Fraser University, attached potentiometers to a body and used the



(a) (b)

Fig. 1.1. Two of the famous Muybridge's sequences: (a) Galloping Horse, 1878 - (b) Animal Locomotion Plate 99 , 1887

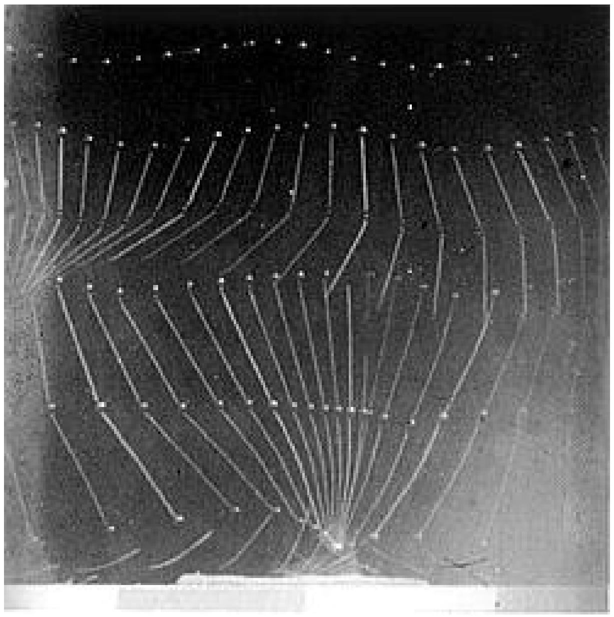


Fig. 1.2. One of the interesting Marey's picture. Note the motion suite markers used to study the movements

output to drive computer animated figures for choreographic studies and clinical assessment of movement abnormalities.

In 1983 Ginsberg and Maxwell at MIT presented a system for motion capture called Op-Eye that relied on sequenced LEDs. They build a body suit with LEDs on the joints and other anatomical landmarks. Two cameras with special photo detectors returned the 2-D position of each LED in their fields of view. The computer then used the position information from the two cameras to obtain a 3-D world representation of each LED.

In the last years many commercial motion and tracking systems have been developed for animation, medical and military purpose ([2, 17], see also the Related Work Chapter ahead).

1.2 Motion Capture Categories

Over the years, mocap has taken many forms, each with its own strengths and weaknesses. In this section a summary of three of the main mocap categories is shown. It is possible to subdivide the motion capture systems in:

- Mechanical Systems
- Electromagnetic Systems
- Optical Systems

1.2.1 Mechanical Systems

In this kind of systems the people involved in the procedure of motion capture wear a set of straight metal pieces fastened on the back of the performer. These pieces are forced to move with the person and thus sensors placed on them can read the pose. Examples of such a mechanical systems are gloves and mechanical arms. These kind of systems are unaffected by change in lights but some kind of movements cannot be tracked. Animation like jumping for example cannot be tracked.

1.2.2 Electromagnetic Systems

In this case the performer wears magnetic receivers that can be used to track location respect to a static transmitter. With these systems the position and rotation of the joints are measured absolutely, i.e. respect to a world reference frame. There still be some problems with magnetic distortion as distance from the transmitter increases. Performer need to be connected in some ways to the control station.

1.2.3 Optical Systems

Usually the performer wears reflective parts that are followed by several cameras and all the information capture from the reflective part is then triangulated.

The markers can be also emitting; in this case the performer need to be wired and the movements can be limited. In the last years systems using no markers have been developed from many researchers working in the mocap field giving even more flexibility to the types of motion and location to be used for the tracking [9, 11, 38, 40, 55, 56, 59, 64]. Using these kind of systems it is possible to use larger volume and the performers can also do all the possible movements. There could be problems with changing in lights.

1.3 Motivations

In recent years, a lot of effort was put in developing new approaches for tracking human beings without using markers during the process. The motivations to move in this direction are several but the most important are the relatively low cost of marker-less systems with respect to the others and the possibility to track in all the movements of a human body. In these systems, in fact, it is possible to use cameras as sensors and computer using consumer level graphics cards and hardware.

Another characteristic of such a system should be the real-time acquisition of data. This can be useful in applications where immediate feedback is needed and can also improve the performance in the tracking phase because of the smaller interval between each frame.

The accuracy of the pose is another requirement of a motion capture systems. Either the measured angular position of the joints of a human being and the position of the extremities must be as precise as possible. In medical and sport applications this is the most important characteristic.

The system should be simple to use for the people driving it but also for the performers. Thus the tracking procedures and the absence of markers on the body are requirements to be fulfilled. Imagine, for example, an actor that has to wear a suit and then stick on it all the sensors, and then wait for the calibration of the sensor on his body. And this has to be done for each capturing sessions. The procedure is time consuming. A system where an actor come without any suits and wearing costumes for a particular scene can be comfortable and convenient.

1.4 System Overview

In this thesis a system for motion capture that use video streams from multiple cameras as input was developed. The system performs body acquisition, pose computation and tracking.

When we are taking into consideration a body motion capture system we have to face some tasks that are:

- acquisition and representation of the body
- model acquisition
- tracking

In the first task we have to choose if we want to work directly with $2D$ image data of the person or if we want a $3D$ reconstruction of the body. In this thesis we have chosen to reconstruct the $3D$ volume using voxels. The approach allows us to produce simple and robust algorithm.

The model acquisition task deals with the problem of converting the voxel representation to a math representation that allow us to compute body state and parameters.

Finally, because the person is not in a static position but is moving in the area of interest, we need a tracking algorithm to improve the measurements taken in each frame of the sequence.

As with traditional animation and many other arts, mocap is actually composed of a number of phases:

- System and room setup
- Calibration of the system
- Capturing
- Data arrangement
- Data post processing

In Figure 1.4 these phases are shown. Each mocap systems need to be placed in a room, the cameras need to be placed and pointed with the right orientation and optics, the acquisition instruments, like computers, need to be configured, lights set up to that they do not interfere with the acquisition procedure.

When the room setting is correct the system needs to be calibrated, i.e., the camera parameters and configuration estimated.

The third phase is crucial to the system and represents the acquisition of the data. When the system is running and calibrated a person can perform inside the acquisition area.

When we have all the data, or during the process of acquisition, usually a data arrangement procedure is performed to control some parameters such as noise or spiky points.

The last phase, called post processing, is used to apply improvements on the graphics of the animation or to analyze with slow off-line algorithm the results.

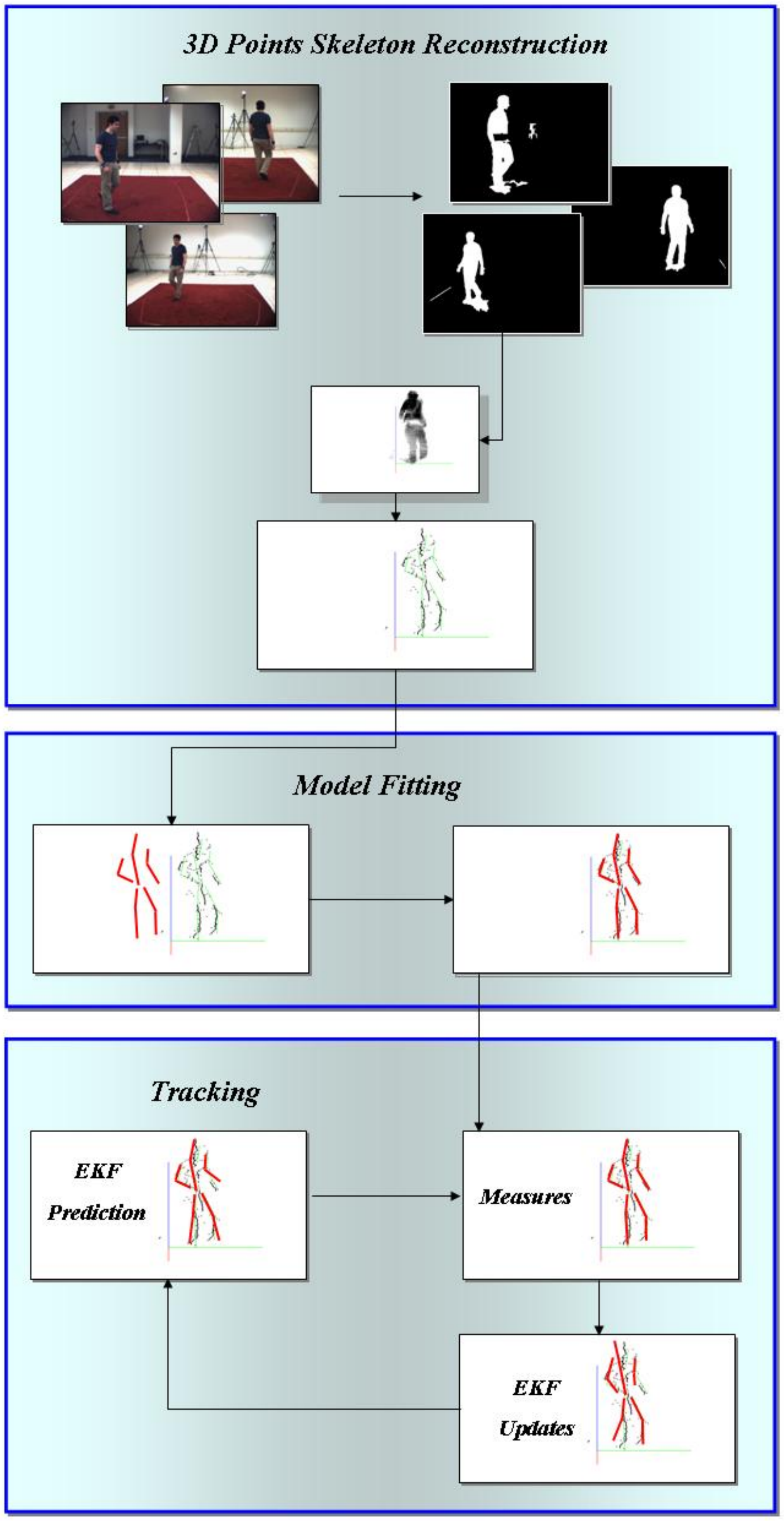


Fig. 1.3. The System Main Components. The first part uses multiple frames. Model fitting is done using the modified ICP algorithm.

The system flowchart is shown in Figure 1.3. In this Figure are represented only the main components of the mocap system: the 3d voxel reconstruction, the model fitting phase and the tracker.

The 3D voxel reconstruction takes multiple synchronized video streams and computes the reconstruction of the shape represented by silhouettes. These silhouettes are computed from each camera frame as shown in Figure 1.3. The system computes the volume of interest from the silhouettes in each of the camera views. Then, for each camera, the silhouette is projected in the 3D space and intersection on planes with other silhouettes is computed. Each intersection can be shown to be a blob on a plane and we can find for each blob the centroid. The centroid is used as 3D middle axis point and saved in a set.

Using the set of middle body axis points a 3D stick body model is fitted on them using the ICP modified technique. We assume that the model is rigid and that the time between each frame is small, thus the assumption is that we can start the ICP algorithm from a model that is not too far from the set of 3D points.

Finally, the tracking procedure executes the predict-update cycle at each frame. Measurements of locations of specific points on the body are extracted from the stick body model and an Extended Kalman filter is used to adjust the model position and configuration to best fit the extracted measurements.

1.5 Original contributions

In the last years many researchers worked on body tracking. Many of the solutions adopted in the literature by researchers are reported in Chapter 2.

The techniques we are interesting in are the one using optical system and marker-less trackers. One of the solution adopted to track people in these kind of systems is to reconstruct the volume of the body and then using some algorithm find the pose in each frame.

The contributions of our work lay in this third phase of the motion capture procedure. The first contribution is a new algorithm to find the skeletal points of a 3D volume. The algorithm using a slicing technique finds the medial axis of a volume in a fast way using the graphic card processor and the texture units (see Chapter 3). This algorithm produce good results in quality and performance compared to other works in literature.

The second contribution is the introduction of a new tracking strategy based on a hierarchical application of the ICP standard algorithm to find the match

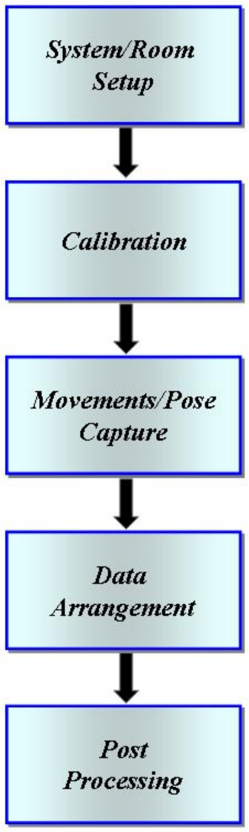


Fig. 1.4. Mocap Phases.

between a stick body model and a set of 3D points. The algorithm use a traversing version of ICP where all the 3D points are weighted so that every limbs of the model can best fit on the right portion of the body (see Chapter 4).

We completed the tracking procedure by integrating the classical Extended Kalman Filter algorithm with our previous results as measurements.

In our experiments, our technique shows performance comparable with other systems reported in literature.

1.6 Organization of the thesis

This thesis is divided in six chapters.

Chapter 1 contains a review of tracking strategies found in literature. A taxonomy of these technique is presented.

Chapter 2 regards the 3D volume reconstruction technique used in this work. In particular we introduce a new technique to find the middle axis point of an object. This technique is used to find the skeletal points of a body.

Chapter 3 faces the problem of fitting a body model to a set of 3D points. A new technique based on the ICP algorithm is introduced.

Chapter 4 shows the tracking algorithm used in the system. It is based on the Extended Kalman Filter technique. It takes as measurement input the body model fitted on the 3D volume reconstructed.

Chapter 5 explains how the system is implemented. Hardware and software used are reported in this chapter.

Chapter 6 concludes the work showing the results obtained applying the algorithms introduced. To measure the performance the HumanEva dataset was used as groundtruth.

2

Related Work

Motion capture is the problem of extracting the parameters of the model of a human body or some of its part from video data. The model is usually chosen a priori and the algorithm that estimates the model parts extracts the parameters such as joints, limbs, angles, dimensions and so on.

We can subdivide the motion capture domain in three fields:

- Face Analysis
- Non-facial Gesture Analysis
- Body Analysis.

Face analysis is concerned with face detection, face tracking, face recognition and facial expression recognition.

Gesture analysis is concerned with the representation of hands and arms and of their movements such as general gestures, manipulation gestures, full-body gestures (as dance), pointing gestures and sign language gestures.

The third field is the one we are interested in. In the field of body motion analysis, large body movements are studied. That is, we are interested in how people move around and what they are doing. There are several application areas, for example in animation movies to give more realistic movements to the virtual actors; in sports to analyze the motion of athletes; in medical studies to better understand the human musculoskeletal apparatus. In Human Computer Interaction (HCI) using the body as the interface with the machine. There are several studies on this last field of motion capture, for example, applications that count people [53, 72] or applications tracking humans walking around from one room to another one [56,66]. Another interesting field is precise human tracking, where exact posture of the limbs of human body is estimated, for example [60]. Finally the last area is recognition of human movements. In this field we want to recognize whether a human is walking or running or if it is doing other kind of movements [70].

There are several taxonomies we can use to classify the field of human body motion using computer vision. The methods proposed in previous works depend on: number of sensors, type of the sensors used, mobile sensors versus static sensor, model-based approach versus nonmodel-based approach, 2D versus 3D, pose estimation versus tracking, one person versus many, number of limbs in model, distributed system versus centralized system, rigid or non-rigid or elastic motion, etc.

Several surveys exist in literature covering the contribution in the field of human body pose estimation. Shah [78] gives an overview of different methods for motion extraction belonging to optical flow or motion correspondence. He gives a taxonomy of the human motion capture problem as: action recognition, body parts recognition and body configuration estimation.

Moeslund and Granum [58] give an overview of the taxonomy used by other authors and then define theirs taxonomy from a general system's point of view, i.e., any system for analyzing human body motion can be explained by four blocks: Initialization, Tracking, Pose estimation and Recognition. In 2006 Moeslund et al. published another survey [59] based on the first one and expanding it with works up to 2006.

Aggarwal and Cai [3] focus on three major areas related to interpreting human motion: motion analysis involving human body parts, tracking of human motion using single or multiple cameras, and recognizing human activities from image sequences.

Gavrila [32] identifies a number of applications and provides an overview of developments in human body motion. The various methodologies are grouped in 2D approaches with or without explicit shape models and 3D approaches. Across these three classes he describe the approach dealing with recognition.

In the next sections we will list work classified by the kind of initialization used, by the type of sensors used, by the type of tracking and by the human model.

2.1 Classification by Initialization

The systems in literature can be classified by the kind of initialization required by the system itself to start to work. Many of the systems require manual initialization of the human body model and then they perform the tracking procedure.

Some systems use an automatic procedure to acquire the body model and usually the person is required to perform a set of calibration movements or to stay freeze for some seconds in a determined pose.

In Yamamoto and Koshikawa [89] the operator of the system finds the initial pose of the user. Other systems use an automatic approach to find the initial body pose as in Caillette and Howard [20] with a statistical EM approach or in Rohr [71] where the author find the blob of person in the image and than initialized a cylinder shapes human model. In Ivekovic, Trucco and Petillot [40] the body model is automatically initialized using a particle swarm optimization technique.

2.2 Classification by Markers

Another kind of classification we can use for the human pose tracking system is by the type of markers used. In most of the works the sensors used are cameras. The classification thus regard type of markers used during the acquisition. The two main classes used also in literature for this classification are marker and marker-less systems.

2.2.1 Marked Systems

Most of existing approach for analyzing human movements assume the body of the person to be marked in some way (e.g.: Hoffman and Flinchbaugh [39], Seitz and Dyer [77], Webb and Aggarwal [86]). These kind of system use data capture from sensors to triangulate the 3D pose of a person. Usually the markers are applied directly to the body of the subject or velcroed to a suit worn by the subject. Some systems use retroreflective markers that reflect light back to the camera. In this way the cameras can ignore everything excepts the markers on the body.

Other systems use active markers to solve the problem of markers swap between frames. In passive markers systems there is always the problem that each marker is identical to another one and thus it is possible, during the tracking, to make identification mistakes. To solve these kind of problem it is possible to use active markers, such as LEDs, that turn on and off in a known sequences.

In both categories (active and passive markers) commercial systems already exist (i.e. Vicon System and Industrial Light & Magic [17]).

2.2.2 Marker-less Systems

In the last years the research was focused on this kind of tracking system. A marker-less system is usually less expensive. It has the drawback to be less accurate.

Works in this field are for example [11, 40, 55, 56]. Our work stays in this category as we are not going to use any kind of marker placed on the body.

2.3 Classification by Tracking Strategy

We can classify the works in literature by the tracking strategy (or the absence of a tracker) used to estimate the body pose. Mikic et al. [56] use a stick figure body model surrounded by cylinders and ellipsoids and a Kalman filter to track the pose on each video frame. Deutscher et al. [26] developed a modified particle filter for search in high dimensional configuration spaces. They use a skeleton combined with conical section and the modified particle filter. Balan et al. [9] use a triangulated mesh model estimating the parameters of the model from the images. Using this model they estimate the pose and shape of the body. Kehl and V. Gool [45] show a method for a markerless body tracking using multi camera views and image cues. Plankers et al. [69] fit an implicit surface body model to 3D stereo data and silhouettes contours on video sequences. Sminchisescu and Triggs [82] use a method for recovering the motion from monocular video sequences based on superquadric ellipsoids and a modified particle filter. Carranza et al. [21] use a triangular mesh body model and silhouettes extraction from synchronized video frames to reproduce the actor body from different virtual views. The papers of Ballan (Ballan et al. [10] and Ballan and Cortelazzo [11]) show an approach for markerless motion tracking of a skinned mesh using optical flow and silhouettes. Ivekovic, Trucco and Petillot [40] find the body pose using still images and Particle Swarm Optimisation. The technique can be used for the initialization of a tracker in the subsequent frames.

2.4 Classification by Model

There are a lot of models used to represent the human body (see Granieri and Badler [35] and Norman et. al [8]. Some system start using a general model which is an average of standard humans, see e.g. Baumberg and Hogg [13] or Rohr [71]. Other authors measure the body parts of the current user and generate a model based on these measure. The measurement process can be done either

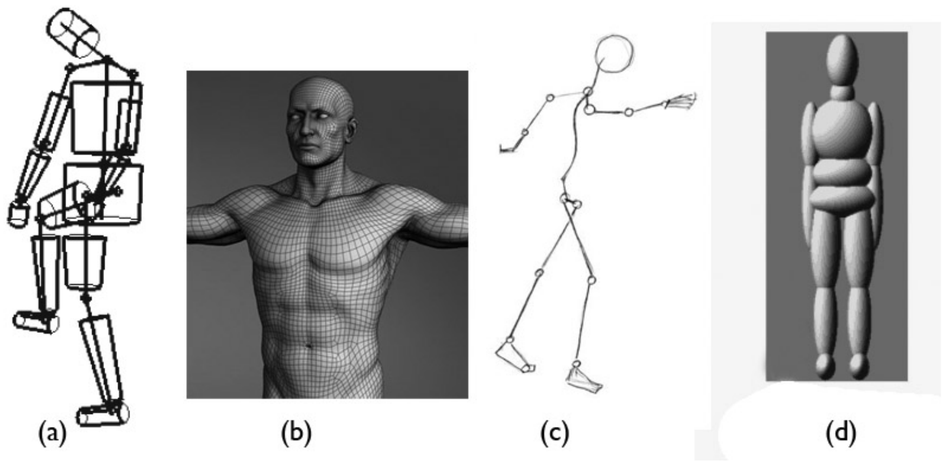


Fig. 2.1. Some of the body models you can find in literature. (a) model composed by truncated cones and spheres connected to joints by links that can be rigid or loose. (b) mesh surface body model. (c) stick figure body model. (d) model composed by spheroid.

in a on-line or off-line computation. Gu et. al [37] use 13 cameras around the user to capture the user silhouettes. The system is essentially based on shape from contour and uses deformable superquadrics as modeling primitive. From the superquadrics model, a parametric surface model is derived which is used in mannequin modeling. Kakadiaris and Metaxas [41] gives a method for on-line model generation where the model is computed while the user performs a known sequence of movements. Also in the work of Wren et. al [88] they find in an on-line way the user initial pose by building and refining a model. The same strategy is used by Mikic [56] where the model is computed in the first few frames of a sequence and a Bayes net used to refine the limbs measurements.

Figure 2.4 shows some of the body models you can find in literature.

2.5 Conclusion

In our work we have taken in consideration a marker-less system based on a skeletal body model. The body model is a simple stick figure model. Using a simple model give us the possibility to be faster during the computation phase of the algorithm. The tracking strategy used is an extended Kalman filter applied after the model fitting phase.

3

Volume Reconstruction and Skeletonization

In this chapter we are going to introduce our system starting from the volume reconstruction and skeletonization phases.

3D voxel reconstruction is the problem of reconstructing a volume using multiple-view silhouettes.

3.1 Shape from Silhouette

Shape from silhouette consist in recovering a volumetric approximate description of the human body (the *visual hull* [48, 49]) from its silhouettes projected onto a number of cameras. Its main advantage over other reconstruction techniques is that it integrates the information from multiple cameras instead of using only one camera. Moreover, implementations have been demonstrated that achieves real-time performances [51] by exploiting the graphical hardware.

Silhouettes are obtained by background subtraction with the software distributed with the HumanEva dataset [80]. The reconstruction is accomplished using the technique described in [51,55], with a plane parallel to the floor sweeping the working volume. At each step the silhouettes are projected onto the current plane, using the projective texture mapping feature of OpenGL and the GPU acceleration, as described in [29]. The slice of the volume corresponding to the plane is reconstructed by doing the intersection of the projected silhouettes.

3.1.1 Introducing the reconstruction algorithm

The algorithm to compute the 3D shape is composed by several phases. First of all the 3D space is subdivided in voxel (an abbreviation for volume element) that will form the base unit for the process. Voxel can be considered as little cubes that fill a 3D space. Subsequently each pixel in the image space of the acquiring cameras is re-projected in the 3D space. At the end of the process a three dimensional buffer is obtained containing boolean values. Each value is true if every camera re-project on it. This buffer can be used for example to compute a three-dimensional mesh (e.g. using marching cubes algorithm) and used in any CAD package.

There are several implementation in literature for this kind of reconstruction but most of them are too complex and time consuming. To obtain good results in performance and quality of the volume it is necessary to use high number of voxel. To reconstruct the geometry of a volume in a cubic space using 512 voxel as side of the cube, the number of bytes needed are $512 \times 512 \times 512 \times 4$, that is equal to 512 Megabyte.

3.1.2 Implementation

To implement the reconstruction algorithm the OpenGL library has been used. The time consuming steps of a 3D volume reconstruction algorithm can be transferred, using the library, from the main processor to the 3D graphic processor on the graphic card. This processor it is in fact better than a classic CPU for some operations: the random access memory, the computation of vectors and matrix multiplication, the image compression. Specifically, in our case of 3D volume reconstruction, the benefits using the graphic processor are several. For the volumetric reconstruction using silhouettes, it is possible to perform the entire computation in real time and with less memory consumption. Using the same cube of the example above, only a buffer of 512×512 bit can be used.

The use of the Graphic Processor (GPU) gives us the possibility to obtain better performance in the reconstruction phase but will bring some problems to access its commands. It has a set of instructions and specific data structures. To access all the functionalities of the GPUs an interface is needed. Usually this interface is given by languages as DirectX or OpenGl. These languages take care of managing all the GPU functionalities in a transparent manner for the software engineers. Thus it is essential to adapt the algorithms and to map them on the interface given by the graphic languages.

