Université de Montréal

Video-based Analysis of Gait Pathologies

par
Hoang Anh Nguyen

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en informatique

Decembre, 2014

RÉSUMÉ

L'analyse de la marche a émergé comme l'un des domaines médicaux le plus importants récemment. Les systèmes à base de marqueurs sont les méthodes les plus favorisées par l'évaluation du mouvement humain et l'analyse de la marche, cependant, ces systèmes nécessitent des équipements et de l'expertise spécifiques et sont lourds, coûteux et difficiles à utiliser. De nombreuses approches récentes basées sur la vision par ordinateur ont été développées pour réduire le coût des systèmes de capture de mouvement tout en assurant un résultat de haute précision. Dans cette thèse, nous présentons notre nouveau système d'analyse de la démarche à faible coût, qui est composé de deux caméras vidéo monoculaire placées sur le côté gauche et droit d'un tapis roulant. Chaque modèle 2D de la moitié du squelette humain est reconstruit à partir de chaque vue sur la base de la segmentation dynamique de la couleur, l'analyse de la marche est alors effectuée sur ces deux modèles. La validation avec l'état de l'art basée sur la vision du système de capture de mouvement (en utilisant le Microsoft Kinect) et la réalité du terrain (avec des marqueurs) a été faite pour démontrer la robustesse et l'efficacité de notre système. L'erreur moyenne de l'estimation du modèle de squelette humain par rapport à la réalité du terrain entre notre méthode vs Kinect est très prometteur: les joints des angles de cuisses (6,29° contre 9,68°), jambes (7,68° contre 11,47°), pieds (6,14° contre 13,63°), la longueur de la foulée (6.14cm rapport de 13.63cm) sont meilleurs et plus stables que ceux de la Kinect, alors que le système peut maintenir une précision assez proche de la Kinect pour les bras (7,29° contre 6,12°), les bras inférieurs (8,33° contre 8,04°), et le torse (8,69°contre 6,47°). Basé sur le modèle de squelette obtenu par chaque méthode, nous avons réalisé une étude de symétrie sur différentes articulations (coude, genou et cheville) en utilisant chaque méthode sur trois sujets différents pour voir quelle méthode permet de distinguer plus efficacement la caractéristique symétrie / asymétrie de la marche. Dans notre test, notre système a un angle de genou au maximum de 8,97° et 13,86° pour des promenades normale et asymétrique respectivement, tandis que la Kinect a donné 10,58°et 11,94°. Par rapport à la réalité de terrain, 7,64°et 14,34°, notre système a montré une plus grande précision et pouvoir discriminant entre les deux cas.

ABSTRACT

Gait analysis has emerged as one of the most important medical field recently due to its wide range of applications. Marker-based systems are the most favoured methods of human motion assessment and gait analysis, however, these systems require specific equipment and expertise, and are cumbersome, costly and difficult to use. Many recent computer-