One of the most important feature used in our algorithm is Multi Texturing, i.e., the possibility to access more texture at the same time for each pixel in the viewport. This feature is now present in all the graphic cards, and was introduced in the standard set of functions of OpenGl in the 1998 (version 1.3). The use of different images at the same time is accomplished with the texture units. They are hardware implementation of texturing functions, and they work independently from the GPU. At the end of texturing phase, the colors obtained from the different images are combined together to obtain another color (in the specific case of the OpenGl the command used is the glTexEnv). One of the limitation founded using the graphic languages is that the results of the operation are saved in a video buffer, that is restrain in the dimension of the viewport. At the end of the GPU computation it is necessary to transfer the data using the OpenGl function glReadPixels. Using this function reduces the complexity of the code implementation but also the global algorithm performance. Future version of the code will implements the same algorithm using extensions of the OpenGl language (ARB vertex buffer object, ARB pixel buffer object e EXT framebuffer object) taking advantage of the asynchronous data transfer.

3.1.3 The algorithm

The algorithm works now in this way: the images captured from each cameras are transferred in the video memory using the texture units. Each texture units will be bind to a specific texture and a specific camera projection matrix (we supposed to have already calibrated the cameras). Each voxel will be represented for the entire reconstruction process as a data structure composed by only two vectors: one bi-dimensional vector that will determine the position of the pixel containing the result of the output buffer and a three-dimensional vector containing the position of the centroid of each voxel. Then the algorithm use two phases to obtain the final result.

A *vertex level* elaboration phase where the only target is to map the texture in the specific pixels of the output buffer. In this phase it is necessary to set the ModelView matrix equal to an identity matrix. A *pixel level* elaboration phase that is the core of the reconstruction algorithm. Using the data from the previous vertex level, the GPU processor will apply some math operation and texture access, to obtain the result in a specific pixel. Giving as result a pixel value that is black or white it is possible to set the boolean value in the 3D output buffer (0 if the voxel is assigned to empty space and 1 if it is assigned to the object to be reconstructed).

It is easy to see that running the algorithm will result in an high memory usage problem. If the space to be reconstructed is subdivided in $l \times l \times l$ voxel, the algorithm should use l^3 vertex, and visualized in OpenGl using the POINT primitive. This is inefficient in term of memory usage and system calls. We can move around the problem in a easy way. Using a bi-dimensional buffer we can divide the rendering in more phases based on horizontal slices for example.

The algorithm can compute the rendering slice by slice using a number of voxel equal to l^2 and reducing the memory space used. We still have the problem of the number of vertex used in the computation. But to solve this we only have to use the linear interpolation on the data from the vertex level and the pixel level computation. It is possible to suppose that all the slices are taken uniformly along the Z axis and that they are mapped on the output buffer directly (i.e. the voxels of the K-th horizontal slice with centroid coordinates (x, K, z) are mapped to the pixel (x, z)). Thus rendering a QUAD where the corners are the corners of the texture to be mapped on a slice will produce the same result as rendering l^2 vertex, using the POINT primitive, on a specific horizontal slice. Our solution make use of only l QUAD primitives.

3.1.4 Reconstruction of the Volume

The method of reconstructing a volume using the silhouettes does not make use of any information about the color or the texture on the surface of the body. The algorithm needs all the camera to be calibrated and also for each camera a binary image where each pixel is equal to 0 if it belongs to the background and 1 if it belongs to the shape to be reconstructed. The result of the the algorithm is a volume corresponding to the intersection of all the cones generated by the foreground silhouettes.

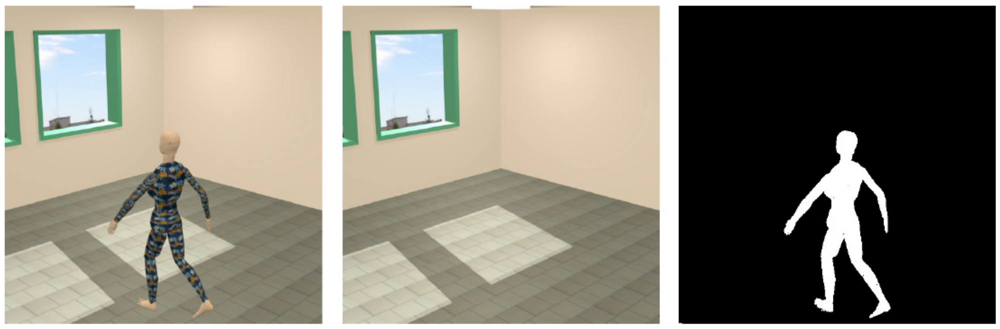


Fig. 3.1. The foreground/background subtraction phase using a virtual room created using Blender.

The reconstruction technique is composed by the next phases:

- Background/Foreground acquisition: in this phase the silhouettes need to be segmented out from the background. Standard algorithm can be used in this phase (in our case we used the segmentation algorithm given in the HumanEva dataset). The result from this phase is a set of binary images corresponding to the object silhouettes (see figure 3.1.4 for an example taken from a virtual room);
- Voxel Labeling: each voxel is ideally re-projected into the images. It will be assigned to the 3D object to be reconstructed only if the corresponding value of mapped image pixels is equal to 1 for all the images.

The main phase of the algorithm is thus the one that label the voxel to be part of the object or to be part of the environment. The entire process is implemented using only operation of the graphic card processor, letting free the main processor for other computations. The algorithm is inspired by the method explained in [28, 29, 51, 55, 85].

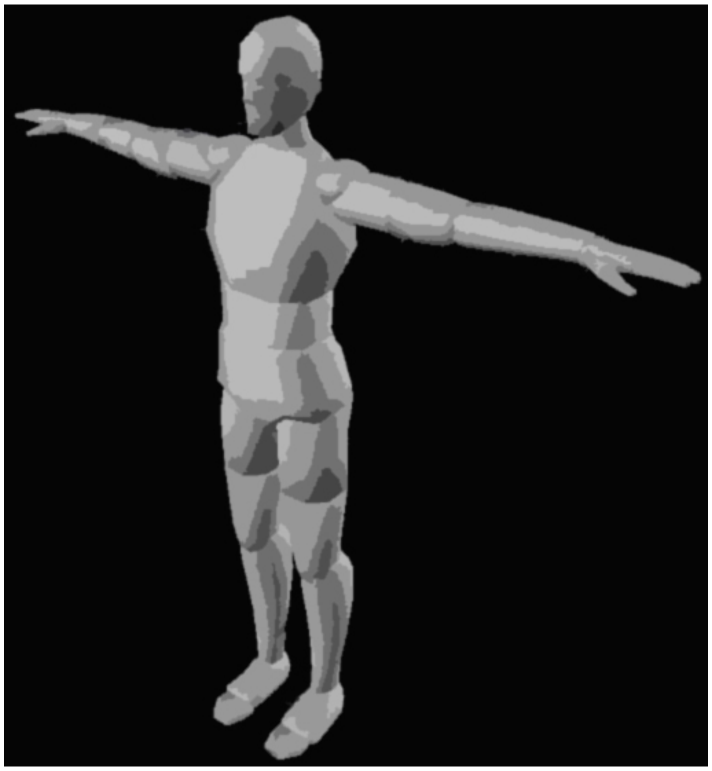


Fig. 3.2. The Vitruvian or T-Pose of a body.

These are the main steps used by the reconstruction OpenGl algorithm:

Texture Setting: this step requires to know the number of texture units to be used, the mapping of each of the silhouettes image and the projection matrices of the cameras. When the texture unit is enable the right silhouette is assigned to it and a projection matrix is created corresponding to a reprojection to the 3D space. In this phase each texture unit is initialized with a blend mode set to MODULATE thus the final color resulting from the intersection of all the silhouettes will be the product of all the colors (in our case the product will be equal to 1 if and only if all the images intersect in that point with color equal to 1)

Slice Rendering: slice in our algorithm is a simple OpenGL QUAD parallel to one of the world reference frame axis. This step use a QUAD OpenGl primitive to render each slice using as reference only the Z coordinate of the slice. The process goes on as described in the OpenGl standard. The graphic processor multiplies each QUAD with the ModelView matrix and the Projection matrix, and then it interpolates al the textures. The final result of the texture units modulation is a binary AND. In figure 3.1.4 an example of some slices taken in this phases is shown.

Reconstruction: this phase only loops on the slice to reconstruct the entire volume. In our algorithm this phase do also a blob segmentation to find the skeletal point. This improvment of the algorithm is explained in section 3.2.

Figure 3.2 shows the steps to get the final volume reconstruction.

3.2 Skeletonization

In the previous section we showed the procedure to reconstruct a 3D volume starting from the silhouettes. In this section we introduce an improvement on the algorithm suitable to find the medial skeletal axis points.

The medial axis (or skeleton) of a 3D object is the locus of the centers of maximal spheres contained in the object. In principle it is a surface, even if it can degenerate to a curve or a point. A close relative is the *centerline* (or *curve-skeleton*) that is a curve in 3D space that captures the main object's symmetry axes and roughly runs along the middle of an object. This definition matches with the stick-figures model, hence the data onto which the model is to be registered will be points on the body's centerline.

There are many techniques in literature to find skeletons or centerlines of a 3D object, for example [33,55]. However, they are too computationally demand-

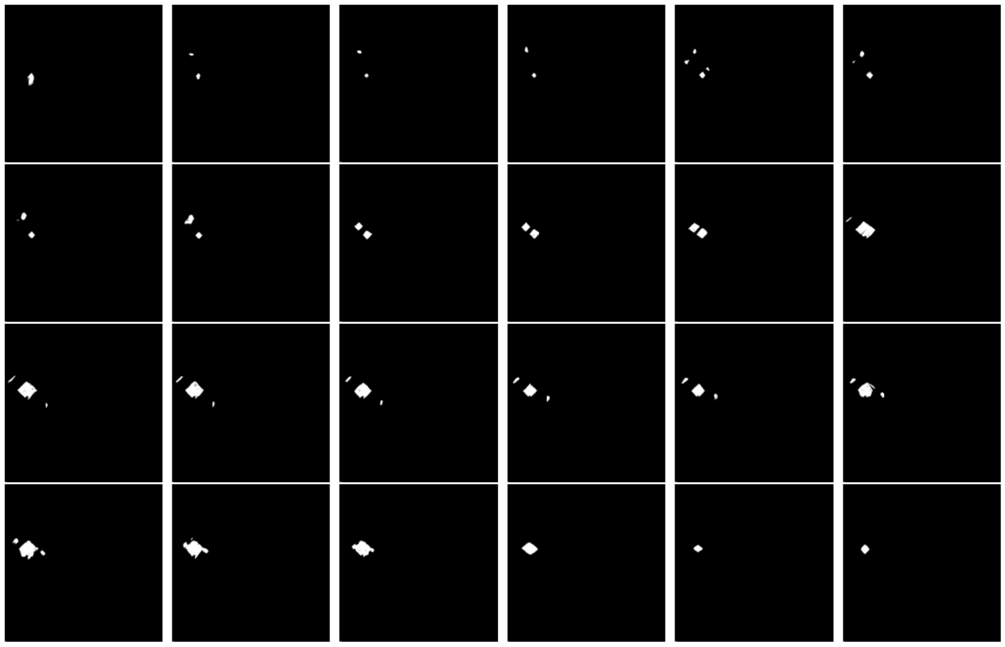


Fig. 3.3. Rendering of 30 horizontal slices.

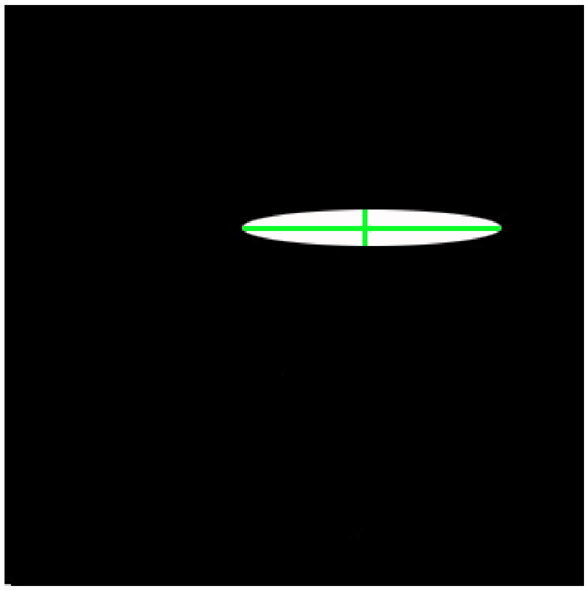


Fig. 3.4. Principal axis of a blob.

ing to fit our design, as our target speed is 100 frames per second. With this requirement in mind, we introduce a new method based on slicing the volume along three axis-parallel directions. For our purpose we can thus simplify the step of saving all the 3D voxel. In fact we don't need to save all the voxels inside the volume but only the voxels representing the centroid of the blobs. In term of memory space we have a big improvement because of the fact that for each slice the number of blob is usually not more than three and thus for each slice we have at most 3 points.

In each slice – which is a binary image – we compute the centroid of every connected component (called also blobs) and add it to the set of centerline points.

Several refinement to this procedure can be apply: blobs with an area considered too small can be filtered. Another refinement is to filter the points of blobs with a shape that is stretched in one direction. Think about a person standing in the T pose where the arms are parallel to the floor. In this position the horizontal slices passing through the arms will have blobs with shape showed in figure 3.2. As it is possible to see this kind of blob has two principle axis that are quite different in their lengths. If we find a centroid point in this type of blob it is possible to filter it out because of the fact we are interested in only medial axis points. The slicing along the Z-axis comes for free from the previous volumetric reconstruction stage, whereas slicing along X and Y must be done purposively, but uses the same procedure with GPU acceleration. We therefore add two passage more of the slicing algorithm adding computation time to the procedure but obtaining better result in the skeletal points reconstruction.

In OpenGl this part of the algorithm is easy achieved. When we have blended all the textures on a slice we have as results a binary image with some blobs on it. We can then find all the blobs on it and after that find the centroids of the blobs. Using a simple data structure we can save all the 3D points and use them in the phase of body model fitting (see Chapter 4).

Our method is similar to [55] which computes the centerline of a body by finding the centroids of the blobs produced by intersecting the body with planes orthogonal to Z-axes. Using a single sweep direction has some problems with some configurations of the body. Consider for example the so called "Vitruvian" or "T-pose": using only the scan along the Z-axes we completely loose the arms because by cutting the body at the arms height produces one single elongated blob containing a slice of the torso and the two arms, whose centroid is located on the vertebral column.

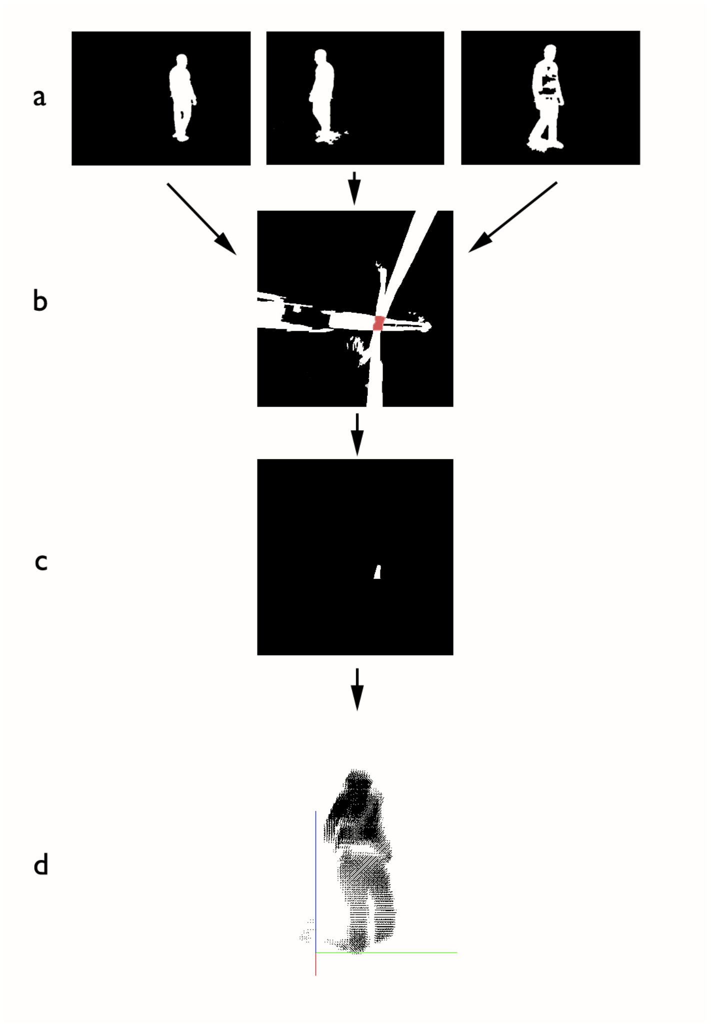


Fig. 3.5. a) Silhouettes; b) projection onto the sweeping plane; c) intersection (slice) d) final volumetric reconstruction

Our method, introduced in [61], solves this problem using three sweeps, thus it can be considered as a refinement of [55]. On the other hand, it can also be regarded as a coarse approximation of [85], where first 2D skeletons are extracted for each axis-parallel 2D slice of the 3D volume and then they are intersected to obtain the 3D centerline of the object. When the centroid belongs to the centerline the method is correct, even if returns only a subsampling of the centerline. We argue that this is approximately the case (i.e., the centroid lies close to the centerline) for most configurations of the human body. Yet, when the 2D shape is strongly non convex and the centroid falls outside the shape itself the method yields spurious points. However, the subsequent fitting procedure explained in Chapter 4, has been designed to be robust with respect to outliers, hence, for the sake of speed, we are prepared to tolerate this effect.

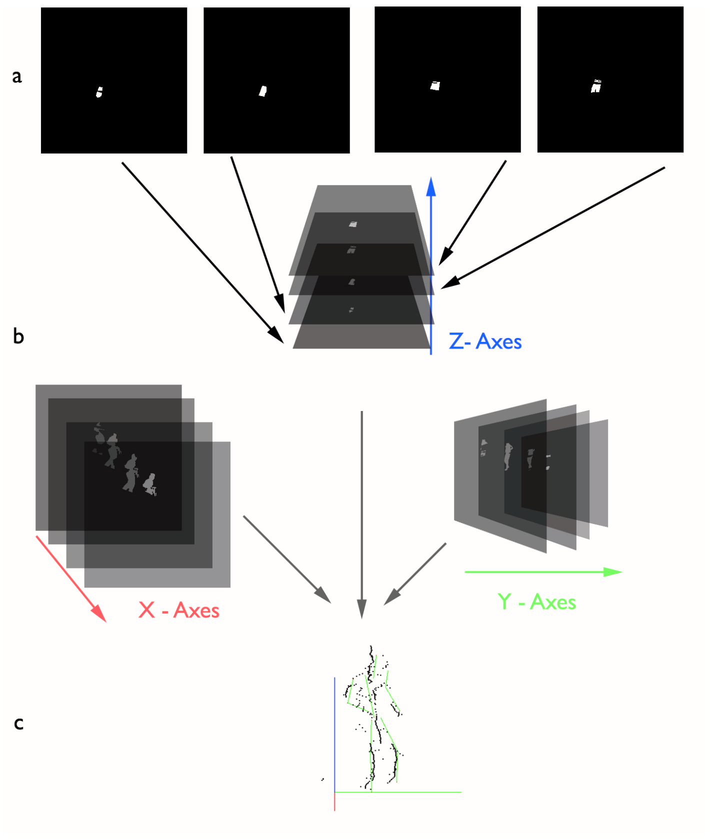


Fig. 3.6. a) slices along Z; b) slices along X and Y; c) centerline points with the stick figure overlaied.

In this chapter we introduced an algorithm to reconstruct a 3D volume based on silhouettes. A new method for finding medial skeletal points has been introduced. In the next chapter it will be explained how a stick body model can be fitted on the 3D skeletal points.

4

Hierarchical Articulated ICP and Human Body Model

4.1 Human Body Model

In this section we describe the articulated model representing the human body pose we used in the paper.

4.1.1 H-Anim Body Model

We based the structure of the body model used in the experiments on the H-Anim standard [36]. Humanoid Animation (H-Anim) is a standard for the definition and animation of virtual humanoids. It was developed by the Humanoi Animation Working Group and now it is a ISO standard. The scope of this standard is to give a framework on which define the humanoids to be used in virtual worlds. An humanoid defined using this standard can be seen as a set of bones and joints covered with an external surface representing the skin or the clothes. The standard is based on the VRML and X3D languages but no limitations are given about the use of other languages.

The standard defines four levels of articulation, i.e. how many joints we would like to use in the animation. Building to a particular level of articulation will ensure that the humanoid will be compatible with animations and other humanoids built to the same or higher level of articulation. The standard also specifies that the four level of articulation suggested are provided for information only and are not required by the H-Anim standard. This means that we can define our level of articulation. You can see the complete joints and segments definition of the standard in Figure 4.1.1.

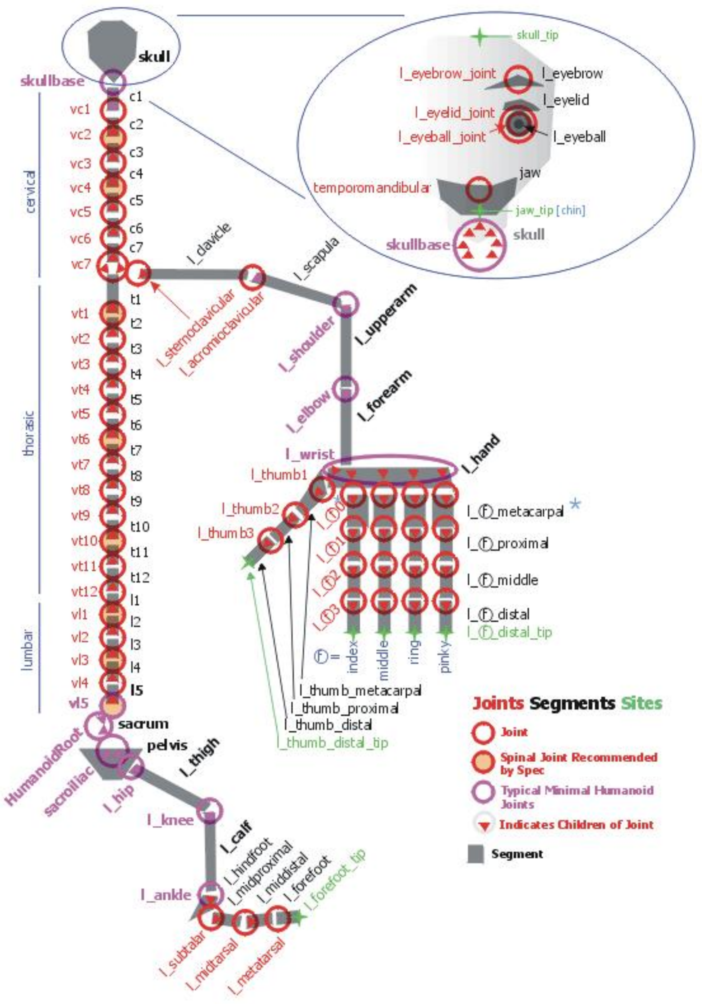


Fig. 4.1. The H-Anim body model. This is the complete model with all the joints and segments/limbs. The complete articulated body it is also called by the standard as Level Four Articulated Body. The picture is taken from the H-Anim web site [36].

4.1.2 Our Body Model

The body model we use in our experiments consists of a kinematic chain of ten sticks and nine joints, as depicted in Figure 4.2. Taking in consideration the H-Anim standard our Level of Articulation is close to the so called Level One of articulation. The torso is at the root of tree, children represents limbs, each limb being described by a fixed-length stick and the corresponding rotation from its parent. Hence, the motion of one body segment can be described as the motion of the previous segment in the kinematic chain and an angular motion around a body joint. Only the torso contains a translation that accounts for the translation of the whole body. Rotations are represented with 3×3 matrices. For the sake of simplicity, all the joints are spherical (three d.o.f.).

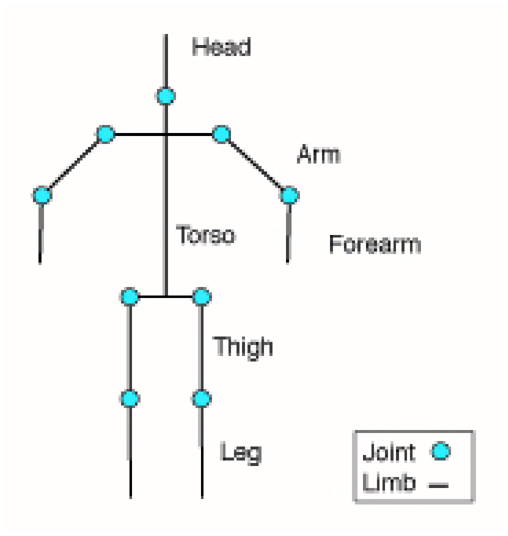


Fig. 4.2. The stick figure body model.

It is known in literature that a loose-limbed body model works better in a tracking algorithm [81]. In fact using this kind of models when we lose track of the body it is easier to recover it. Because our algorithm is intended to be used in a framework that already takes in consideration a body pose estimation in each frame, we use a rigid-limbed body model instead of loose-limbed. This choice reduce the number of computation and, in our experiments, we found no alignment problems.

In the experiments we used the classical matrix representation for the rotations of the joints. Note that only for the torso line segments we need to use a 4×4 rototranslation matrix. For all the other joints it is possible to use a simple 3×3 rotation matrix because we know exactly where all the joints are positioned (the model is rigid).

4.2 Weighted Extended Orthogonal Procrustes (WEOP)

In this section we describe the solution adopted to solve the estimation of the transformation parameters between two sets of points.

In literature there are several methods to solve the registration problem. We focused on a method called *Orthogonal Procrustes* problem. The method was explained by Schoenemann [74]. In his work he gave the direct least squares solution to find the transformation $\mathbf{T}$ such that a given matrix $\mathbf{A}$ is transformed to a given matrix $\mathbf{B}$ in such a way that the sum of the residual matrix $\mathbf{E} = \mathbf{AT} - \mathbf{B}$ is minimized. The first generalization of the method was proposed by Schoenemann and Caroll [75] where an unknown scale factor was introduced in the formulation. This second problem was also called *Extended Orthogonal Procrustes.*

For our purpose we need a further generalization of the problem. Lissitz et al. [52] and Koschat and Swayne [47] introduced two different methods to give a weight to the columns or rows of the matrices and Goodall [34] introduced another approach that can differently weight the homologous points. We based our algorithm on this last extension and we give the formulation using the notation in Akca [5].

Given the matrices $\mathbf{A}_{(p\times k)}, \mathbf{B}_{(p\times k)}$ and the weights matrices $\mathbf{W}_{\mathbf{p}(p\times p)}$ and $\mathbf{W}_{\mathbf{k}(k\times k)}$ the Extended Procrustres Problem is to find the matrix $\mathbf{T}$ such that

$$E = sAT + jt^\top - B \tag{4.1}$$

is minimized. The aim is achieved if we satisfy the next two conditions.

$$tr\{(sAT + jt^\top - B)^\top W_p(sAT + jt^\top - B)W_k\} = min \tag{4.2}$$

$$T^\top T = TT^\top = I \tag{4.3}$$

In our case it is simple to see that $k = 3$ because we work with 3D points and p represents the number of 3D points used in the algorithm.

We assume that $W_k = I$ because we don't need to weight in different ways the three orthogonal axis, and $s = 1$ because we assume we don't need a scale factor.

It is possible to reformulate equation 4.2 as:

$$tr\{(AT + jt^\top - B)^\top W_p(AT + jt^\top - B)\} = min \tag{4.4}$$

Now we decompose W_p using the Cholesky decomposition:

$$W_p = Q^\top Q \tag{4.5}$$

and thus equation 4.4 becomes:

$$\begin{aligned}
&tr\{(AT + jt^\top - B)^\top Q^\top Q(AT + jt^\top - B)\} = \\
&= tr\{(T^\top A^\top + tj^\top - B^\top)Q^\top Q(AT + jt^\top - B)\} = \\
&= tr\{(T^\top A^\top Q^\top + tj^\top Q^\top - B^\top Q^\top)(QAT + Qjt^\top - QB)\} = \\
&= tr\{(QAT + Qjt^\top - QB)^\top (QAT + Qjt^\top - QB)\} = min \qquad (4.6)
\end{aligned}$$

To simplify the notation let $A_w = QA$, $B_w = QB$ and $j_w = Qj$. Substituting in equation 4.6

$$tr\{(A_w T + j_w t^\top - B_w)^\top (A_w T + j_w t^\top - B_w)\} = min. \qquad (4.7)$$

This last equation is in the form of the *Extended Procrustes Problem*. The solution to the problem (derived in [5]) is based on the Singular Value Decomposition (SVD). Let

$$UDV^\top = A_w \left(I_p - \frac{\mathbf{j}_w \mathbf{j}_w^\top}{\mathbf{j}_w^\top \mathbf{j}_w} \right) B_w^\top \qquad (4.8)$$

be the SVD decomposition of the matrix on the right-hand side[1], where $\mathbf{j}_w = W\mathbf{j}$. The sought transformation is given by (we omit the scale c that is not needed in our case):

$$R = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^\top \qquad (4.9)$$

$$\mathbf{t} = (B_w - RA_w) \frac{\mathbf{j}_w}{\mathbf{j}_w^\top \mathbf{j}_w} \qquad (4.10)$$

The diagonal matrix in (4.9) is needed to ensure that the resulting matrix is a rotation [44].

4.3 Hierarchical Articulated ICP

This section describes the Hierarchical Articulated ICP algorithm for registering an articulate stick model to a cloud of points (these points come from the skeletonization phase). It is based on the well-known Iterative Closest Point (ICP) [14, 22] that estimates the rigid motion between a given set of $3D$ data points and a set of $3D$ model points.

We assume that the data are 3D points distributed roughly around the centerline of the body's segments. They come from the skeletonisation of a volumetric reconstruction of the body, as in [19, 54, 55]

[1] Please note that $A\frac{\mathbf{j}_w \mathbf{j}_w^\top}{\mathbf{j}_w^\top \mathbf{j}_w}$ is a matrix of the same size as A with identical columns, each of them equal to the centroid of the points contained in A.

The data are registered to the model using a hierarchical approach that starts from the torso and traverses the kinematic chain down to the extremities. At each step ICP computes the best rigid transformation of the current limb that fits the data while preserving the articulated structure.

For each node of the tree we find – with ICP – a transformation matrix T_i corresponding to the rotation of the limb with respect to its parent.

The closest point search works from the data to the model, by computing for each data point its closest point on the body segments. Only the matches with the current segment are considered, all the other should be – in principle – discarded. To compute the closest point of a data point we project the data point on the lines passing trough each of the body segments. In Figure 4.3 you can see an example of the process.

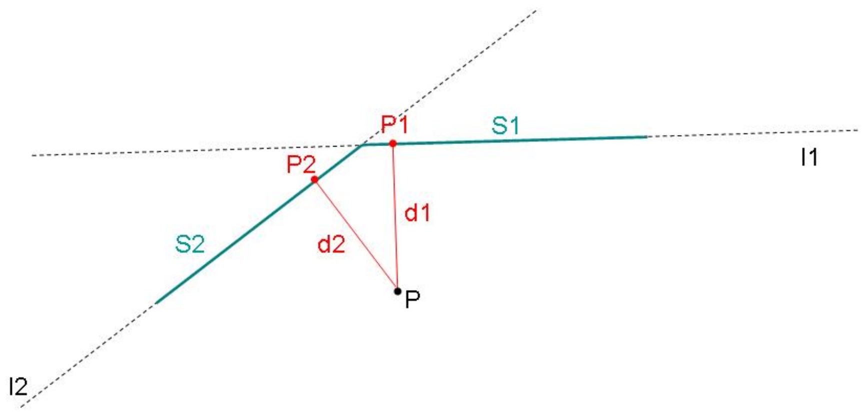


Fig. 4.3. A 2-D example of the computation of the closest point of P. Starting from point P we compute the projections on the lines $l1$ and $l2$ ($S1$ and $S2$). The projected points $P1$ and $P2$ are used to find the closest points.

However, the rotation in 3D space of a line segment cannot be computed unambiguously, for the rotation around the axis is undetermined. In order to cope with this problem we formulate a *Weighted* Extended Orthogonal Procrustes Problem and give a small but non-zero weight also to points that match the descendants of the current segment in the kinematic chain. In this way they contribute to constrain the rotation around the segment axis. Think, for example, of the torso: by weighting the points that match the limbs as well, even if they cannot be aligned with single rigid transformation, the coronal (aka frontal) plane can be correctly recovered.

In order to improve ICP robustness against false matches and spurious points, following [31], we discard closest pairs whose distance is larger than a threshold computed using the X84 rejection rule. Let e_i be the closest-point distances, a robust scale estimator is the Median Absolute Deviation (MAD):

$$\sigma^* = 1.4826 \,\mathrm{med}_i \, |e_i - \mathrm{med}_j \, e_j|. \tag{4.11}$$

The X84 rejection rule prescribes to discard those pairs such that $|e_i - \mathrm{med}_j \, e_j| > 3.5\sigma^*$.

The Hierarchical Articulate ICP is is described step by step in Algorithm 1.

The Iterative Closest Point (ICP) [14, 22] is customary used for registration of 3D sets of points. The standard algorithm estimates the rigid motion between a given set of $3D$ data points and a set of $3D$ model points. In summary:

Algorithm 1 ICP

```
Input: two sets of $p$ 3D points, the data $\{\mathbf{a}_i\}_{i=1...p}$ and the model $\{\mathbf{b}_i\}_{i=1...p}$
Output: the rigid motion $T$ that brings the data onto the model
1. For each point $\mathbf{a}_i$ of the data set find the closest point $\mathbf{b}_i$ in the model set.
2. Given these putative correspondences, estimate the global motion transformation $T$ be-
   tween all the points by solving an Extended Orthogonal Procrustes Problem (see below).
3. Apply the transformation $T$ to the data points.
4. Repeat from step 1 until the average distance between closest points is less than a given
   threshold.
```

Obviously using line segments as model we have ambiguities on the rotation. Consider in fact to align a line segment to a set of $3D$ points. Suppose for example that this segment is the torso of our human body model. After we have align it with the set of corresponding closest points the standard ICP algorithm stops and gives a transformation matrix. This transformation matrix doesn't take into account the rest of body limbs. In fact any rotation about the line segment axis does not affect the alignment of the line segment itself. But for our tracking algorithm it is important to know also this rotation because it gives us the right position of all the other joints and limbs such that shoulders and hips. To overcome this problem we used the WEOP method. WEOP allows to weights each point in the set. We can give an high weight to the points that correspond to the torso line segment and a low weight to all the other points. In this way we can drive the rototranslation matrix such to satisfy our task.

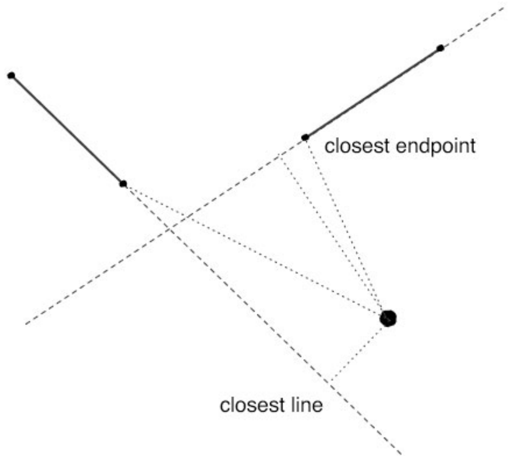


Fig. 4.4. An example showing that the closet segment is not necessarily the one belonging to the closet line.

There are two sets: the data made of $3D$ points (call it D) and the model made of line segments (call it S). We also need a set of lines (call it L) corresponding to the line segments, i.e., for each line segment $s_i \in S$ there exist one line $l_i \in L$ such that s_i lie on the line l_i. Let $\|D\| = n$ the number of points and let $\|S\| = \|L\| = m$ the number of segments and lines. In our case $m = 10$ (see section 4.1.2) and n is not defined a priory. Now for each point $d_j \in D$ we find the closest line l_j to the point. Let d_{lj} be the projection of d_j onto l_j. If the point d_{lj} falls inside the correspondent segment s_j than it is the closest point of d_j, otherwise search for the closest point of d_j among the segments endpoints.

Point 6, where a transformation is computed given some putative correspondences, deserves to be expanded, in order to make the paper self-contained. The problem to be solved is an instance of the *Extended Orthogonal Procrustes Problem* (EOPP) [75], which can be stated as follows: transform a given matrix A into a given matrix B by a similarity transformation (rotation, translation and scale) in such a way to minimize the sum of squares of the residual matrix. More precisely, since we introduced weights on the points, we shall consider instead the *Weighted Extended Orthogonal Procrustes Problem* (WEOPP) problem.

Algorithm 2 Hierarchical Articulate ICP

```
Input: The model $\mathcal{S}$ composed by segments and the data set $\mathcal{A}$ of 3D points
Output: a set of rigid motions (referred to the kinematic chain) that brings the model onto the data

1. Traverse the body model tree structure using a level-order or a preorder traversal method.
2. Let $s_j \in \mathcal{S}$ be the current body segment.
3. Compute the closest points:
    a) For each data point $\mathbf{a}_i \in \mathcal{A}$ and for each segment $s_\ell \in \mathcal{S}$ compute its projection $\mathbf{p}_{i\ell}$ onto the line containing $s_\ell$ ;
    b) if $\mathbf{p}_{i\ell} \in s_\ell$ then add $\mathbf{p}_{i\ell}$ to $\mathcal{M}$ (the set of the closest-point candidates), otherwise add the endpoint of $s_\ell$ to $\mathcal{M}$.
    c) Find $\mathbf{b}_i$, the closet point to $\mathbf{a}_i$ in $\mathcal{M}$.
4. Weight the points: If $\mathbf{b}_i$ belongs to $s_j$ than its weight is 1, otherwise it is $\varepsilon$ (chosen heuristically) for all the descendant and 0 for all the others.
5. If the distance of $\mathbf{b}_i$ to $\mathbf{a}_i$ is above the X84 threshold then the weight is set to 0.
6. Solve for the transformation of $s_j$.
7. Apply the transformation to $s_j$ and its the descendants.
8. Repeat from step 3 until the weighted average distance between closest points points is less than a given threshold.
```

The *Weighted Orthogonal Procrustes Problem* (WOPP) problem is a special case of WEOPP and the solution can be derived straightforwardly by setting $\mathbf{u} = \mathbf{0}$. In our case we use WEOPP for the torso and WOPP (only rotation) for the limbs.

The hierarchical articulate ICP is deterministic, every limb is considered only once and brought into alignment with ICP. The transformation that aligns a limb s_j is determined mostly by the points the matches s_j and secondarily by the points that matches its descendants. The transformation is applied to s_j and its descendants, considered as a rigid structure. The output of the algorithm represents the pose of the body. In a tracking framework, the pose obtained at the previous time-step is used as the initial pose for the current frame.

A similar algorithm has been independently proposed in [73]. The main difference is in the way the basic ICP is applied to the articulated structure, which leads to different schema. In [73] at each step of the algorithm the subtree of the selected joint is rigidly aligned using ICP with no weights, i.e., all the descendants of the joint plays the same role in the minimization. As a result, the same joint needs to be considered more than once to converge to the final solution. In this regard our approach is less computationally demanding. On the other hand one error in the alignment of a limb propagates downward without recovery, whereas in [73] a subsequent sweep may be able to correct the error, hence it is probably more tolerant to a looser initialization.

5 Tracking

In this chapter it is described the kinematic of the body model and the tracking algorithm. The notation used is similar to Mikic [56]. We have taken as base work for the tracking strategy this paper.

5.1 Direct and Inverse Body Kinematic

As introduced in section 4.1.2 our body model consists of ten limbs and nine joints. There are several possibilities to represents the body model configuration. We are searching of a model with fewer parameters because of the fact that we need fewer parameters to track and thus fewer constraints to impose during the tracking algorithm. In our process we have to deal with measurements that are 3D points related to the body. The way to reduce all these data is to use a body configuration based on the joints angles. In literature there exist already a framework that can be used in our problem. The solution is to use a kinematic chain based on angles and twists as described in many books like [30, 65, 76].

5.1.1 Twist rotation representation

The twist representation of rotation is an elegant solution that can be used for our purpose. This representation it is based on the observation that every rigid motion transformation can be represented as a rotation around an axis of a determined angle and a translation along this axis. We can represent the twist as a vector or as a matrix [18, 65]:

$$\xi = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \text{ or } \hat{\xi} = 0 \begin{pmatrix} 0 & -\omega_z & \omega_y & v_1 \\ \omega_z & 0 & -\omega_x & v_2 \\ -\omega_y & \omega_x & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.1}$$

where ω is a 3D unit vector pointing in the axis direction. It is possible to specify the amount of rotation using a scala angle θ that is multiplied by the twist. A twist can always be converted into a classical 4×4 roto-translation matrix using the formula:

$$\mathbf{T} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = e^{\hat{\xi}} = \mathbf{I} + \hat{\xi} + \frac{(\hat{\xi})^2}{2!} + \frac{(\hat{\xi})^3}{3!} + \dots \tag{5.2}$$

It also possible to verify that the velocity of a point $\mathbf{p}(t)$ is:

$$\dot{\mathbf{p}}(t) = \omega \times (\mathbf{p}(t) - \mathbf{q}) \tag{5.3}$$

where $-\omega \times \mathbf{q} = \mathbf{v}$ and $\mathbf{q} \in \mathbb{R}^3$ is a point on the axis. It is possible to see [56] that the solution to the equation 5.3 is:

$$\bar{\mathbf{p}}(t) = e^{\hat{\xi}_t} \bar{\mathbf{p}}(0) \tag{5.4}$$

where $e^{\hat{\xi}_t}$ is the function that maps the initial location of a point $\mathbf{p}$ to a new location of the point after the rotation of t radians around the axis ω and $\bar{\mathbf{p}} = [\mathbf{p}1]^\top$ is the homogeneous point representation. Thus we can represent a rotation of θ radians around an axis as

$$e^{\hat{\xi} \cdot \theta} \tag{5.5}$$

If, as in our case, we have an open kinematic chain with $K + 1$ segments linked together with K joints, it can be shown that:

$$g_p(\Theta) = e^{\hat{\xi}_1 \cdot \theta_1} \cdot e^{\hat{\xi}_2 \cdot \theta_2} \cdot \ldots \cdot e^{\hat{\xi}_k \cdot \theta_k} \cdot g_p(0) \tag{5.6}$$

where $g_p(\Theta)$ represent the rigid body transformation between the base reference frame and a point on the last segment of the chain. The configuration of the model can thus be described by the angles $\Theta = [\theta_1 \quad \theta_2 \ldots \theta_k]^\top$.

5.1.2 Kinematic Chains

The body model we are using in our experiments is made of five kinematics chains:

- Torso, Head
- Torso, Left Upper Arm, Left Lower Arm
- Torso, Right Upper Arm, Right Lower Arm
- Torso, Left Thigh, Left Calf
- Torso, Right Thigh, Right Calf

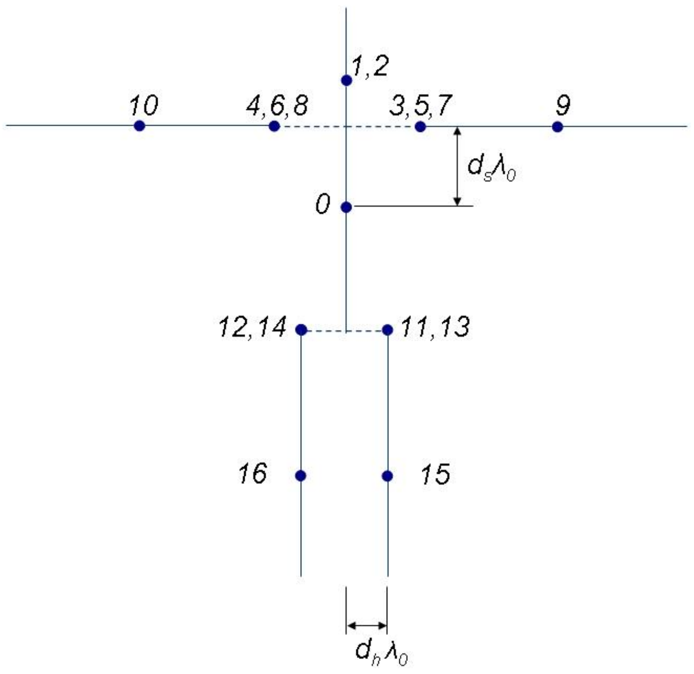


Fig. 5.1. The stick figure body model. In this picture it is possible to see where are located every joints. The number on the joints correspond to the number in Table 5.4. d_s is the distance between the torso reference frame and the shoulder reference frame. d_h is the distance between the torso reference frame and the hip reference frame. For the dimension of each limbs we use the notation of Mikic [56] where $\lambda_i^{(j)}$ means the length of limb number i along the j-th direction. In our work only the torso, called limb 0, has a dimension $\lambda_0^{(1)}$ that is equal to the distance between shoulder. All the other limbs have only one dimension.

The position of each joint is fixed respect to the torso reference frame. The body part dimensions are initialized before the tracking algorithm. For the experiments done on the HumanEva data-set all the body part dimensions are evaluated as mean over all the frame of the dataset. After the first computation of the limbs dimensions they rest fixed along all the tracking.

We used sixteen axes of rotation (as in [56]) but we used for some of them a different formulation. We take as initial zero position the one where the body is in a standing position with arms pointing out the side of the body and legs not astride but down along the hip width (as shown in picture 5.1 and 5.2). All the angles are restricted to be in a determined range of value. In this way we can represent the body physical constraint of movements (for example it is not possible to bend backward the elbow or bend forward the knee).

The rotation about the axes are modeled using the twist and the exponential formula introduced in section 5.1.1. All the angles, except the torso, are relative to the torso. In Table 5.1 all the axis of rotation of each angles are listed together with their work range.

For the orientation and position of the torso an axis-angle representation is used. The torso is completely identified by two vectors and a scalar: $\mathbf{t}_0 = [t_{0x} \quad t_{0y} t_{0z}]$, $\omega_0 = [\omega_{0x} \quad \omega_{0y} \quad \omega_{0z}]$ and θ_0. The first is the translation vector of the torso reference frame respect to the world reference frame. The second vector and the scalar are respectively the axis of rotation and the angle around this axis. Equation 5.16 gives the relation between these parameters and the rotation matrix $\mathbf{R}_0$ where the elements of the matrix are listed in the equations from 5.7 to 5.15.

$$r0_{11} = 1 - \frac{1 - \cos(|\omega_0|\theta_0)(\omega_{0y}^2 + \omega_{0z}^2)}{|\omega_0|} \tag{5.7}$$

$$r0_{12} = \frac{\omega_{0z}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0x}\omega_{0y}}{|\omega_0|^2}(1 - \cos(|\omega_0|\theta_0)) \tag{5.8}$$

$$r0_{13} = \frac{-\omega_{0y}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0x}\omega_{0z}}{|\omega_0|^2}(1 - \cos(|\omega_0|\theta_0)) \tag{5.9}$$

$$r0_{21} = -\frac{\omega_{0z}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0x}\omega_{0y}}{|\omega_0|^2}(1 - \cos(|\omega_0|\theta_0)) \tag{5.10}$$

$$r0_{22} = 1 - \frac{1 - \cos(|\omega_0|\theta_0)(\omega_{0x}^2 + \omega_{0z}^2)}{|\omega_0|} \tag{5.11}$$

Joint	Angle	Rotation Axis	Range
Neck (1)	θ_1	X Torso	$\left[-\frac{\pi}{2} \quad \frac{\pi}{2}\right]$
Neck (2)	θ_2	Y Torso	$\left[-\frac{\pi}{2} \quad \frac{\pi}{2}\right]$
Left Shoulder (1)	θ_3	X Torso	$[-\pi \quad \pi]$
Right Shoulder (1)	θ_4	X Torso	$[0 \quad 2\pi]$
Left Shoulder (2)	θ_5	Z Torso	$\left[-\pi \quad \frac{\pi}{2}\right]$
Right Shoulder (2)	θ_6	Z Torso	$\left[-\pi \quad \frac{\pi}{2}\right]$
Left Shoulder (3)	θ_7	Y Shoulder	$\left[-\frac{\pi}{2} \quad \frac{3}{4}\pi\right]$
Right Shoulder (3)	θ_8	Y Shoulder	$\left[-\frac{3}{4}\pi \quad \frac{\pi}{2}\right]$
Left Elbow	θ_9	Z Shoulder	$[-\pi \quad 0]$
Right Elbow	θ_{10}	Z Shoulder	$[-\pi \quad 0]$
Left Hip (1)	θ_{11}	X Torso	$\left[-\frac{\pi}{3} \quad \frac{\pi}{2}\right]$
Right Hip (1)	θ_{12}	X Torso	$\left[-\frac{\pi}{2} \quad \frac{\pi}{3}\right]$
Left Hip (2)	θ_{13}	Y Torso	$\left[-\pi \quad \frac{\pi}{2}\right]$
Right Hip (2)	θ_{14}	Y Torso	$\left[-\pi \quad \frac{\pi}{2}\right]$
Left Knee	θ_{15}	Y Hip	$[0 \quad \pi]$
Right Knee	θ_{16}	Y Hip	$[0 \quad \pi]$

Table 5.1. Range of the valid values for each body model joint angles. The third column of the table specify the axis of rotation of the angles.

$$r0_{23} = \frac{\omega_{0x}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0y}\omega_{0z}}{|\omega_0|^2}(1-\cos(|\omega_0|\theta_0)) \tag{5.12}$$

$$r0_{31} = \frac{\omega_{0y}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0x}\omega_{0z}}{|\omega_0|^2}(1-\cos(|\omega_0|\theta_0)) \tag{5.13}$$

$$r0_{32} = -\frac{\omega_{0x}}{|\omega_0|}\sin(|\omega_0|\theta_0) + \frac{\omega_{0y}\omega_{0z}}{|\omega_0|^2}(1-\cos(|\omega_0|\theta_0)) \tag{5.14}$$

$$r0_{33} = 1 - \frac{1-\cos(|\omega_0|\theta_0)(\omega_{0x}^2+\omega_{0y}^2)}{|\omega_0|} \tag{5.15}$$

$$\mathbf{R}_0 = \begin{pmatrix} r0_{11} & r0_{12} & r0_{13} \\ r0_{21} & r0_{22} & r0_{23} \\ r0_{31} & r0_{32} & r0_{33} \end{pmatrix} \tag{5.16}$$

It is also possible to compute the $\mathbf{R}_0$ matrix, given the axis of rotation ω_0 and the angle θ_0, using the next passages:

$$c = \cos(\theta_0) \tag{5.17}$$
$$s = \sin(\theta_0) \tag{5.18}$$
$$t = 1 - c \tag{5.19}$$

$$x = \omega_{0x} \tag{5.20}$$
$$y = \omega_{0y} \tag{5.21}$$
$$z = \omega_{0z} \tag{5.22}$$

$$R = \begin{pmatrix} xx & xy & xz \\ xy & yy & yz \\ xz & yz & zz \end{pmatrix} \tag{5.23}$$

$$W = \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix} \tag{5.24}$$

the result is the matrix

$$R_0 = c\mathbf{I}_3 + tR + sW \tag{5.25}$$

The complete configuration of the body can thus be parameterized by two vectors and seventeen angles: $\mathbf{t}_0$, ω_0, $\theta_0 \ldots \theta_{16}$. For all the angles from θ_1 to θ_{16} the rotation matrix are:

$$\mathbf{R}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_1 & -s_1 \\ 0 & s_1 & c_1 \end{pmatrix} \tag{5.26}$$

$$\mathbf{R}_2 = \begin{pmatrix} c_2 & 0 & s_2 \\ 0 & 1 & 0 \\ -s_2 & 0 & c_2 \end{pmatrix} \tag{5.27}$$

$$\mathbf{R}_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_3 & -s_3 \\ 0 & s_3 & c_3 \end{pmatrix} \tag{5.28}$$

$$\mathbf{R}_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_4 & -s_4 \\ 0 & s_4 & c_4 \end{pmatrix} \tag{5.29}$$

$$\mathbf{R}_5 = \begin{pmatrix} c_5 & -s_5 & 0 \\ s_5 & c_5 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.30}$$

$$\mathbf{R}_6 = \begin{pmatrix} c_6 & -s_6 & 0 \\ s_6 & c_6 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.31}$$

$$\mathbf{R}_7 = \begin{pmatrix} c_7 & 0 & s_7 \\ 0 & 1 & 0 \\ -s_7 & 0 & c_7 \end{pmatrix} \tag{5.32}$$

$$\mathbf{R}_8 = \begin{pmatrix} c_8 & 0 & s_8 \\ 0 & 1 & 0 \\ -s_8 & 0 & c_8 \end{pmatrix} \tag{5.33}$$

$$\mathbf{R}_9 = \begin{pmatrix} c_9 & -s_9 & 0 \\ s_9 & c_9 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.34}$$

$$\mathbf{R}_{10} = \begin{pmatrix} c_{10} & -s_{10} & 0 \\ s_{10} & c_{10} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.35}$$

$$\mathbf{R}_{11} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{11} & -s_{11} \\ 0 & s_{11} & c_{11} \end{pmatrix} \tag{5.36}$$

$$\mathbf{R}_{12} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{12} & -s_{12} \\ 0 & s_{12} & c_{12} \end{pmatrix} \tag{5.37}$$

$$\mathbf{R}_{13} = \begin{pmatrix} c_{13} & 0 & s_{13} \\ 0 & 1 & 0 \\ -s_{13} & 0 & c_{13} \end{pmatrix} \tag{5.38}$$

$$\mathbf{R}_{14} = \begin{pmatrix} c_{14} & 0 & s_{14} \\ 0 & 1 & 0 \\ -s_{14} & 0 & c_{14} \end{pmatrix} \tag{5.39}$$

$$\mathbf{R}_{15} = \begin{pmatrix} c_{15} & 0 & s_{15} \\ 0 & 1 & 0 \\ -s_{15} & 0 & c_{15} \end{pmatrix} \tag{5.40}$$

$$\mathbf{R}_{16} = \begin{pmatrix} c_{16} & 0 & s_{16} \\ 0 & 1 & 0 \\ -s_{16} & 0 & c_{16} \end{pmatrix} \tag{5.41}$$

We can express the body model configuration as a vector of 23 parameters:

$$\text{Body Configuration } = [\mathbf{t}_0 \quad \omega_0 \quad \theta_0 \ldots \theta_{16}] \tag{5.42}$$

During the experiments we considered as measurements 15 points located on the body model. They are shown in picture 5.2. At each frame, using the technique in Chapter 4, we move the model on a different pose. The model itself gives us the fifteen reference points.

It is easy to see that for each point the rotations that affect its position are only those on the relative kinematic chain. For example the left foot orientation is affected by rotation on the torso, left hip, and left knee:

$$\text{Left Foot Orientation } = \mathbf{R}_0 \cdot \mathbf{R}_{11} \cdot \mathbf{R}_{13} \cdot \mathbf{R}_{15}. \tag{5.43}$$

In general the position of a point $\bar{\mathbf{p}}(\Theta)$ on the model respect to the torso reference frame can be expressed as:

$$\bar{\mathbf{p}}(\Theta) = \mathbf{M}_1 \cdot \mathbf{M}_2 \cdot \ldots \cdot \mathbf{M}_k \cdot \bar{\mathbf{p}}(0) \tag{5.44}$$

where the matrices $\mathbf{M}_i$ are:

$$\mathbf{M}_i = \begin{pmatrix} \mathbf{R}_i & \mathbf{t}_i \\ 0 & 1 \end{pmatrix} \tag{5.45}$$

If we need to represent the point in the world reference frame it is easy to see that we need only to do the transformation:

$$\bar{\mathbf{p}}_0 = \mathbf{M}_0 \cdot \bar{\mathbf{p}}(\Theta) \tag{5.46}$$

This give us the direct kinematic for each points, that is, the function that transform from the angles space to the points space. In Table 5.2 there are all the formulas to compute the 15 reference points given the configuration of the model.

In our experiments we need also the inverse kinematic of the model, that is, the function that transform from the reference points to the angles. Table 5.4 gives all the formula needed to compute the angles taken in consideration in our model.

i	Point $\mathbf{p}_i$	Equation to compute $\mathbf{p}_i$
1	Torso centroid	t_0
2	Head Tip	$R_0(R_1(R_2 \cdot t_{2\_3}) + t_{3\_1}) + t_0$
3	Neck	$R_0 \cdot t_{10\_0} + t_0$
4	Left Shoulder	$R_0 \cdot t_{4\_1} + t_0$
5	Right Shoulder	$R_0 \cdot t_{5\_1} + t_0$
6	Left Hip	$R_0 \cdot t_{6\_1} + t_0$
7	Right Hip	$R_0 \cdot t_{7\_1} + t_0$
8	Left Elbow	$R_0(R_3(R_5 \cdot t_{8\_4}) + t_{4\_1}) + t_0$
9	Right Elbow	$R_0(R_4(R_6 \cdot t_{9\_5}) + t_{5\_1}) + t_0$
10	Left Knee	$R_0(R_{11}(R_{13} \cdot t_{10\_6}) + t_{6\_1}) + t_0$
11	Right Knee	$R_0(R_{12}(R_{14} \cdot t_{11\_7}) + t_{7\_1}) + t_0$
12	Left Hand Tip	$R_0(R_3(R_5(R_7(R_9 \cdot t_{12\_8}) + t_{8\_4})) + t_{4\_1}) + t_0$
13	Right Hand Tip	$R_0(R_4(R_6(R_8(R_{10} \cdot t_{13\_9}) + t_{9\_5})) + t_{5\_1}) + t_0$
14	Left Foot	$R_0(R_{11}(R_{13}(R_{15} \cdot t_{14\_10} + t_{10\_6})) + t_{6\_1}) + t_0$
15	Right Foot	$R_0(R_{12}(R_{14}(R_{16} \cdot t_{15\_11} + t_{11\_7})) + t_{7\_1}) + t_0$

Table 5.2. Direct Kinematic.

5.1.3 Kalman Filter

Our tracking strategy is based on the use of a Kalman filter. In practice for each frame we take from the ICP algorithm the 15 reference points, and than using the state at the previous frame we pass these data to the Kalman filter that gives us the current state.

The Kalman filter [12, 42, 43, 87] estimates the state of a dynamic system. It is a recursive predictive filter that is based on the use of state space techniques and recursive algorithm. It is an estimator of the state of a dynamic system and it takes into consideration that the system can be disturbed by noises. Moreover the Kalman filter, to give better estimated state, uses measurements that are related to the state.

Vector $\mathbf{t}$	Values
t_0	p_0
t_{2_3}	$[0 \quad 0 \quad \lambda_1^{(2)}]$
t_{3_1}	$[0 \quad 0 \quad \lambda_0^{(2)}]$
t_{4_1}	$[0 \quad \lambda_0^{(1)} \quad ds\lambda_0^{(2)}]$
t_{5_1}	$[0 \quad -\lambda_0^{(1)} \quad ds\lambda_0^{(2)}]$
t_{6_1}	$[0 \quad d_H\lambda_0^{(1)} \quad -\lambda_0^{(2)}]$
t_{7_1}	$[0 \quad -d_H\lambda_0^{(1)} \quad -\lambda_0^{(2)}]$
t_{8_4}	$[0 \quad 2\lambda_2^{(2)} \quad 0]$
t_{9_5}	$[0 \quad 2\lambda_3^{(2)} \quad 0]$
t_{10_6}	$[0 \quad 0 \quad -2\lambda_4^{(2)}]$
t_{11_7}	$[0 \quad 0 \quad -2\lambda_5^{(2)}]$
t_{12_8}	$[0 \quad 2\lambda_6^2 \quad 0]$
t_{13_9}	$[0 \quad 2\lambda_7^2 \quad 0]$
t_{14_10}	$[0 \quad 0 \quad -2\lambda_8^{(2)}]$
t_{15_11}	$[0 \quad 0 \quad -2\lambda_9^{(2)}]$

Table 5.3. $\mathbf{t}$ vector values. The values of the λ_i^2 correspond to the length of the i-th segment. Note that the values of $d_s = 0.8$ and $d_H = 0.7$ are taken from [56]. They can also be estimated, as in the case of our experiments on the HumanEva dataset, measuring the ground-truth data.

The Kalman filter consists of two steps:

- prediction step
- correction step

The first step basically predict a new state based on the physical model of the system. The second step corrects the predicted model using the observed data that are converted in an observed model. The error covariance in this way is minimized.

The two steps of the filter are than repeated along all the time steps as depicted in Figure 5.3.

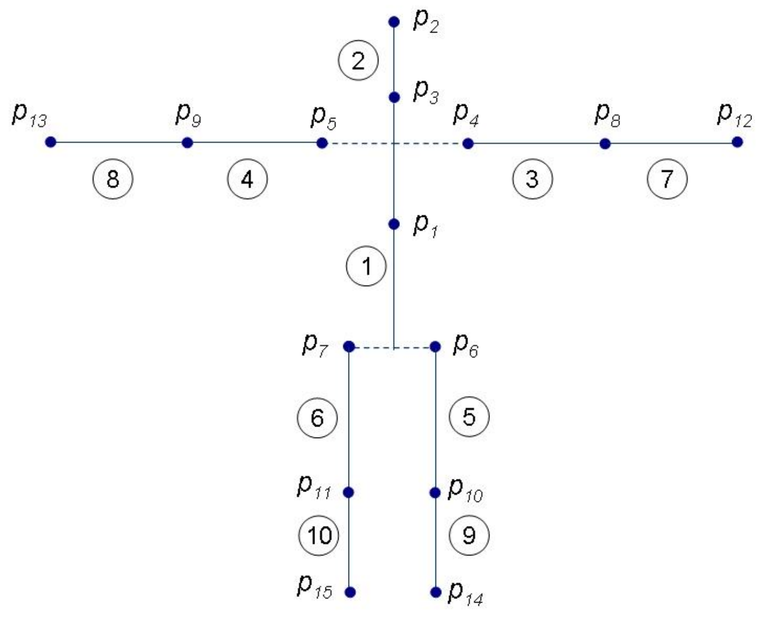


Fig. 5.2. The fifteen reference points. The numbers inside the circle specify the limbs number.

The Kalman filter algorithm can be summarized with the equations in Tables 5.5 and 5.6 (for a complete description see [42, 87]).

As we can see in Tables 5.5 and 5.6 the first step of the filter projects one step forward in time the state of the system $\hat{x}_{k-1}$ and the covariance matrix P_{k-1}. The second step of the filter first of all compute the Kalman gain K_k. With the gain and using the measure of the process z_k the filter generate a posteriori state estimate $\hat{x}_k$ and a posteriori covariance matrix P_k used in the next iteration.

This formulation of the Kalman filter takes into consideration only linear systems. But in our case we have a measurement equation (direct kinematic) that is nonlinear. When the system or the measurement equations are nonlinear we can use a generalization of the Kalman filter called Extended Kalman Filter (EKF). The EKF linearizes about the current estimated state and the system have to be represented by continuously differentiable functions [12, 87]. In Table 5.7 the new formulation for the extended version of the filter is shown.

Angle	Formula	Vector $\mathbf{v}$	Vector $\mathbf{p} = [x \quad y \quad z]^\top$
θ_1: Neck	$\theta_1 = -\arccos(\mathbf{e_3 v})sgn(y)$	$\mathbf{v} = \frac{1}{\sqrt{y^2+z^2}}[0 \quad y \quad z]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_2 - \mathbf{p}_3)$
θ_2: Neck	$\theta_2 = \arccos(\mathbf{e_3 v})sgn(x)$	$\mathbf{v} = \frac{1}{\sqrt{x^2+y^2+z^2}}[x \quad 0 \quad \sqrt{y^2+z^2}]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_2 - \mathbf{p}_3)$
θ_3: Left Shoulder (1)	$\theta_3 = \arccos(\mathbf{e}_2)sgn(z)$	$\mathbf{v} = \frac{1}{\sqrt{y^2+z^2}}[0 \quad y \quad z]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_8 - \mathbf{p}_4)$
θ_4: Right Shoulder (1)	$\theta_4 = \arccos(\mathbf{e}_2\mathbf{v})sgn(z)$	$\mathbf{v} = \frac{1}{\sqrt{y^2+z^2}}[0 \quad y \quad z]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_9 - \mathbf{p}_5)$
θ_5: Left Shoulder (2)	$\theta_5 = -\arccos(\mathbf{e}_2)sgn(x)$	$\mathbf{v} = \frac{1}{\sqrt{x^2+y^2+z^2}}[x \quad \sqrt{y^2+z^2} \quad 0]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_8 - \mathbf{p}_4)$
θ_6: Right Shoulder (2)	$\theta_6 = -\arccos(\mathbf{e}_2\mathbf{v})sgn(x)$	$\mathbf{v} = \frac{1}{\sqrt{x^2+y^2+z^2}}[x \quad \sqrt{y^2+z^2} \quad 0]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_9 - \mathbf{p}_5)$
θ_7: Left Shoulder (3)	$\theta_7 = -\arccos(\mathbf{R}_3\mathbf{R}_5\mathbf{e}_3(\mathbf{v}_1 \times \mathbf{v}_2))\cdot$ $(-sgn(\mathbf{e}_1(\mathbf{v}_1 \times \mathbf{v}_2)))$	$\mathbf{v}_1 = \mathbf{p}_4 - \mathbf{p}_8$ $\mathbf{v}_2 = \mathbf{p}_{12} - \mathbf{p}_8$	
θ_8: Right Shoulder (3)	$\theta_8 = \pi - \arccos(\mathbf{R}_4\mathbf{R}_6\mathbf{e}_3(\mathbf{v}_2 \times \mathbf{v}_1))\cdot$ $(-sgn(\mathbf{e}_1(\mathbf{v}_2 \times \mathbf{v}_1)))$	$\mathbf{v}_1 = \mathbf{p}_5 - \mathbf{p}_9$ $\mathbf{v}_2 = \mathbf{p}_{13} - \mathbf{p}_9$	
θ_9: Left Elbow	$\theta_9 = \arccos(\mathbf{v}_1\mathbf{v}_2) - \pi$	$\mathbf{v}_1 = \mathbf{p}_4 - \mathbf{p}_8$ $\mathbf{v}_2 = \mathbf{p}_{12} - \mathbf{p}_8$	
θ_{10}: Right Elbow	$\theta_{10} = \arccos(\mathbf{v}_1\mathbf{v}_2) - \pi$	$\mathbf{v}_1 = \mathbf{p}_5 - \mathbf{p}_9$ $\mathbf{v}_2 = \mathbf{p}_{13} - \mathbf{p}_9$	
θ_{11}: Left Hip (1)	$\theta_{11} = \arccos(-\mathbf{e}_3\mathbf{v})sgn(y)$	$\mathbf{v} = \frac{1}{\sqrt{y^2+z^2}}[0 \quad y \quad z]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_{10} - \mathbf{p}_6)$
θ_{12}: Right Hip (1)	$\theta_{12} = \arccos(-\mathbf{e}_3\mathbf{v})sgn(y)$	$\mathbf{v} = \frac{1}{\sqrt{y^2+z^2}}[0 \quad y \quad z]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_{11} - \mathbf{p}_7)$
θ_{13}: Left Hip (2)	$\theta_{13} = -\arccos(-\mathbf{e}_3\mathbf{v})sgn(x)$	$\mathbf{v} = \frac{1}{\sqrt{x^2+y^2+z^2}}[x \quad 0 \quad \sqrt{y^2+z^2}]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_{10} - \mathbf{p}_6)$
θ_{14}: Right Hip (2)	$\theta_{14} = -\arccos(-\mathbf{e}_3\mathbf{v})sgn(x)$	$\mathbf{v} = \frac{1}{\sqrt{x^2+y^2+z^2}}[x \quad 0 \quad \sqrt{y^2+z^2}]^\top$	$\mathbf{p} = \mathbf{R}_0^\top(\mathbf{p}_{11} - \mathbf{p}_7)$
θ_{15}: Left Knee	$\theta_{15} = \pi - \arccos(\mathbf{v}_1\mathbf{v}_2)$	$\mathbf{v}_1 = \mathbf{p}_{14} - \mathbf{p}_{10}$ $\mathbf{v}_2 = \mathbf{p}_6 - \mathbf{p}_{10}$	
θ_{16}: Right Knee	$\theta_{16} = \pi - \arccos(\mathbf{v}_1\mathbf{v}_2)$	$\mathbf{v}_1 = \mathbf{p}_{15} - \mathbf{p}_{11}$ $\mathbf{v}_2 = \mathbf{p}_7 - \mathbf{p}_{11}$	

Table 5.4. The Inverse Kinematics formulae used in the experiments. Note that all the points **p** are respect to the Torso Reference Frame. The vectors $\mathbf{e}_i$ are 3D unit vector in the direction of the i-th axis.

The Prediction Kalman filter step

$$\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_{k-1}$$

$$P_k^- = AP_{k-1}^- A^\top + Q$$

Table 5.5. The prediction formulae used in the kalman filter first step.

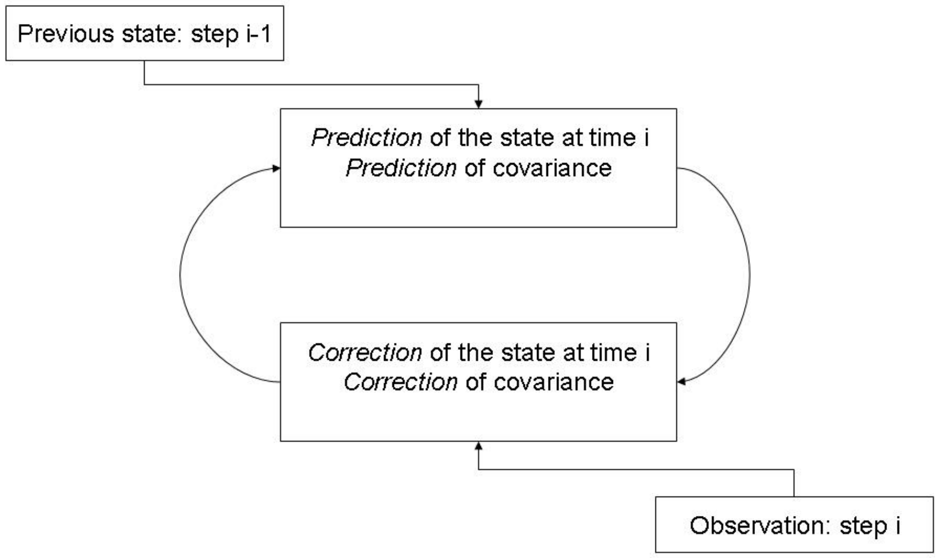

Previous state: step i-1
Prediction of the state at time i
Prediction of covariance
Correction of the state at time i
Correction of covariance
Observation: step i

Fig. 5.3. The working schema of a Kalman filter

The Correction Kalman filter step
$K_k = P_k^- H^\top (H P_k^- H^\top + R)^{-1}$
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
$P_k = (I - K_k H) P_k^-$

Table 5.6. The correction formulae used in the kalman filter second step.

5.2 The Tracker

In this section we will show how we implemented the tracker using the model and the EKF shown in previous sections of this chapter. The formulation and notation are similar to those used by Mikic [56]. The human body model we used has 23 parameters:

The Extended Kalman Filter correction step
$K_k = P_k^- H_k^\top (H_k P_k^- H_k^\top + R_k)^{-1}$
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0))$
$P_k = (I - K_k H_k) P_k^-$

Table 5.7. The correction formulae used in the Extended Kalman filter second step. Note that now H is the Jacobian matrix of partial derivatives of h respect to x.

$$\mathbf{x} = \begin{bmatrix} \mathbf{t}_0^\top & \omega_0^\top & \theta_0 & \theta_1 \dots \theta_{16} \end{bmatrix}^\top \tag{5.47}$$

Moreover our measurement vector is composed by 15 points:

$$\mathbf{z} = \begin{bmatrix} \mathbf{p}_1^\top & \mathbf{p}_2^\top & \dots \mathbf{p}_{15}^\top \end{bmatrix}^\top \tag{5.48}$$

We can assume that all the motion can be represented as noise and thus the state transition matrix A is equal to an identity matrix. The measurement equation (direct kinematic) is shown in Table 5.2. The only thing we need to complete the EKF formulation is the Jacobian.

To simplify the notation the structure of the Jacobian is shown in Table 5.8 and only an example is given for the complete formulation of one body point.

The computation of the Jacobian is easily obtained using tools like the *Symbolic Math Toolbox* of the Matlab environment [1]. We report here only an example of the computation of the Jacobian for the right hand finger tip point $\mathbf{p}_{13}$.

$$\frac{\partial \mathbf{p}_{13}}{\partial \omega_0} = \frac{\partial R_0}{\partial \omega_0}(R_4(R_6(R_8(R_{10} \cdot t_{13\_9}) + t_{9\_5})) + t_{5\_1}) \tag{5.49}$$

$$\frac{\partial \mathbf{p}_{13}}{\partial \theta_0} = \frac{\partial R_0}{\partial \theta_0}(R_4(R_6(R_8(R_{10} \cdot t_{13\_9}) + t_{9\_5})) + t_{5\_1}) \tag{5.50}$$

$$\frac{\partial \mathbf{p}_{13}}{\partial \theta_4} = \frac{\partial R_4}{\partial \theta_4}(R_6(R_8(R_{10} \cdot t_{13\_9}) + t_{9\_5})) \tag{5.51}$$

$$\frac{\partial \mathbf{p}_{13}}{\partial \theta_6} = \frac{\partial R_6}{\partial \theta_6}(R_8(R_{10} \cdot t_{13\_9}) + t_{9\_5}) \tag{5.52}$$

$$\frac{\partial \mathbf{p}_{13}}{\partial \theta_8} = \frac{\partial R_8}{\partial \theta_8}(R_{10} \cdot t_{13\_9}) \tag{5.53}$$

	$\mathbf{t}_0$	ω_0	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	θ_{10}	θ_{11}	θ_{12}	θ_{13}	θ_{14}	θ_{15}	θ_{16}
$\mathbf{p}_1$	I_3	χ	$\times$																
$\mathbf{p}_2$	I_3	χ	$\times$	$\times$	$\times$														
$\mathbf{p}_3$	I_3	χ	$\times$																
$\mathbf{p}_4$	I_3	χ	$\times$																
$\mathbf{p}_5$	I_3	χ	$\times$																
$\mathbf{p}_6$	I_3	χ	$\times$																
$\mathbf{p}_7$	I_3	χ	$\times$																
$\mathbf{p}_8$	I_3	χ	$\times$			$\times$		$\times$											
$\mathbf{p}_9$	I_3	χ	$\times$				$\times$		$\times$										
$\mathbf{p}_{10}$	I_3	χ	$\times$											$\times$		$\times$			
$\mathbf{p}_{11}$	I_3	χ	$\times$												$\times$		$\times$		
$\mathbf{p}_{12}$	I_3	χ	$\times$			$\times$		$\times$		$\times$		$\times$							
$\mathbf{p}_{13}$	I_3	χ	$\times$				$\times$		$\times$		$\times$		$\times$						
$\mathbf{p}_{14}$	I_3	χ	$\times$											$\times$		$\times$		$\times$	
$\mathbf{p}_{15}$	I_3	χ	$\times$												$\times$		$\times$		$\times$

Table 5.8. The Jacobian structure. χ represents a 3×3 matrix. $\times$ represents a 3×1 vector. The empty cells represent 3×1 zero vectors. Each non-empty cell represents a partial derivative $\frac{\mathbf{p}_i}{\partial \mathbf{x}_j}$ where $\mathbf{x}_j$ is a state variable specify by the Table column.

$$\frac{\partial \mathbf{p}_{13}}{\partial \theta_{10}} = \frac{\partial R_{10}}{\partial \theta_{10}} \tag{5.54}$$

The tracking procedure is explained by the algorithm 3

Algorithm 3 Tracking Algorithm

Input: the vector of the state at previous time step $\mathbf{x}_{t-1}$, the measurement vector at present time step $\mathbf{z}_t$, the covariance matrix of the previous time step $\mathbf{P}_{t-1}$.
Output: the vector of the state at present time step $\mathbf{x}_t$, the covariance matrix of the present time step $\mathbf{P}_t$

```
1. Using the ICP algorithm find the 15 reference points of the actual time step thus to obtain
   a measurement vector $z_t$.
2. Using the state vector $\mathbf{x}_{t-1}$ and the covariance matrix $\mathbf{P}_{t-1}$ compute the prediction of the
   Kalman filter.
3. Using the predicted state, the predicted covariance matrix and the measurement vector,
   compute the Jacobian and the corrected state vector $\mathbf{x}_t$ and covariance matrix $\mathbf{P}_t$
4. Repeat from step 1 passing $\mathbf{x}_t$ as $\mathbf{x}_{t-1}$ and $\mathbf{P}_t$ as $\mathbf{P}_{t-1}$
```

6

System Implementation

The system is now composed by different pieces of code implemented in Matlab or in C/C++. The system works off-line but the intention is to port all the code in C/C++ language and to use it as an online real-time system.

6.1 Camera Calibration

The first step to use the system is to calibrate the cameras in the room. For the calibration of the intrinsic parameters we used a standard check-board calibration grid (see Figure 6.1) and the Camera Calibration Toolbox implemented for the Matlab environment and also included in the Open Source Computer Vision library distributed by Intel [24].

For each camera we computed the focal length $F_c \in \mathbb{R}^2$, the principal point $C_c \in \mathbb{R}^2$ and the distortion coefficient $K_c \in \mathbb{R}^5$. All the camera used in the system are Basler f602c model with 640x480 resolution. We assumed that the pixel are square. For the calibration of the extrinsic camera parameters, corresponding to the rotation $R_c \in SO(3)$, and the translation $T_c \in \mathbb{R}^3$ we used the CVLab Matlab library written by Andrea Fusiello. The extrinsic parameters refer to a global references frame based on a corner of a check-board made by A4 sheets of paper laid on the floor (see Figure 6.3).

6.2 Frame Capture

The architecture of our hardware system is based on a client/server structure. For each camera we have a client pc that sends the frames to a server. The server collects all the frames from each camera.

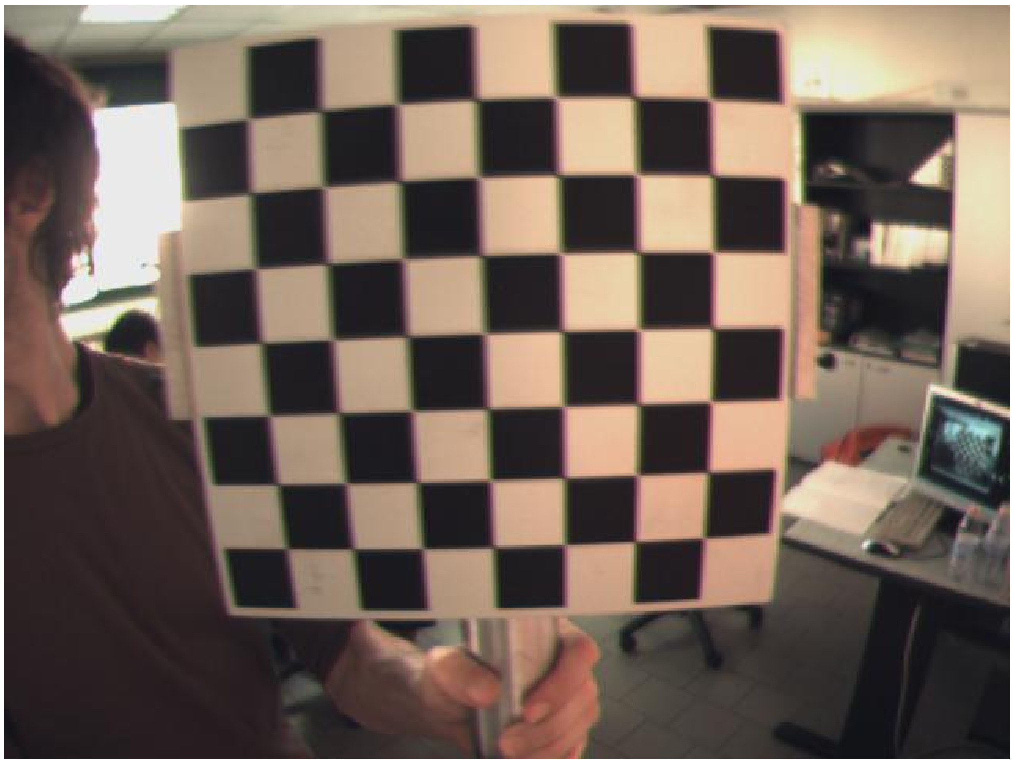


Fig. 6.1. The checkboard used for the intrinsic camera parameters.

Clients and server at the start up do not know each others. The first operation executed by the client is the camera initialization. Each clients check if the camera is connected and if it is, it enables the frames download. If all goes well the client starts to initialize the socket.

The server should be launched whenever all the clients are ready. The server checks the number of active cameras and checks the ip address of all the computers used in the system.

After the initialization procedure all the clients stop and wait a command from the server. The type of commands are: start, stop, and suspend.

When the system is running it acquires the frames from the client computers and then sends them to the server. Some image elaboration can be done directly by the clients, for example silhouettes extraction and computation of the undistorted frames.

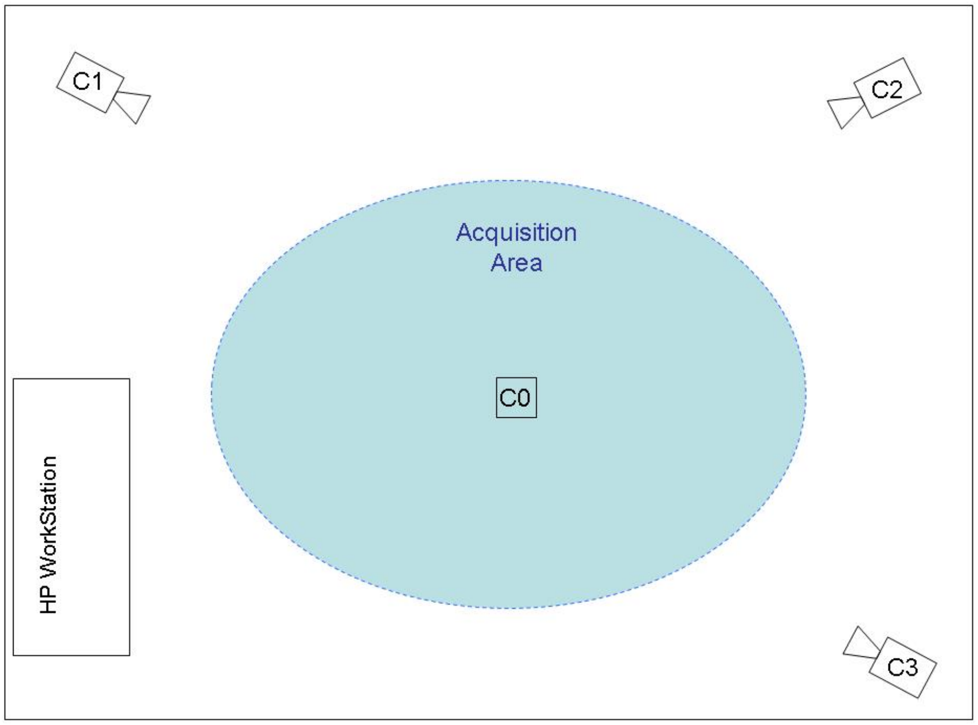


Fig. 6.2. Camera setup for the architecture. The camera number zero is pointing to the floor. All the camera are Basler model f602c

6.3 Background subtraction

The method used for silhouettes extraction is similar to the one used for the HumanEva dataset [80]. The method is based on two phases:

1. Background Statistic Computation
2. Background/Foreground Segmentation

The first part of the algorithm computes the needed statistic for the background. Frames from a background only movie are taken and the next equations are used to collect data on the background:

$$\mathbf{Mean} = \frac{\sum_{i=1}^{N} \mathbf{img}_i}{N} \tag{6.1}$$

$$\mathbf{Variance} = \frac{\sum_{i=1}^{N} \mathbf{img}_i^2}{N} - \left(\frac{\sum_{i=1}^{N} \mathbf{img}_i}{N} \right)^2 \tag{6.2}$$

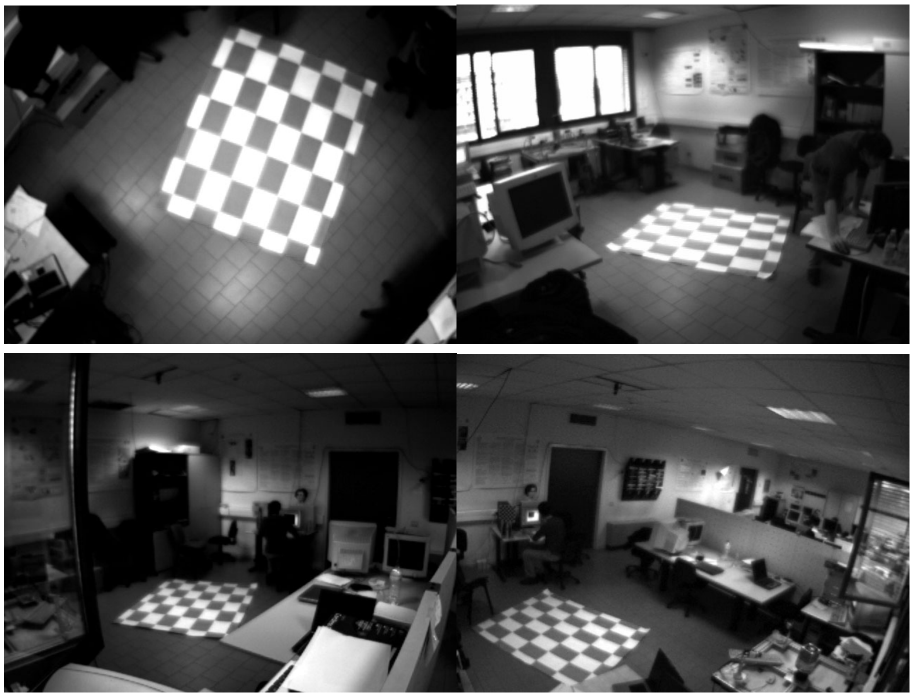


Fig. 6.3. The calibration checkboard used for the extrinsic camera parameters computation. Note that in these images there still be distortion due to the camera lens.

where N is the number of frames. The equations 6.1 and 6.2 are simply the mean and variance of the background taken on all the frames representing the background itself. These data are used in the second phase to extract the silhouettes. In this phase for each frame of the captured human being in the scene the algorithm compute the probability that a pixel is part of the background or foreground. The equation used is:

$$\mathbf{BGProbability} = \mathbf{pdf}(\mathbf{frame}, \mu, \sigma) \tag{6.3}$$

where $\mathbf{pdf}(\mathbf{frame}, \mu, \sigma)$ is the probability density function of the normal distribution with mean μ and standard deviation σ, evaluated at the values in **frame**. The algorithm than classify a pixel as background if its probability value is under a certain risk value. In picture 6.3 is shown the process for one frame.

6.4 3D Volume Reconstruction, ICP and Tracking

The technique of 3D reconstruction used in this work is fully explained in Chapter 3. This part of the architecture is implemented in C++ code. The code takes

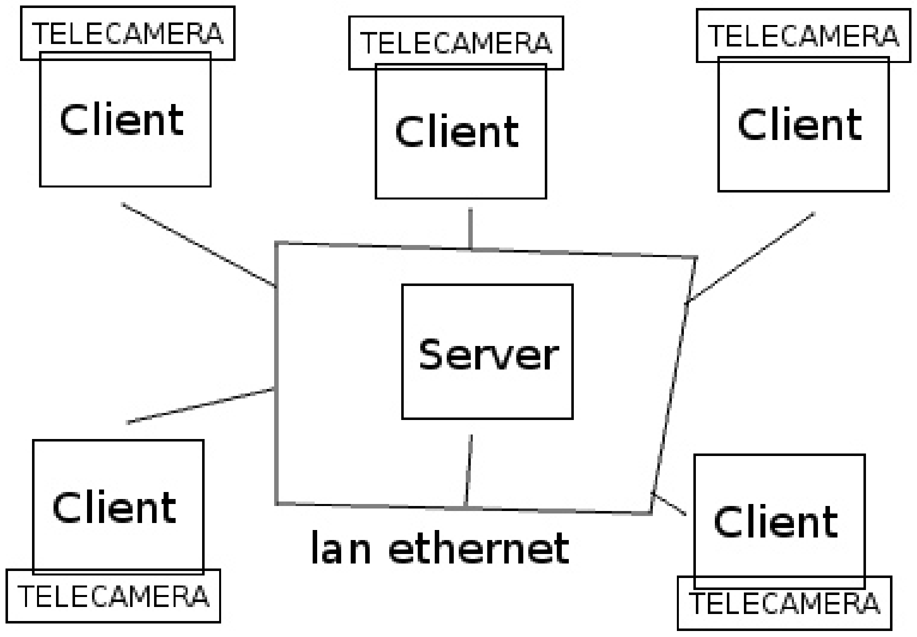


Fig. 6.4. The schema of the architecture.

as input the camera calibration parameters and the silhouettes. It produces as output a set of 3D points that are the centroid of each slices found. They represent the medial axis of the body to be reconstructed.

As explained in Chapter 4 using the set of 3D points we can fit on them an articulated skeletal model of a body using an ICP technique. This part of the code is written in Matlab with some procedure compiled as mex files.

The tracker algorithm (see Chapter 5) use the stick body model as measurement for the Extended Kalman Filter. Also this part of the code is implemented in Matlab environment.

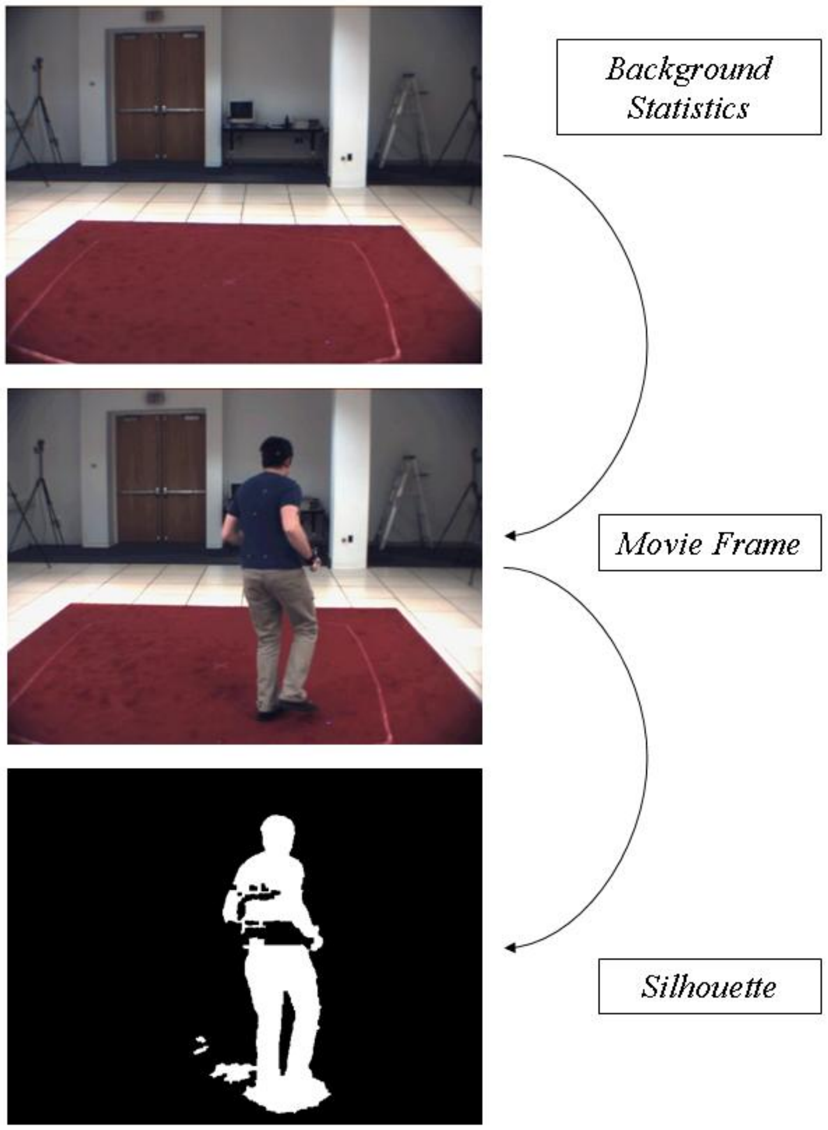


Fig. 6.5. The background subtraction process. Starting from the statistic of the background (the first image is the mean image taken over all the background frames), we use the actual frame to compute the **pdf** function and to obtain the silhouette.

7 Experimental Results

This chapter will show some experiments on real cases. All the methods described in this thesis are applied to achieve the results. As shown in Figure 7.1 starting from a sequence of image taken by several cameras we obtain a background segmentation. Using a standard technique we reconstruct a 3D volume of the body and than using one technique introduced in our thesis and already presented in the papers [61,62] we obtain a set of 3D points on a skeleton representation of the 3D volume. This set of points gives us the possibility to apply an ICP based algorithm to fit a 3D sticks model of a body. The model obtained in each frame is used inside a Extended Kalman Filter technique to track the poses of people.

7.1 Experiments on the HumanEva dataset

The body tracker has been tested on sequences taken from the HumanEva-I dataset [80]. All the sequences in HumanEva-I have been calibrated using the Vicon proprietary software and the motion data saved in the common `c3d` file format.

The dataset contains multiple subjects performing a variety of actions like walking, running, boxing, etc. Figures 7.2 and 7.3 show some sample frames from these sequences together with the output of the silhouette extraction.

The camera calibration parameters are already pre-computed in the dataset. After the extraction of the silhouettes the frame are undistorted and passed to the procedure that generates the slices and than the set of the 3D points on the centroid of the blobs.

As starting position of the model a ground-truth frame was used, i.e., we used a frame from the sequences and place the model using as reference points the points given by the dataset on that frame.

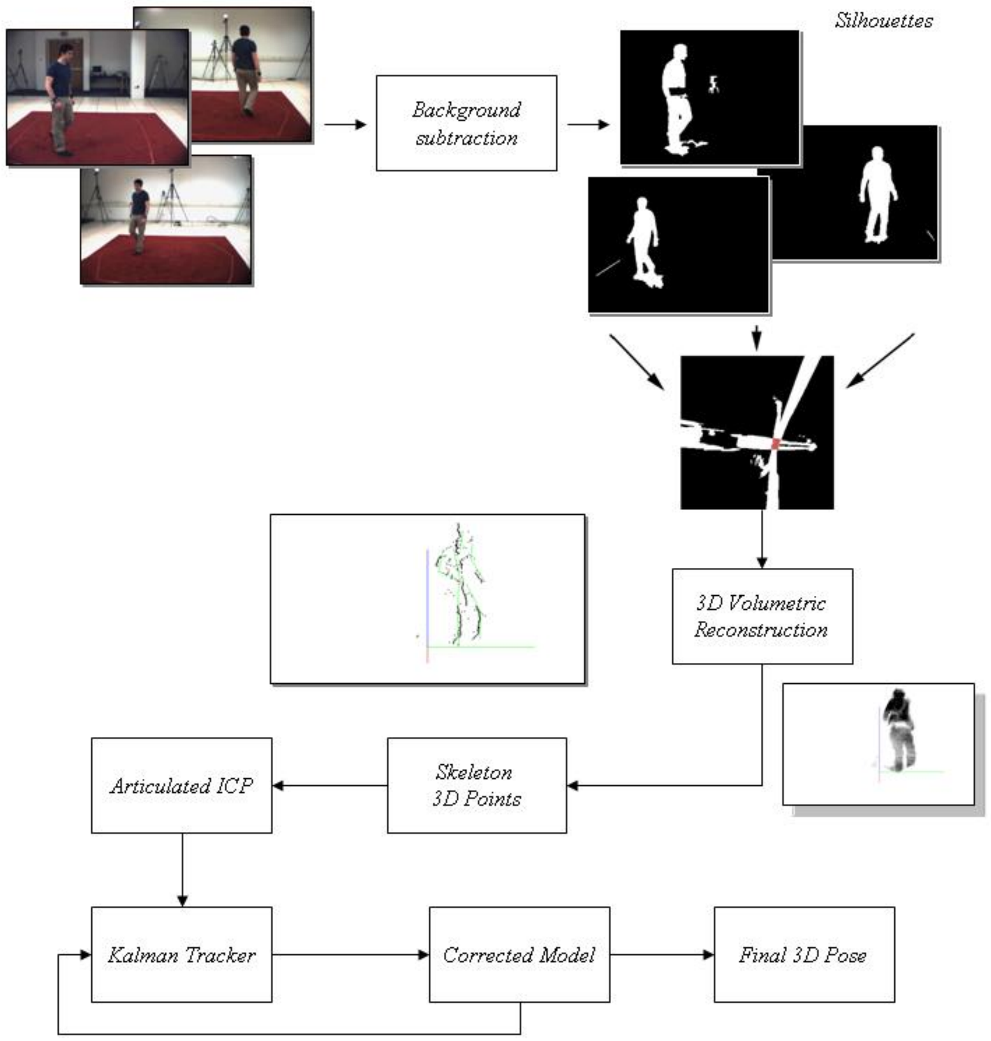


Fig. 7.1. A graphical summary of the algorithm pipeline.

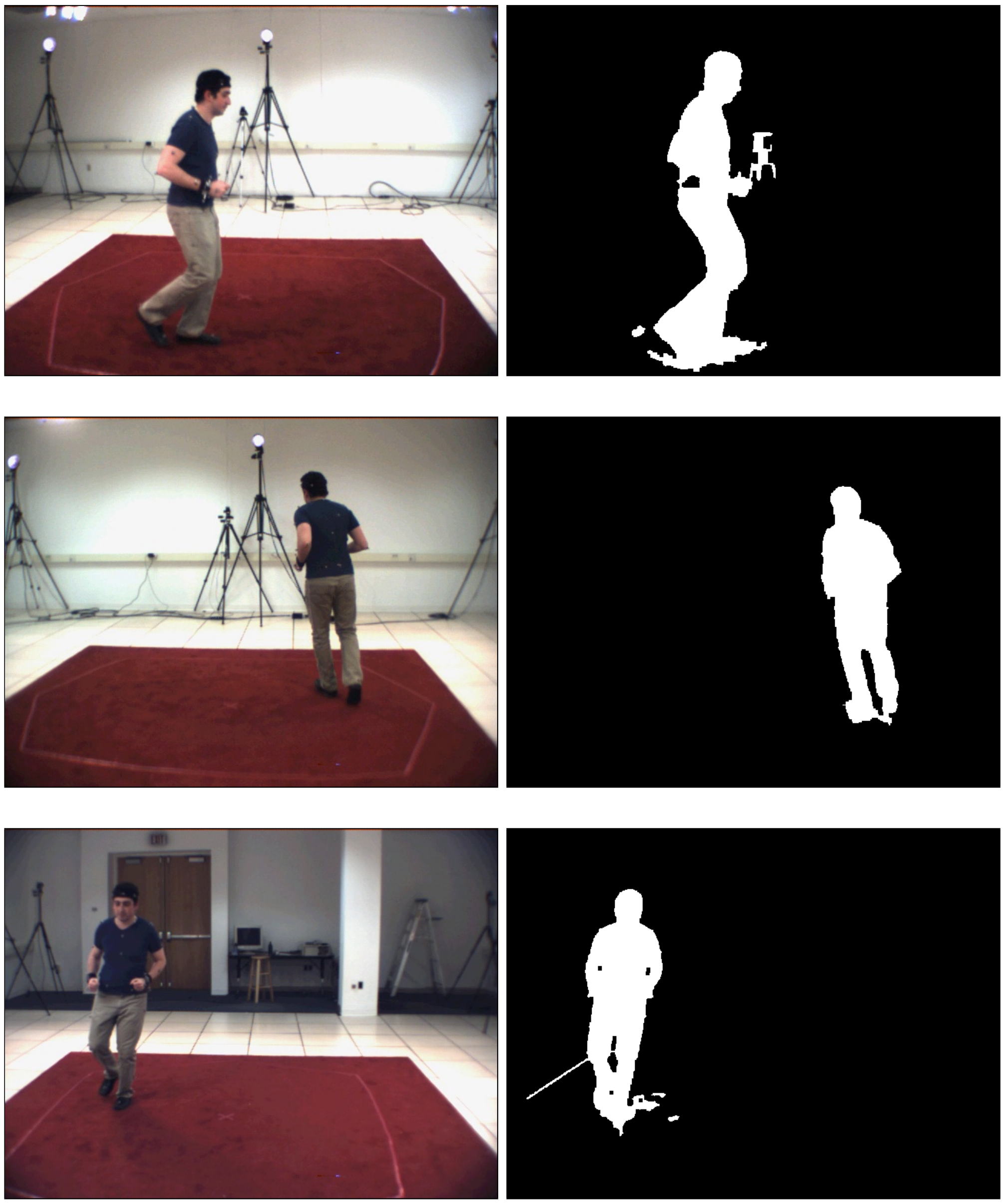


Fig. 7.2. Sample frames of *"S2 Jog"* and silhouettes.

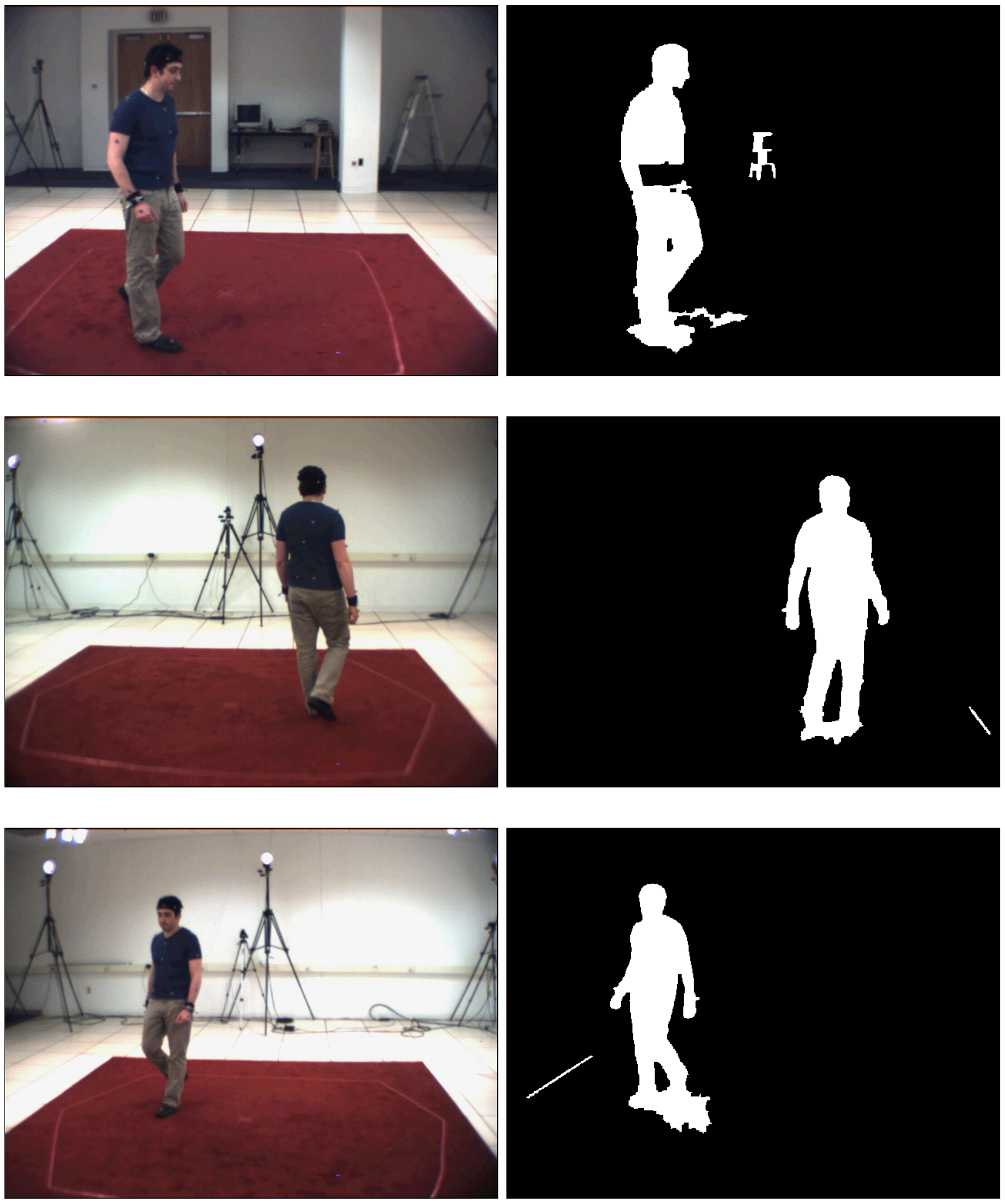


Fig. 7.3. Sample frames of *"S2 Walk"* and silhouettes.

In Figure 7.8 some of the frames taken by the dataset during the tracking are shown.

In the first two of our works [61, 62] the validation of the algorithm was done by comparing the angles of the ground-truth with the angles of the computed model.

In the first of our works we didn't use the Kalman filter technique. The results after introducing the Kalman filter are shown in Table 7.1. Note that in all the experiments involving the Kalman filter the metric used to compute the error is the RMSE as explained in section 7.2.

Figures 7.4, 7.5, 7.6, 7.7 report the ground truth and estimated joint angles of the angles in two of the sequences used. It can be seen that the estimated angles follows fairly closely the ground truth. There some spikes where the error grows but the tracker is able to recover in the subsequent frames.

This results are remarkable if one considers the coarseness of the volumetric reconstruction, due to the small number of cameras (three) and the poor quality of image silhouettes.

The magnitude of the error is comparable with results reported in the state-of-the art literature [7, 84]. For the "leaves" limb, as head, lower arm and calf the angles error is usually bigger than the other, as expected.

In these experiments we also measured the error of the reference points comparing them to the ground-truth 3D points. The results are shown in Table 7.2. Plots of the errors measured in mm are shown in Figures 7.9, 7.10, 7.11, 7.12, 7.13.

The current matlab implementation takes about 30 seconds to process a frame on a laptop with an Intel Core Duo Processor T2250. However, the algorithm is still in a prototypal stage, and thanks to design choices that favored fast (in principle) algorithms, we are confident that there is lot of space for improvement.

7.2 Evaluation Metrics

For human motion tracking and pose estimation there are a lot of metrics used in literature to measure the errors. The most common are the joint-angle distance and the tip position distance. For the experiments done using the HumanEva dataset we used both of the metrics to evaluate the goodness of the method. In the first two works [61, 62] we used for a quantitative comparison the following angular error for each joint, in each frame of the sequence:

$$e(R_1, R_2) = \angle(R_1 R_2^\top) \tag{7.1}$$

where $\angle(\cdot)$ denotes the angle of the axix-angle representation of the rotation, and can be computed with $\angle(R) = \arccos((\mathrm{tr}(R) - 1)/2)$.

After the introduction of the Kalman filter because of the fact we knew also the state of the model, that is made of angles expressed in radians, we have used a different metrics, most suitable to compare the results with other works in literature. The evaluation it is performed now using the *Root Mean Squared Error* (RMSE), which has the property to be expressed in the same unit of the quantity being estimated. The formula used for compute this error is:

$$\mathbf{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{n}} \tag{7.2}$$

For the position error we propose a simply measure based on the locations of joints and limbs endpoints and reference points. THe reference points used were described before in Chapter 5. We use as the model points of the HumanEva dataset as reference points. Assume that these ground truth points are $X = \{x_1, x_2, \cdots, x_N\}$, where $x_i \in \mathbb{R}^3$ is the 3D position of the i-th point in the world. The error for each reference points can be measured as:

$$d(x_i) = \sqrt{\sum_{j=1}^{M} \frac{x_{ij} - \hat{x}_{ij}}{M}} \tag{7.3}$$

where M is the number of frames of the sequence.

7.3 Conclusion

The results show that the obtained performance are good and informative for a tracking algorithm, even if at this time not computed in real time. It is possible to see that even if sometimes a joints goes far from the real position it recovers in the next frames. This effect is due also for the application of the Kalman filter that helps to contain large variation in the angles between frames. Also in the position error plots it is possible to see that even if in some frames there are a large error, the algorithm can recover in the next frames due to the ability to fit the model in a correct position.

The results are encouraging and application using home-made system with simple webcams could be tested in the future.

The future perspectives are to implement the system using faster algorithm and compiled code to obtain real time performance.

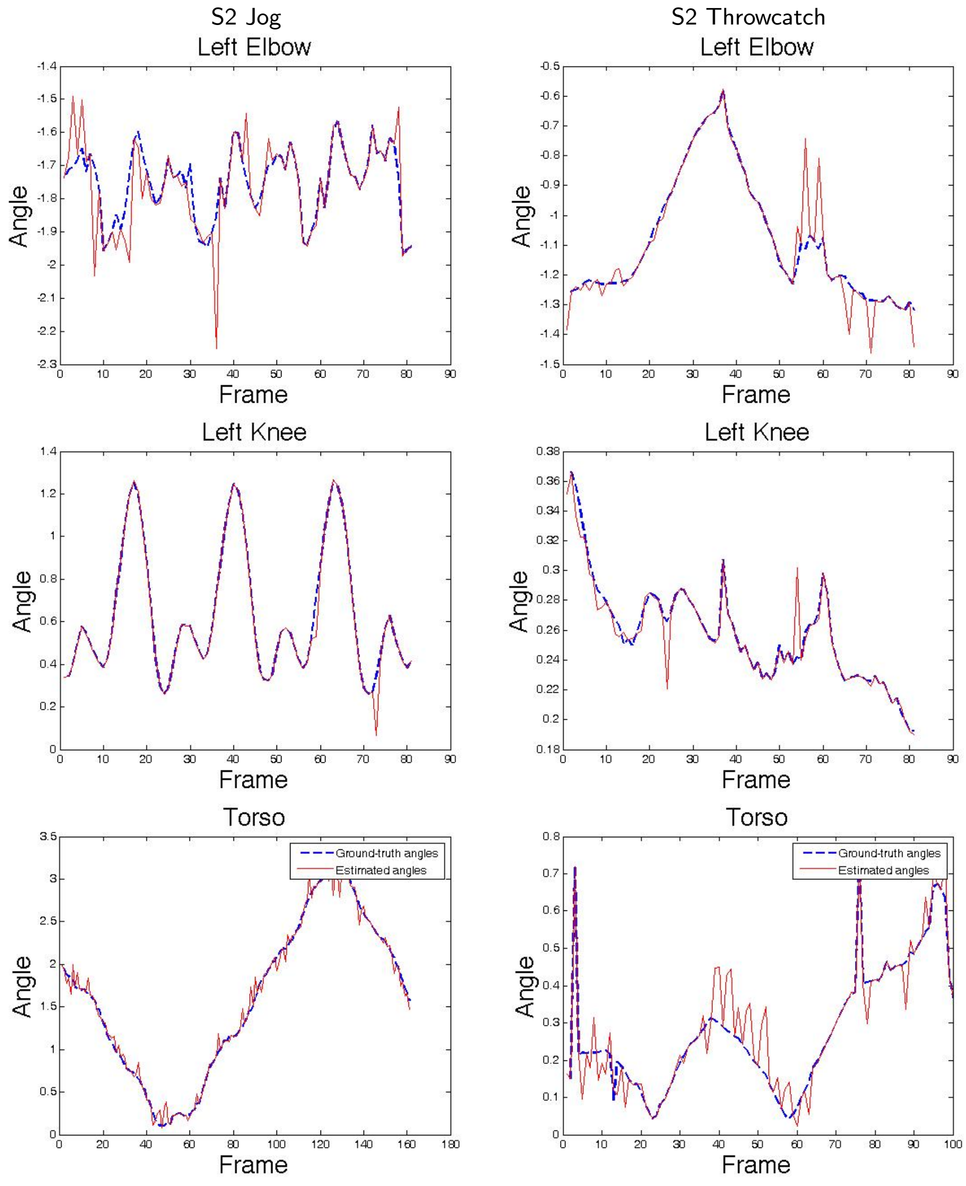


Fig. 7.4. Plots comparing ground truth and estimated joint angles of the torso, knee and elbow in two of the sequences used in the experiments.

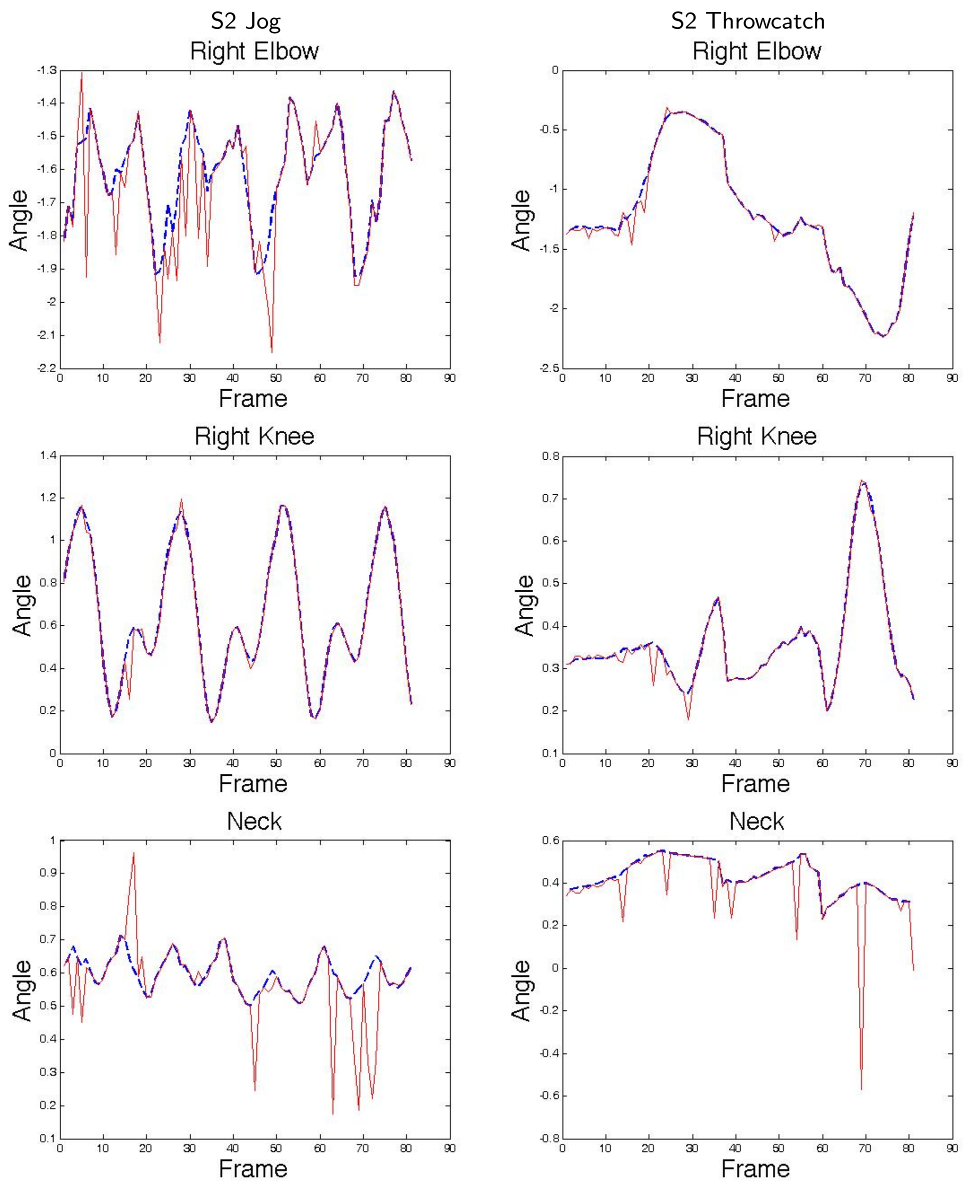


Fig. 7.5. Plots comparing ground truth and estimated joint angles of the Right Elbow, Right Knee and the Neck used in the experiments.

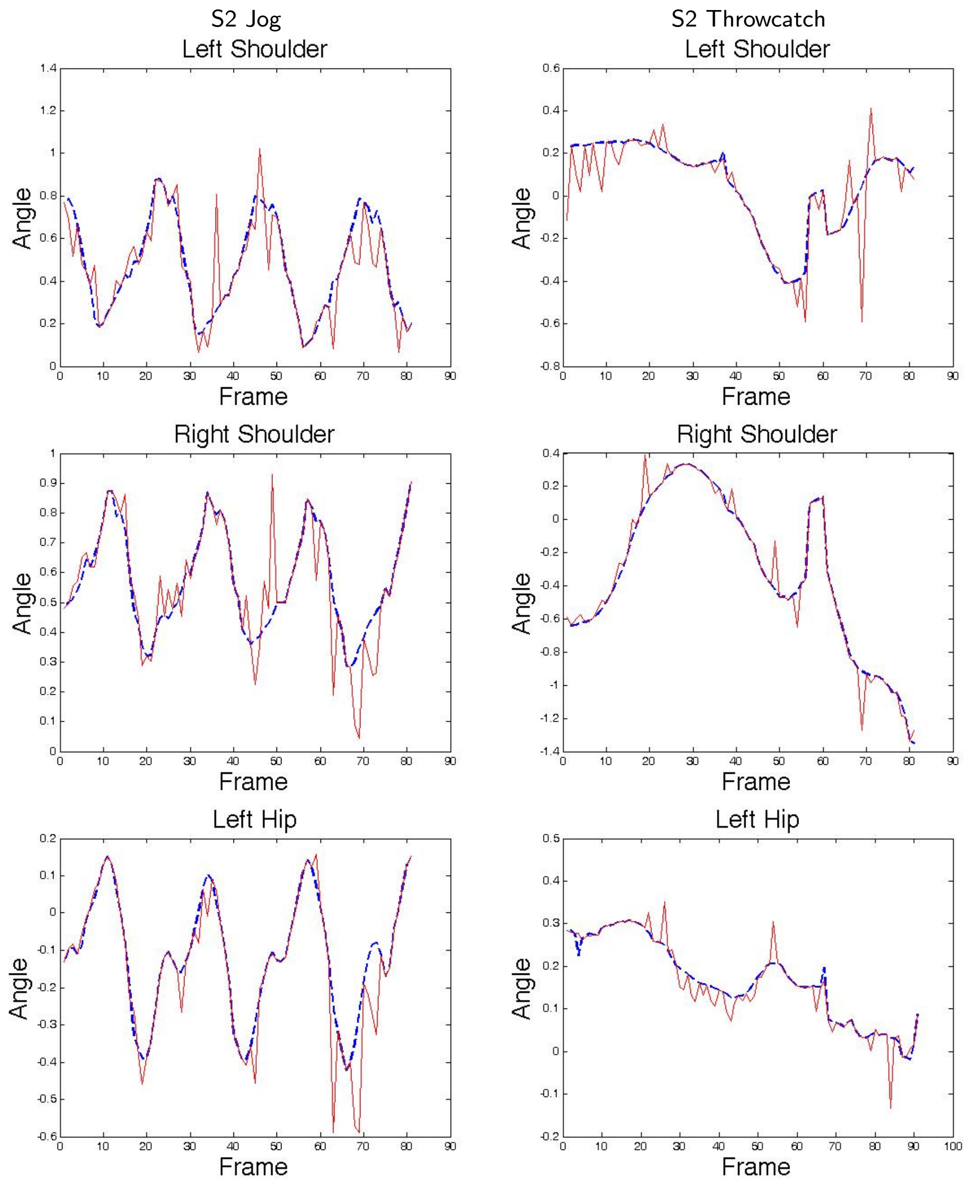


Fig. 7.6. Plots comparing ground truth and estimated joint angles of the Left and Right Shoulder and Left Hip in two of the sequences used in the experiments.

	SEQ7 - Subject 2 - Jog	*SEQ8 - Subject 2 - Throw-Catch*	*SEQ9 - Subject 2 - Gestures*	*SEQ10 - Subject 2 - Box*	*SEQ11 - Subject 3 - Walking*	*SEQ15 - Subject 3 - Box*
Torso	0.10	0.07	0.08	0.06	0.07	0.12
Neck(1)	0.11	0.12	0.15	0.09	0.56	0.09
Neck(2)	0.08	0.10	0.14	0.09	0.06	0.05
Left Shoulder(1)	0.18	0.07	0.12	0.09	0.08	0.08
Right Shoulder(1)	0.13	0.87	0.06	0.28	0.06	0.66
Left Shoulder(2)	0.15	0.10	0.12	0.09	0.04	0.07
Right Shoulder(2)	0.10	0.07	0.12	0.10	0.08	0.13
Left Shoulder(3)	0.33	0.13	0.36	0.09	0.14	0.21
Right Shoulder(3)	0.97	0.70	0.94	1.4	1.4	1.2
Left Elbow	0.08	0.04	0.07	0.06	0.04	0.05
Right Elbow	0.10	0.05	0.11	0.14	0.11	0.13
Left Hip(1)	0.06	0.05	0.05	0.08	0.05	0.09
Right Hip(1)	0.07	0.05	0.05	0.06	0.05	0.06
Left Hip(2)	0.05	0.06	0.08	0.07	0.05	0.04
Right Hip(2)	0.05	0.06	0.09	0.07	0.03	0.05
Left Knee	0.05	0.01	0.02	0.08	0.05	0.01
Right Knee	0.03	0.03	0.03	0.09	0.04	0.04

Table 7.1. RMSE of the angular value expressed in radians.

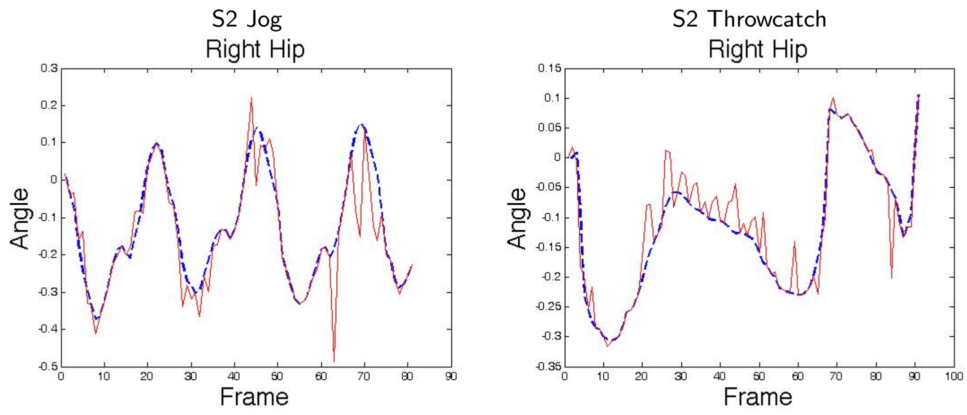


Fig. 7.7. Plots comparing ground truth and estimated joint angles of the Right Hip in two of the sequences used in the experiments.

	SEQ7 - Subject 2 - Jog	*SEQ8 - Subject 2 - Throw-Catch*	*SEQ9 - Subject 2 - Gestures*	*SEQ10 - Subject 2 - Box*	*SEQ11 - Subject 3 - Walking*	*SEQ15 - Subject 3 - Box*
$\mathbf{p}_1$	22	23	21	18	25	12
$\mathbf{p}_2$	33	28	32	30	27	20
$\mathbf{p}_3$	26	29	33	22	28	18
$\mathbf{p}_4$	35	33	40	26	30	23
$\mathbf{p}_5$	33	32	38	25	33	35
$\mathbf{p}_6$	29	27	29	23	27	25
$\mathbf{p}_7$	29	27	29	23	27	20
$\mathbf{p}_8$	73	44	49	43	35	32
$\mathbf{p}_9$	50	39	43	44	34	51
$\mathbf{p}_{10}$	33	29	32	35	33	31
$\mathbf{p}_{11}$	34	32	33	35	31	31
$\mathbf{p}_{12}$	57	46	57	44	38	35
$\mathbf{p}_{13}$	49	43	46	53	42	64
$\mathbf{p}_{14}$	49	36	47	50	47	41
$\mathbf{p}_{15}$	52	42	50	42	38	52

Table 7.2. RMSE of the position value expressed in mm.

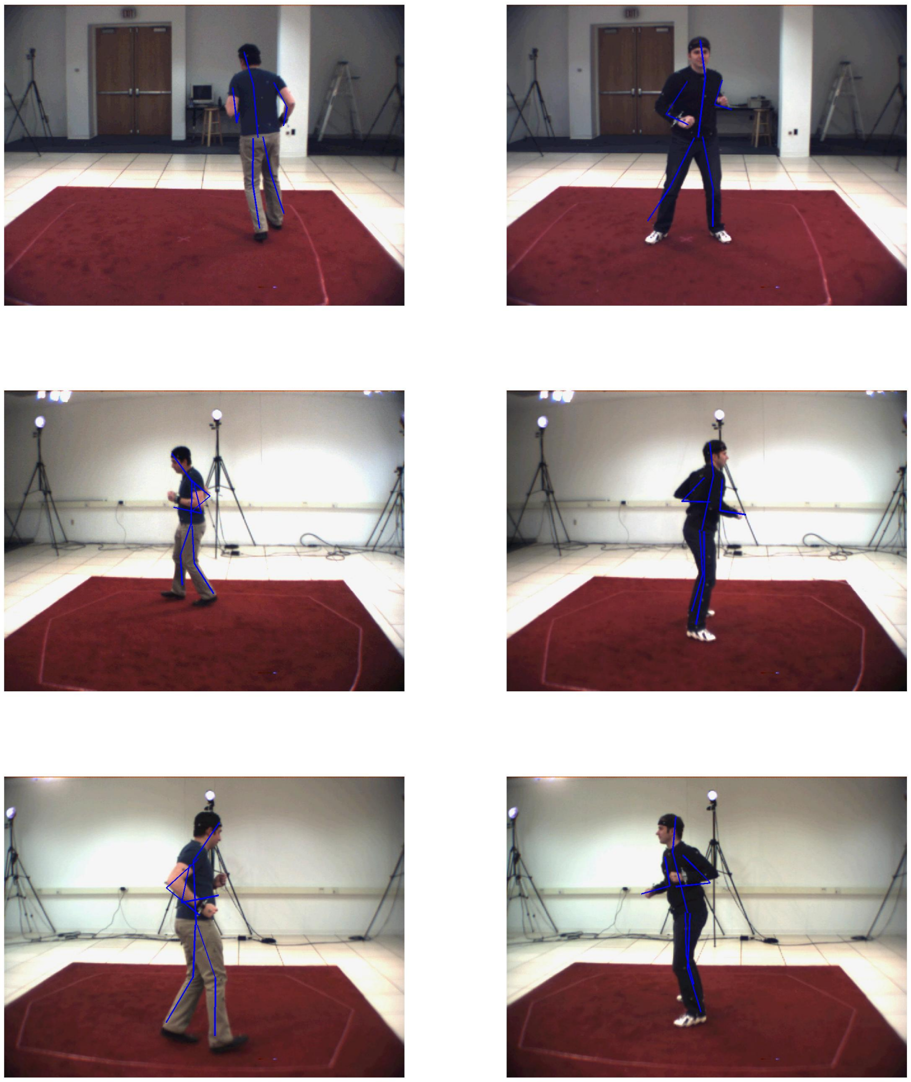


Fig. 7.8. Frames taken from the sequences with the stick skeleton model on them.

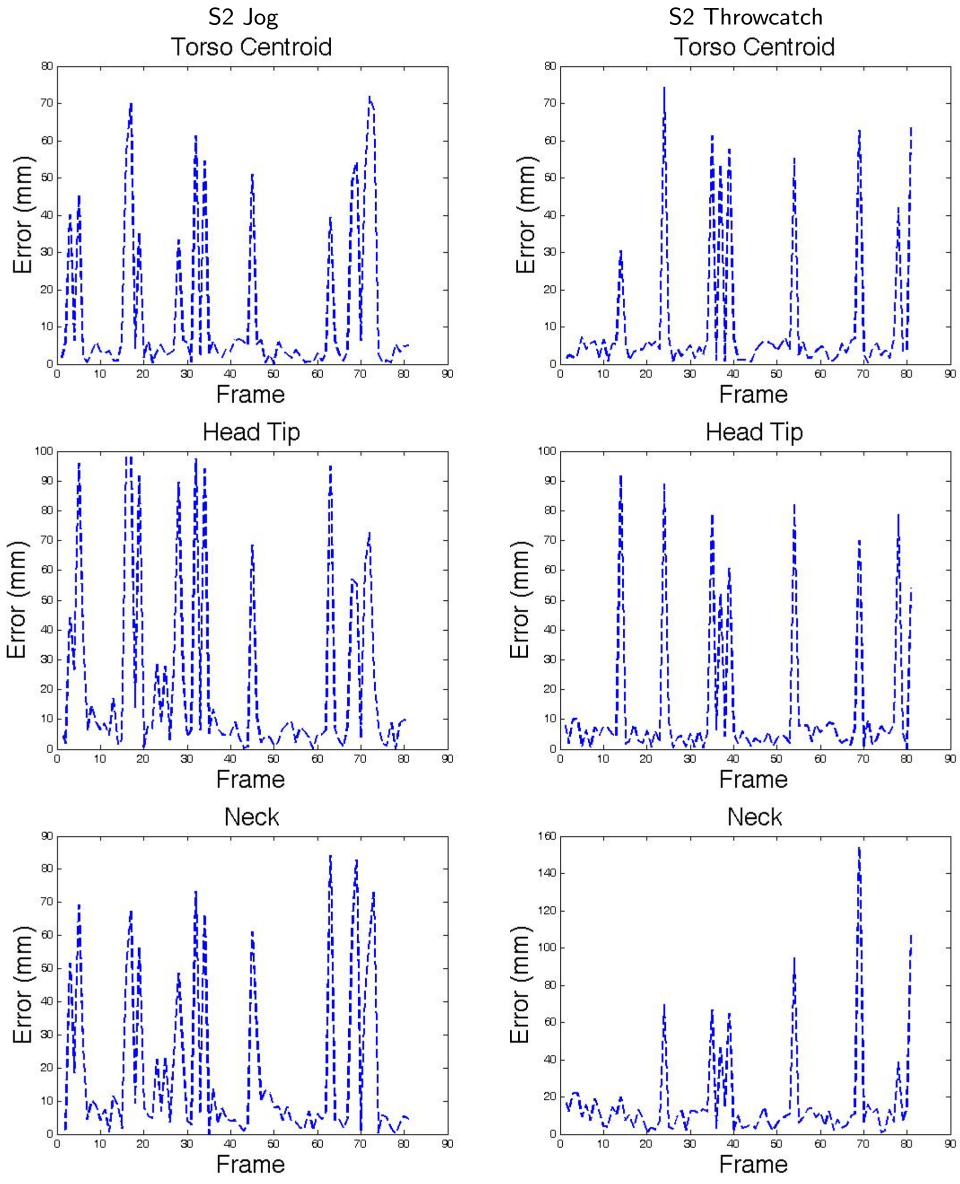


Fig. 7.9. Plots of the position error on two of the sequences used in the experiments.

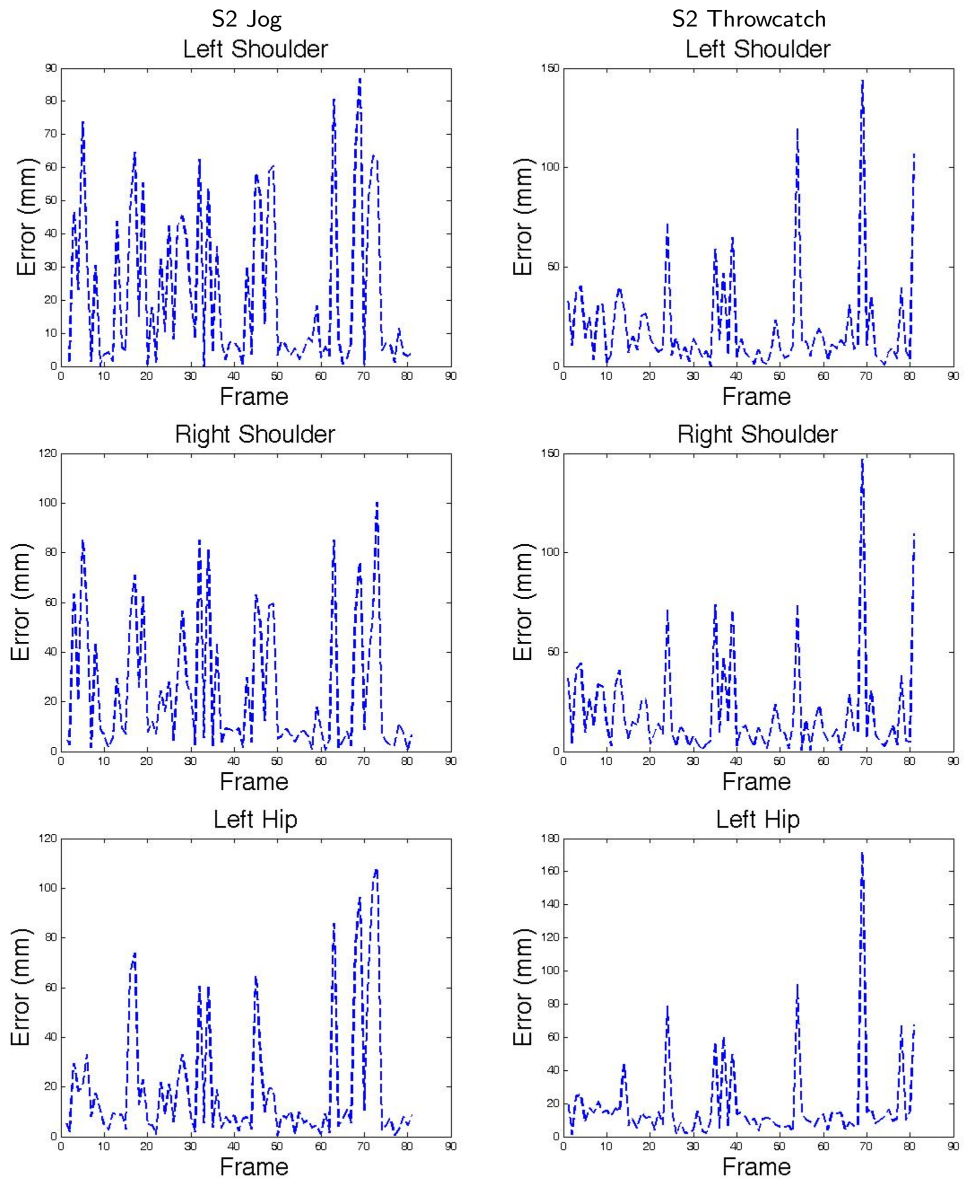


Fig. 7.10. Plots of the position error on two of the sequences used in the experiments.

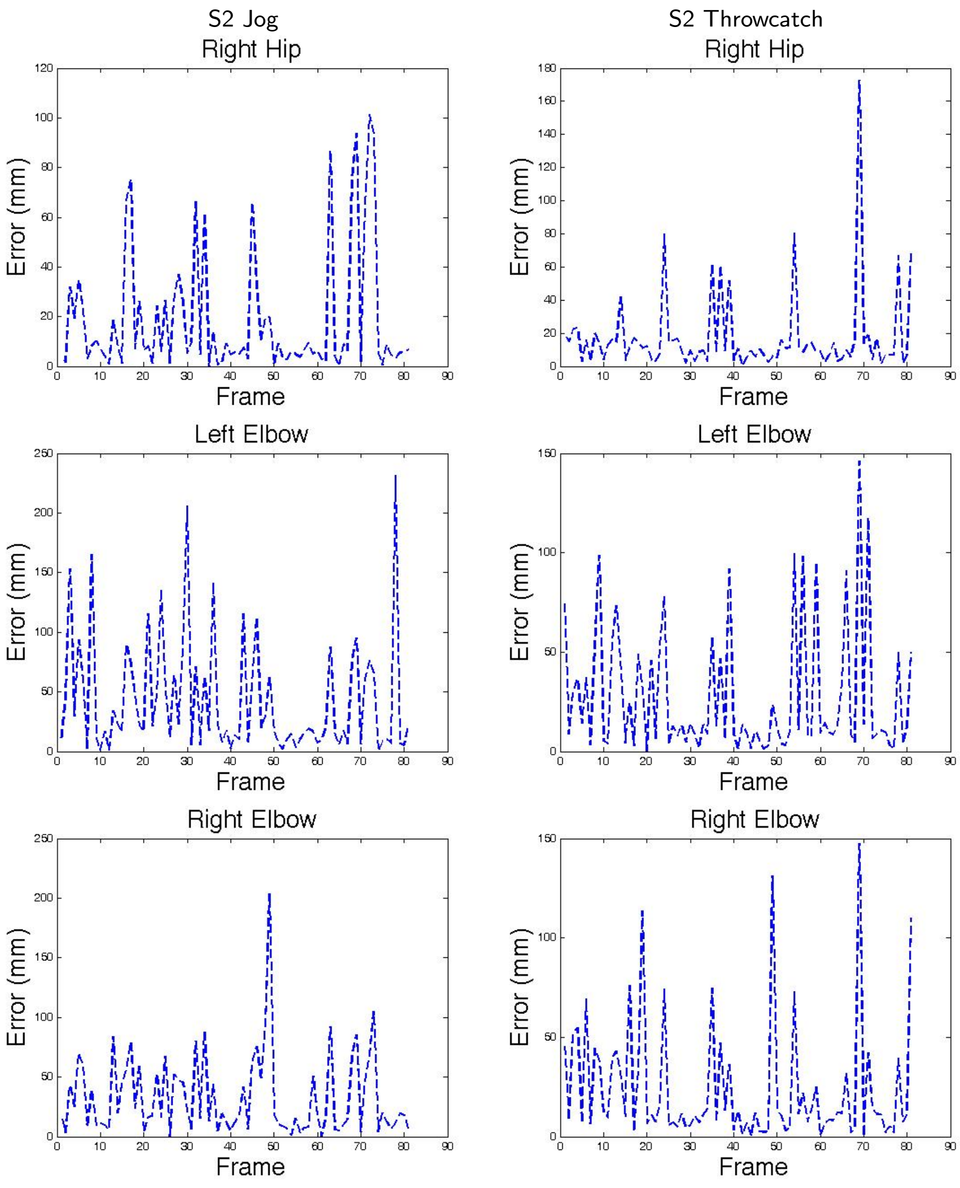


Fig. 7.11. Plots of the position error on two of the sequences used in the experiments.

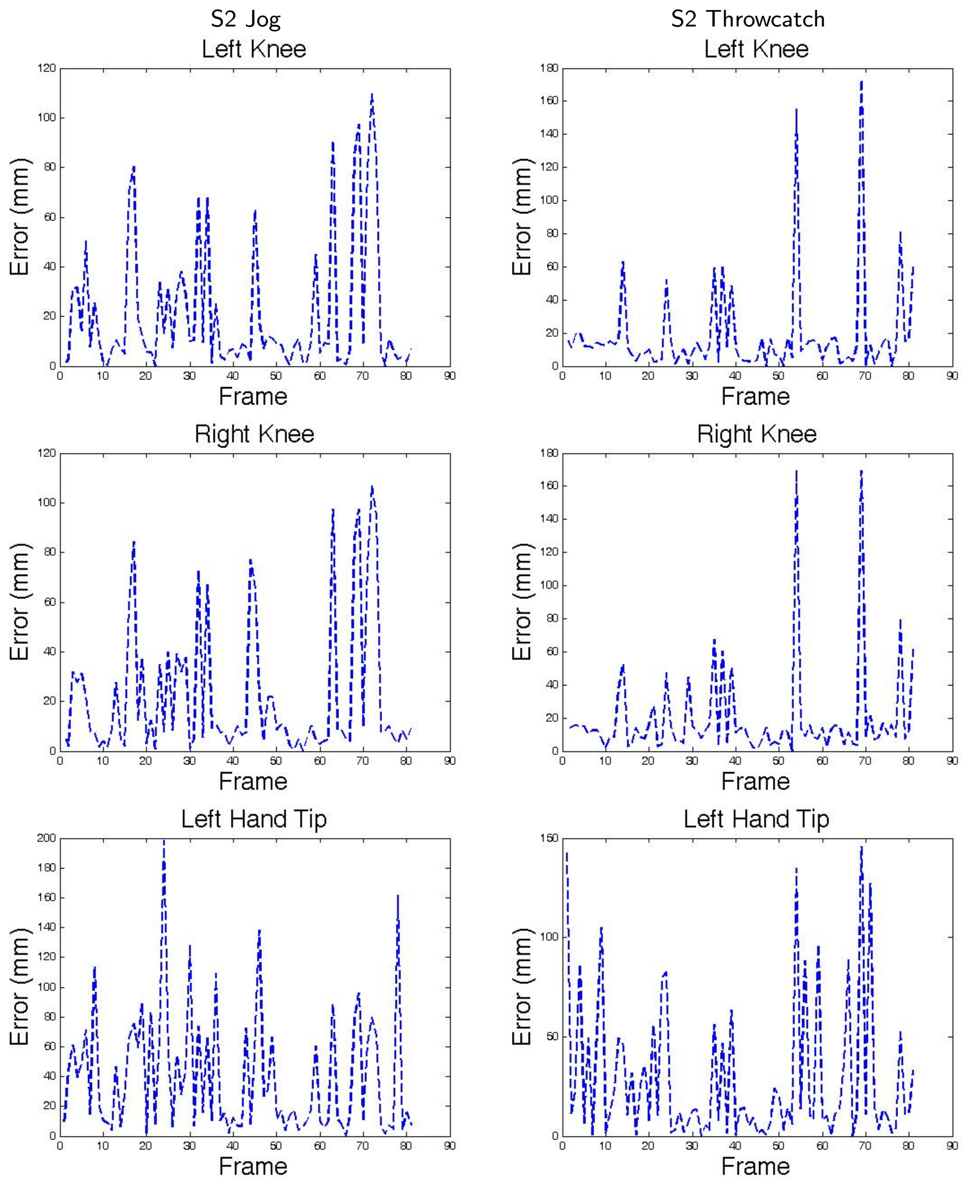


Fig. 7.12. Plots of the position error on two of the sequences used in the experiments.

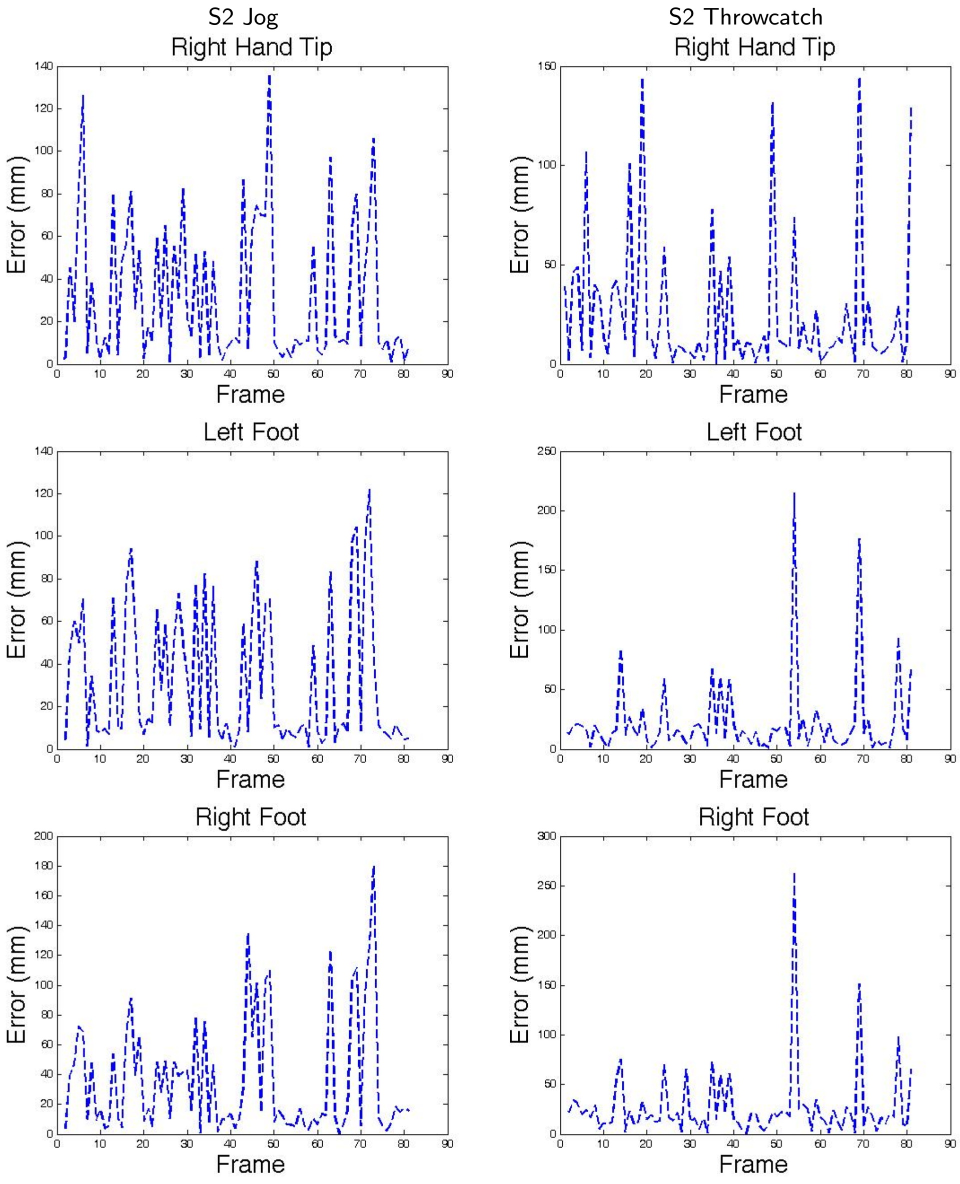


Fig. 7.13. Plots of the position error on two of the sequences used in the experiments.

8

Conclusions

This thesis proposed a new ICP-based algorithm for tracking articulated skeletal model of a human body. The proposed algorithm takes as input multiple calibrated views of the subject, computes a volumetric reconstruction and the centerlines of the body and fits the skeletal body model in each frame using a hierarchic tree traversal version of the ICP algorithm that preserves the connection of the segments at the joints. The proposed approach uses the kinematic constraints and an Extended Kalman Filter to track the body pose.

The first contribution is a new algorithm to find the skeletal points of a 3D volume. The algorithm using a slicing technique find the medial axis of a volume in a fast way using the graphic card processor and the texture units (see Chapter 3). This algorithm produce good results in quality and performance compared to other works in literature [27, 79].

Another contribution is the introduction of a new tracking strategy based on a hierarchical application of the ICP standard algorithm to find the match between a stick body model and a set of 3D points. The algorithm use a traversing version of ICP where also all the 3D points are weighted in such a way every limbs of the model can best fit on the right portion of the body (see Chapter 4).

The application of these techniques shown the feasibility of the method and the performances obtained in terms of quality of estimate pose are comparable with other works in literature.

The results presented here demonstrate the feasibility of the approach, which is is intended to be used in complete system for vision-based markerless human body tracking. Future work will aimed at optimizing the implementation, in order to achieve real-time performances. Moreover we are building our own system composed of five synchronized cameras, which we expect could provide a significant improvement on the quality of the volumetric reconstruction.

8.1 Publications

Some parts of this work have been published in conference proceedings. More specifically, large parts of Chapter 4 have been published in [61]

In [62], we introduced some improvements to the algorithm and tested it on more data.

Moreover, we have submitted a manuscript to EURASIP Journal on Image and Video Processing based on this thesis where we report the results showed in Chapter 7.

ACKNOWLEDGEMENTS

This work was partially supported by PRIN 2006 project 3-SHIRT. Thanks to A. Giachetti for inspiring discussions.

Part II

Articles

Tracking Stick Figures with Hierarchical Articulated ICP

D. Moschini, A. Fusiello
Dipartimento di Informatica, Università di Verona
Strada Le Grazie 15, 37134 Verona, Italy
andrea.fusiello@sci.univr.it

Abstract

This paper presents an ICP-based algorithm for tracking an articulated skeletal model of the human body (stick figure) in 3D. The data are 3D points distributed roughly around the limbs' medial axes. The algorithm fits each stick to a limb in a hierarchical fashion, traversing the body's kinematic chain, while preserving the connection of the sticks at the joints. Experimental results illustrate the algorithm.

1 Introduction

Tracking or capturing the motion of a human body is a problem that has a long history in Computer Vision (see [11] for a survey) and several real-world application, such as human-computer interfaces, motion transfer, animation of virtual characters, activity/gesture/gait recognition, biomechanical studies. Marker-based commercial systems are available that works at very high frame rates and very high precision. While it is out of doubt that such speed/accuracy combination is necessary in biomechanics, it is questionable whether it is needed when animating a virtual character in a videogame or building a user-interface. There is therefore a niche for less expensive systems that work markerless at a reduced speed (up to 100 Hz). In this paper we present part of an ongoing project aimed at building a system with those characteristics.

The literature on markerless body tracking in three dimensions can be broadly split into two groups: those using a stick model for the human body [5, 9], roughly corresponding to its skeleton, and those using a full 3D model of the body's shape, in the form of a polygonal mesh or a volumetric model [2, 8, 12]. Since we aim at a real-time system, we are forced to work with a stick model. Indeed, a stick (or skeletal) model has fewer dependencies on anthropometric parameters than a shape model and can be tracked much faster because of its simplicity.

We use an approach based on the well-known Iterative Closest Point (ICP) algorithm [4]: the model is an articulated stick figure representing the body and it's kinematics, the data are 3D points distributed roughly around the medial axes of the limbs. The data are registered to the model using a hierarchical approach that proceeds by traversing the kinematic chain.

Differently from [12] we do not enforce joints constraints *a-posteriori* (thereby interfering with the result of ICP) but *during* the registration process. To the best of our knowledge this is the only ICP-based approach with this feature. Other approaches based on the EM algorithm enforce the joint constraints, but they are much more computation intensive than ICP.

2 Background

2.1 Human Body Model

In this section we describe the articulated model representing the human body pose we used in the paper. It consists of a kinematic chain of ten sticks and nine joints, as depicted in Figure 1. The torso is at the root of tree, children represents limbs, each limb being described by a fixed-length stick and the corresponding rotation from its parent. Hence, the motion of one body segment can be described as the motion of the previous segment in the kinematic chain and an angular motion around a body joint. Only the torso contains a translation that accounts for the translation of the whole body. Rotations are represented with 3×3 matrices. For the sake of simplicity, each joint has three degrees of freedom and there are no limits on the angles.

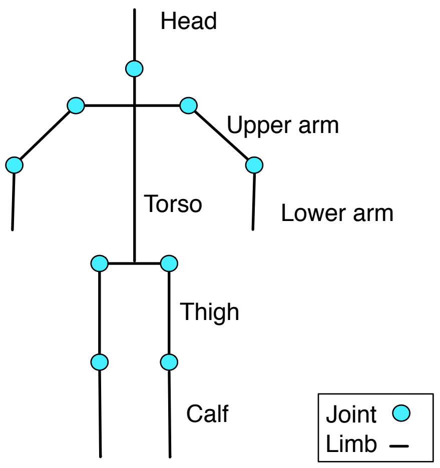


Figure 1: The stick figure body model.

2.2 Iterative Closest Point

The Iterative Closest Point (ICP) [6, 4] is customary used for registration of 3D sets of points. The standard algorithm estimates the rigid motion between a given set of $3D$ data points and a set of $3D$ model points. In summary:

Algorithm 1 ICP

Input: two sets of p 3D points, the data $\{\mathbf{a}_i\}_{i=1\dots p}$ and the model $\{\mathbf{b}_i\}_{i=1\dots p}$
Output: the rigid motion T that brings the data onto the model

```
1. For each point $\mathbf{a}_i$ of the data set find the closest point $\mathbf{b}_i$ in the model set.
2. Given these putative correspondences, estimate the global motion transformation $T$
   between all the points by solving an Extended Orthogonal Procrustes Problem (see
   below).
3. Apply the transformation $T$ to the data points.
4. If the distance between the two sets is less than a given threshold then quit, other-
   wise repeat from step 1.
```

The *Extended Orthogonal Procrustes Problem* (EOPP) [13] can be stated as follows: transform a given matrix A into a given matrix B by a similarity transformation (rotation, translation and scale) in such a way to minimize the sum of squares of the residual matrix. For reason that will be clear in the following, we consider instead the *Weighted Extended Orthogonal Procrustes Problem* (WEOPP) problem, with weights on the points. In formulae:

$$\arg\min_R \left\| (cRA + \mathbf{t}\mathbf{u}^\top - B)W \right\|_F^2 \quad \text{subject to } R^T R = I \tag{1}$$

where matrices A and B are $(3 \times p)$ matrices containing p corresponding point in 3-D space, R is (3×3) orthogonal rotation matrix, $\mathbf{t}$ is a (3×1) translation vector, c is scale factor, $\mathbf{u}$ is a $p \times 1$ vector of ones, W is a $(p \times p)$ diagonal matrix weighting the p points, and $\|\cdot\|_F$ denotes the Frobenius norm.

The solution to the the problem (derived in [1]) is based on the Singular Value Decomposition (SVD). Let

$$UDV^\top = A_w \left(I_p - \frac{\mathbf{u}_w \mathbf{u}_w^\top}{\mathbf{u}_w^\top \mathbf{u}_w} \right) B_w^\top \tag{2}$$

be the SVD decomposition of the matrix on the right-hand side[1], where $A_w = AW$, $B_w = BW$, and $\mathbf{u}_w = W\mathbf{u}$. The sought transformation is given by (we omit the scale c that is not needed in our case):

$$R = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^\top \tag{3}$$

$$\mathbf{t} = (B_w - RA_w) \frac{\mathbf{u}_w}{\mathbf{u}_w^\top \mathbf{u}_w} \tag{4}$$

The diagonal matrix in (3) is needed to ensure that the resulting matrix is a rotation [7]

The *Weighted Orthogonal Procrustes Problem* (WOPP) problem is a special case of WEOPP and the solution can be derived straightforwardly by setting $\mathbf{u} = \mathbf{0}$.

[1]Please note that $A\frac{\mathbf{u}_w \mathbf{u}_w^\top}{\mathbf{u}_w^\top \mathbf{u}_w}$ is a matrix of the same size as A with identical columns, each of them equal to the centroid of the points contained in A.

3 Hierarchical Articulated ICP

This section describes our contribution, namely the Hierarchical Articulated ICP algorithm for registering an articulate stick model to a cloud of points. We assume that the data are 3D points distributed roughly around the medial axes of the body's segments. They can reasonably come from the skeletonisation of a volumetric reconstruction of the body, as in [9, 5, 10]

The data are registered to the model using a hierarchical approach that starts from the torso and traverse the kinematic chain down to the extremities. At each step ICP computes the best rigid transformation of the limb that i) fits the data and ii) satisfy the kinematic constraints (namely, that the limb is connected to its ancestor through the joint) .

The closest point search works from the data to the model, by computing for each data point its closest point on the body segments. Only the matches with the current segment are considered, all the other should be – in principle – discarded.

However, the rotation in 3D space of a line segment cannot be computed unambiguously, for the rotation around the axis is undetermined. In order to cope with this problem we formulate a *Weighted* Extended Orthogonal Procrustes Problem and give a small but non-zero weight also to points that match the descendants of the current segment. In this way they contribute to constrain the rotation around the segment axis. Think, for example, of the torso: by weighting the points that match the limbs as well, even if they are still to be aligned, the coronal (aka frontal) plane can be recovered.

This is the complete algorithm described step by step:

Algorithm 2 HIERARCHICAL ARTICULATE ICP

Input: The model $\mathscr{S}$ composed by segments and the data set $\mathscr{A}$ of 3D points
Output: a set of rigid motions (referred to the kinematic chain) that brings the model onto the data

1. Traverse the body model tree structure using a level-order or a preorder traversal method.
2. Let $s_j \in \mathscr{S}$ be the current body segment.
3. Compute the closest points:
   (a) For each data point $\mathbf{a}_i \in \mathscr{A}$ and for each segment $s_\ell \in \mathscr{S}$ compute its projection $\mathbf{p}_{i\ell}$ onto the line containing s_ℓ ;
   (b) if $\mathbf{p}_{i\ell} \in s_\ell$ then add $\mathbf{p}_{i\ell}$ to $\mathscr{M}$ (the set of the closest-point candidates), otherwise add the endpoint of s_ℓ to $\mathscr{M}$.
   (c) Find $\mathbf{b}_i$, the closet point to $\mathbf{a}_i$ in $\mathscr{M}$.
4. Weight the points: If $\mathbf{b}_i$ belongs to s_j than its weight is 1, otherwise it is ε (chosen heuristically) for all the descendant and 0 for all the others.
5. If the distance of $\mathbf{b}_i$ to $\mathbf{a}_i$ is above a given threshold then the weight is set to 0.
6. Solve for the transformation of s_j with ICP using WEOPP for the torso and WOPP (only rotation) for the limbs.

The output of the algorithm represents the pose of the body. In a tracking framework, the pose obtained at the previous time-step is used as the initial pose for the current frame.

4 Experimental Results

The body tracker presented in section 4 has been implemented and tested on sequences taken from the HumanEva-I dataset [14]. All the sequences in HumanEva-I have been calibrated using the Vicon's proprietary software and the motion data saved in the common `c3d` file format. The dataset contains multiple subjects performing a variety of actions like walking, running, boxing, etc. In particular we used the sequences called *"S1 Box 3"*, *"S2 Throwcatch 3"* and *"S3 Jog 1"*.

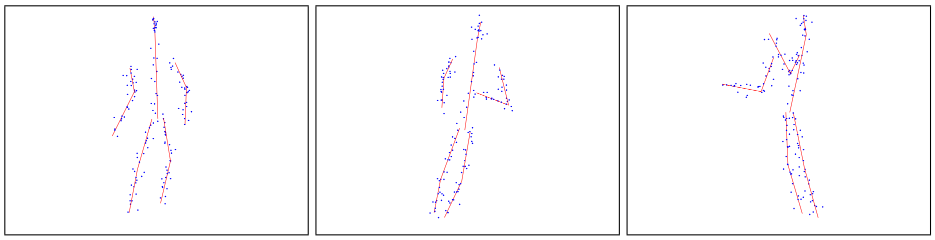


Figure 2: Sample frames of the synthetic data used in the experiments.The ground-truth stick figure and the data points corrupted by noise are shown.

The synthetic input data for our experiments has been created as follows (see Fig. 2). First the sequences have been sub-sampled at $40fps$ instead of the original $120fps$. Then, from each frame of the `c3d` file, reference points have been extracted and used to fit the skeletal model. The segments of the model are then sampled uniformly to obtain about 300 3D points. These points are finally perturbed using a Gaussian noise with an amplitude of half the hips distance.

Validation of the tracking is done by comparing the angles of the ground-truth with the angles of the computed model. Figure 3 reports the ground truth and estimated joint angles of the torso, right shoulder and right elbow in the three sequences. It can be seen that the estimated angles follows closely the ground truth, even without temporal smoothing. No significant mistracking occours.

For a quantitative comparison we computed the following angular error for each joint, in each frame of the sequence:

$$e(R_1, R_2) = \angle(R_1 R_2^\top) \tag{5}$$

where $\angle(\cdot)$ denotes the angle of the axix-angle representation of the rotation, and can be computed with $\angle(R) = \arccos((\mathrm{tr}(R) - 1)/2)$.

Mean and standard deviation of the error are shown in Table 1. The magnitude of the error is comparable with results reported in the state-of-the art literature [3, 15]. For the "leaves" limb, as head, lower arm and calf the angles error is usually bigger than the other, as expected.

The algorithm has been implemented in MATLAB, hence the performances are far from the desired real-time: it takes about 3.5 seconds to process a frame on a laptop with an Intel Core Duo Processor T2250.

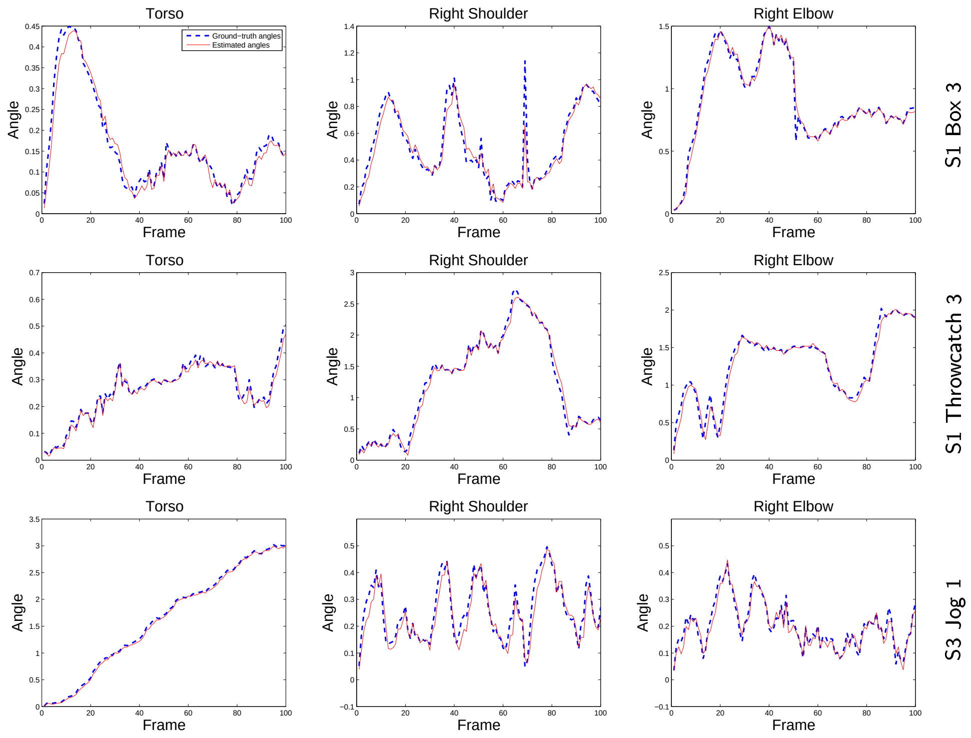


Figure 3: Plots comparing ground truth and estimated joint angles of the torso, right shoulder and right elbow in the three sequences used for the experiments (the sequence name is on the right).

5 Conclusions and Future Work

This paper has proposed a new ICP-based algorithm for tracking articulated skeletal model of a human body. The proposed algorithm takes as input $3D$ data points distributed around the torso and limbs medial axes. It fits the skeletal body model in each frame using a hierarchic tree traversal version of the ICP algorithm that preserves the connection of the segments at the joints. The proposed approach uses only the kinematic constraints and no other assumptions are made on the position of the body. This implies that we can recognized potentially all the body configuration.

The synthetic results presented here demonstrate the feasibility of the approach, which is is intended to be used in complete system for vision-based markerless human body tracking. Therefore, future work will consider improving performances by including a Kalman filter to smooth the estimates and provide a better prediction of the body's pose (which should allow ICP to converge in less iterations) and implementing the front-end of the pipeline, i.e., the shape from silhouette and skeletonisation modules that feeds the algorithm presented here.

Acknowledgments. This paper was partially supported by PRIN 2006 project 3-SHIRT.

		Box	*Throwcatch*	*Jog*
Torso	*Mean*	0.032	0.024	0.029
	Std. dev.	0.017	0.012	0.014
Neck	*Mean*	0.152	0.207	0.232
	Std. dev.	0.078	0.110	0.268
Left shoulder	*Mean*	0.074	0.071	0.063
	Std. dev.	0.039	0.041	0.030
Right shoulder	*Mean*	0.102	0.073	0.061
	Std. dev.	0.051	0.067	0.029
Left hip	*Mean*	0.181	0.120	0.111
	Std. dev.	0.126	0.089	0.075
Right hip	*Mean*	0.376	0.088	0.105
	Std. dev.	0.308	0.060	0.089
Left elbow	*Mean*	0.070	0.056	0.049
	Std. dev.	0.045	0.038	0.028
Right elbow	*Mean*	0.064	0.060	0.049
	Std. dev.	0.040	0.040	0.027
Left knee	*Mean*	0.061	0.049	0.052
	Std. dev.	0.035	0.028	0.030
Right knee	*Mean*	0.066	0.043	0.052
	Std. dev.	0.055	0.025	0.032

Table 1: Mean and standard deviation of the errors (in radians) for each joint of the body nfor the three sequences used in the experiments.

References

[1] D. Akca. Generalized procrustes analysis and its applications in photogrammetry. Technical Report, ETH, Swiss Federal Institute of Technology Zurich, Institute of Geodesy and Photogrammetry, 2003.

[2] D. Anguelov, D. Koller, H.-C. Pang, P. Srinivasan, and S. Thrun. Recovering articulated object models from 3D range data. In *Proc. of the 20th conference on Uncertainty in Artificial Intelligence*, pages 18–26, Arlington, Virginia, United States, 2004.

[3] A. Ashbrook, R. B. Fisher, N. Werghi, and C. Robertson. Construction of articulated models from range data. In *Proc. British Machine Vision Conference*, pages 183–192, 1999.

[4] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[5] G. J. Brostow, I. Essa, D. Steedly, and V. Kwatra. Novel skeletal representation for articulated creatures. In *ECCV04*, pages Vol III: 66–78, 2004.

[6] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

[7] K. Kanatani. *Geometric Computation for Machine Vision*. Oxford University Press, 1993.

[8] S. Knoop, S. Vacek, and R. Dillmann. Modeling joint constraints for an articulated 3D human body model with artificial correspondences in ICP. In *Proc. 5th IEEE-RAS International Conference on Humanoid Robots*, pages 74–79, 2005.

[9] C. Ménier, E. Boyer, and B. Raffin. 3d skeleton-based body pose recovery. In *Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization and Transmission*, Chapel Hill (USA), June 2006.

[10] B. Michoud, E. Guillou, and S. Bouakaz. Human model and pose Reconstruction from Multi-views. In *International Conference on Machine Intelligence*, November 2005.

[11] 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*, 103(2-3):90–126, November 2006.

[12] L. Mundermann, S. Corazza, and T.P. Andriacchi. Accurately measuring human movement using articulated ICP with soft-joint constraints and a repository of articulated models. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, June 2007.

[13] P. Schönemann and R. Carroll. Fitting one matrix to another under choice of a central dilation and a rigid motion. *Psychometrika*, 35(2):245–255, June 1970.

[14] L. Sigal and M.J. Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical Report CS-06-08, Brown University, Department of Computer Science, 2006.

[15] Y. Sun, M. Bray, A. Thayananthan, B. Yuan, and P.H.S. Torr. Regression-based human motion capture from voxel data. In *Proc. British Machine Vision Conference*, pages I:277–286, 2006.

References

1. MATLAB Symbolic Math Toolbox 3.1.4.
urlhttp://www.mathworks.com/products/symbolic/.
2. Organic Motion Stage.
urlhttp://www.inition.co.uk/inition/product.php.
3. J. K. Aggarwal and Q. Cai. Human motion analysis: a review. pages 90–102, 1997.
4. J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Articulated and elastic non-rigid motion: A review. In *In Proc. IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 2–14, 1994.
5. D. Akca. Generalized procrustes analysis and its applications in photogrammetry. Technical Report, ETH, Swiss Federal Institute of Technology Zurich, Institute of Geodesy and Photogrammetry, 2003.
6. D. Anguelov, D. Koller, H.-C. Pang, P. Srinivasan, and S. Thrun. Recovering articulated object models from 3D range data. In *Proc. of the 20th conference on Uncertainty in Artificial Intelligence*, pages 18–26, Arlington, Virginia, United States, 2004.
7. A. Ashbrook, R. B. Fisher, N. Werghi, and C. Robertson. Construction of articulated models from range data. In *Proc. British Machine Vision Conference*, pages 183–192, 1999.
8. Norman I. Badler, Cary B. Phillips, and Bonnie Lynn Webber. *Simulating humans: computer graphics animation and control.* Oxford University Press, Inc., New York, NY, USA, 1993.
9. Alexandru O. Balan, Leonid Sigal, Michael J. Black, James E. Davis, and Horst W. Haussecker. Detailed human shape and pose from images. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2007.
10. Luca Ballan and Guido Maria Cortelazzo. Multimodal 3d shape recovery from texture, silhouette and shadow information. In *3DPVT '06: Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, pages 924–930, Washington, DC, USA, 2006. IEEE Computer Society.
11. Luca Ballan and Guido Maria Cortelazzo. Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. In *3DPVT*, Atlanta, GA, USA, June 2008.
12. Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and data association*, volume 179 of *Mathematics in Science and Engineering.* Academic Press Professional, Inc., San Diego, CA, USA, 1987.
13. A M Baumberg and D C Hogg. An efficient method for contour tracking using active shape models. In *In Proceeding of the Workshop on Motion of Nonrigid and Articulated Objects. IEEE Computer Society*, pages 194–199, 1994.

14. P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.
15. Bobby Bodenheimer, Chuck Rose, Seth Rosenthal, and John Pella. The process of motion capture: Dealing with the data. In D. Thalmann and M. van de Panne, editors, *Computer Animation and Simulation '97*, pages 3–18. Springer NY, September 1997. Eurographics Animation Workshop.
16. J.-Y. Bouguet. Camera calibration toolbox for matlab, 2005.
17. C. Bregler. Motion capture technology for entertainment [in the spotlight]. 24:158–160, 2007.
18. Christoph Bregler and Jitendra Malik. Tracking people with twists and exponential maps. Technical Report UCB/CSD-97-973, EECS Department, University of California, Berkeley, Nov 1997.
19. G. J. Brostow, I. Essa, D. Steedly, and V. Kwatra. Novel skeletal representation for articulated creatures. In *ECCV04*, pages Vol III: 66–78, 2004.
20. Fabrice Caillette and Toby Howard. Real-time markerless human body tracking with multi-view 3-d voxel reconstruction. In *In Proc. ISMAR*, pages 597–606, 2004.
21. Joel Carranza, Christian Theobalt, Marcus A. Magnor, and Hans-Peter Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 22(3):569–577, July 2003.
22. Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.
23. Kong Man Cheung, Takeo Kanade, J.-Y. Bouguet, and M. Holler. A real time system for robust 3d voxel reconstruction of human motions. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, volume 2, pages 714 – 720, June 2000.
24. Intel Corporation. Camera calibration toolbox, 2006.
25. D. Demirdjian, T. Ko, and T. Darrell. Constraining human body tracking. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1071, Washington, DC, USA, 2003. IEEE Computer Society.
26. J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 126–133 vol.2, 2000.
27. Tamal K. Dey and Jian Sun. Defining and computing curve-skeletons with medial geodesic function. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 143–152, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.
28. Cass Everitt. Projective texture mapping.
29. Cass Everitt, Ashu Rege, and Cem Cebenoyan. Hardware shadow mapping. In *In ACM SIGGRAPH 2002 Tutorial Course no.31: Interactive Geometric Computations*, pages 38–51, 2002.
30. King Sun Fu, R. C. Gonzalez, and C. S. G. Lee. *Robotics: control, sensing, vision, and intelligence*. McGraw-Hill, Inc., New York, NY, USA, 1987.
31. A. Fusiello, U. Castellani, L. Ronchetti, and V. Murino. Model acquisition by registration of multiple acoustic range views. In *Proceedings of the European Conference on Computer Vision*, pages 805–819. 2002.
32. D. M. Gavrila. The visual analysis of human movement: a survey. *Comput. Vis. Image Underst.*, 73(1):82–98, 1999.
33. Riccardo Giannitrapani and Vittorio Murino. Three-dimensional skeleton extraction by point set contraction. In *ICIP (1)*, pages 565–569, 1999.
34. Colin Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(2):285–339, 1991.
35. John P. Granieri and Norman I. Badler. Simulating humans in vr. In *The British Computer Society*. Academic Press, 1994.

36. Humanoid Animation Working Group. H-anim. `http://h-anim.org/`, January 2009.
37. Jin Gu, Terry Chang, Ivan Mak, S. Gopalsamy, Helen C. Shen, and M. M. F. Yuen. A 3d reconstruction system for human body modeling. In *CAPTECH '98: Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, pages 229–241, London, UK, 1998. Springer-Verlag.
38. L. Herda, P. Fua, R. Plänkers, D., R. Boulic, and D. Thalmann. Skeleton-based motion capture for robust reconstruction of human motion. In *Computer Animation*, Philadelphia, PA, May 2000.
39. D.D. Hoffman and B.E. Flinchbaugh. The interpretation of biological motion. 42(3):195–202, 1982.
40. Špela Ivekovič, Emanuele Trucco, and Yvan R. Petillot. Human body pose estimation with particle swarm optimisation. *Evol. Comput.*, 16(4):509–528, 2008.
41. Ioannis A. Kakadiaris and Dimitri Metaxas. 3d human body model acquisition from multiple views. In *International Journal of Computer Vision*, pages 618–623, 1995.
42. Kalman, Rudolph, and Emil. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.
43. R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction theory. pages 95–108, 1962.
44. K. Kanatani. *Geometric Computation for Machine Vision.* Oxford University Press, 1993.
45. Roland Kehl and Luc Van Gool. Markerless tracking of complex human motions from multiple views. *Comput. Vis. Image Underst.*, 104(2):190–209, 2006.
46. S. Knoop, S. Vacek, and R. Dillmann. Modeling joint constraints for an articulated 3D human body model with artificial correspondences in ICP. In *Proc. 5th IEEE-RAS International Conference on Humanoid Robots*, pages 74–79, 2005.
47. Martin Koschat and Deborah Swayne. A weighted procrustes criterion. *Psychometrika*, 56(2):229–239, June 1991.
48. A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, 1994.
49. A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, 1994.
50. Y.L. Lee, D.W. Kyoung, and K.C. Jung. Meshless parameterization for dimensional reduction integrated in 3d voxel reconstruction using a single pc. pages 321–333, 2007.
51. Ming Li, Marcus Magnor, and Hans peter Seidel. Hardware-accelerated visual hull reconstruction and rendering. In *In Graphics Interface 2003*, pages 65–71, 2003.
52. Robert Lissitz, Peter Schnemann, and James Lingoes. A solution to the weighted procrustes problem in which the transformation is in agreement with the loss function. *Psychometrika*, 41(4):547–550, December 1976.
53. Osama Masoud, Nikolaos P. Papanikolopoulos, and Senior Member. A novel method for tracking and counting pedestrians in real-time using a single camera. *IEEE Transactions on Vehicular Technology*, 50:1267–1278, 2001.
54. C. Ménier, E. Boyer, and B. Raffin. 3d skeleton-based body pose recovery. In *Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization and Transmission*, Chapel Hill (USA), June 2006.
55. B. Michoud, E. Guillou, and S. Bouakaz. Human model and pose Reconstruction from Multi-views. In *International Conference on Machine Intelligence*, November 2005.
56. Ivana Mikic, Mohan Trivedi, Edward Hunter, and Pamela Cosman. Human body model acquisition and tracking using voxel data. *IJCV*, 53:2003, 2003.
57. 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*, 103(2-3):90–126, November 2006.
58. Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Comput. Vis. Image Underst.*, 81(3):231–268, 2001.

59. Thomas B. Moeslund, Adrian Hilton, and Volker Kruger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, November 2006.
60. Tom Molet, Ronan Boulic, and Daniel Thalmann. A real time anatomical converter for human motion capture. In *In Eurographics Workshop on Computer Animation and Simulation*, pages 79–94, 1996.
61. D. Moschini and A. Fusiello. Tracking stick figures with hierarchical articulated icp. In *THEMIS '08: Proceedings of the First International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences*, pages 61–68, 2008.
62. D. Moschini and A. Fusiello. Tracking human motion with multiple cameras using an articulated model. In *Mirage 2009: Computer Vision / Computer Graphics Collaboration Techniques and Applications*, 2009.
63. L. Mundermann, S. Corazza, and T.P. Andriacchi. Accurately measuring human movement using articulated ICP with soft-joint constraints and a repository of articulated models. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, June 2007.
64. Pat M. Murray, Bernard A. Drought, and Ross C. Kory. Walking patterns of normal men. *J Bone Joint Surg Am*, 46(2):335–360, March 1964.
65. Richard M. Murray, Zexiang Li, and Shankar S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC, March 1994.
66. Atsushi Nakazawa, Hirokazu Kato, and Seiji Inokuchi. Human tracking using distributed vision systems. In *ICPR '98: Proceedings of the 14th International Conference on Pattern Recognition-Volume 1*, page 593, Washington, DC, USA, 1998. IEEE Computer Society.
67. Matti Niskanen, Edmond Boyer, and Radu P. Horaud. Articulated motion capture from 3-d points and normals. In Torr Clocksin, Fitzgibbon, editor, *British Machine Vision Conference*, volume 1, pages 439–448, Oxford, UK, September 2005. BMVA, British Machine Vision Association.
68. R. L. Ogniewicz and O. Kübler. Hierarchic Voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.
69. R. Plankers and P. Fua. Articulated Soft Objects for Video-based Body Modeling. In *International Conference on Computer Vision, Vancouver, Canada*, pages 394–401, 2001.
70. Ramprasad Polana and Al Nelson. Low level recognition of human motion (or how to get your man without finding his body parts. In *In Proc. of IEEE Computer Society Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.
71. K. Rohr and Arbeitsbereich Kognitive Systeme. Human movement analysis based on explicit motion models. In *M. Shah and R. Jain (Eds*, pages 171–198. Kluwer Academic Publishers, 1997.
72. M. Rossi and A. Bozzoli. Tracking and counting moving people. pages III: 212–216, 1994.
73. K. Schindler S. Pellegrini and D. Nardi. A generalisation of the icp algorithm for articulated bodies. In *Proceedings of the British Machine Vision Conference*, 2008.
74. P. Schnemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, March 1966.
75. P. Schnemann and R. Carroll. Fitting one matrix to another under choice of a central dilation and a rigid motion. *Psychometrika*, 35(2):245–255, June 1970.
76. Lorenzo Sciavicco and Bruno Siciliano. *Robotica Industriale.* McGraw-Hill, Inc., 2003.
77. Steven M. Seitz and Charles R. Dyer. Affine invariant detection of periodic motion. In *In Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 970–975, 1994.
78. Mubarak Shah. Motion-based recognition: A survey. *Image and Vision Computing*, 13:129–155, 1995.
79. Andrei Sharf, Thomas Lewiner, Ariel Shamir, and Leif Kobbelt. On–the–fly curve-skeleton computation for 3d shapes. In *Eurographics*, pages 323–328, Prague, september 2007.

80. L. Sigal and M.J. Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical Report CS-06-08, Brown University, Department of Computer Science, 2006.
81. L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *Advances in Neural Information Processing Systems*, volume 16, pages 1539–1546, 2003.
82. Cristian Sminchisescu and Bill Triggs. Estimating articulated human motion with covariance scaled sampling. *International Journal of Robotics Research*, 22:2003, 2003.
83. David J. Sturman. A brief history of motion capture for computer character animation. In *Character Motion Systems, Notes for SIGGRAPH 94, Course 9*, 1994.
84. Y. Sun, M. Bray, A. Thayananthan, B. Yuan, and P.H.S. Torr. Regression-based human motion capture from voxel data. In *Proc. British Machine Vision Conference*, pages I:277–286, 2006.
85. Alexandru Telea and Jarke J. van Wijk. An augmented fast marching method for computing skeletons and centerlines. In *VISSYM '02: Proceedings of the symposium on Data Visualisation 2002*, pages 251–ff, Aire-la-Ville, Switzerland, 2002. Eurographics Association.
86. Jon A Webb and J K Aggarwal. Structure from motion of rigid and jointed objects. *Artificial Intelligence*, 19:107–130, 1982.
87. Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 1995.
88. Christopher Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentl. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:780–785, 1997.
89. M. Yamamoto and K. Koshikawa. Human motion analysis based on a robot arm model. 1991.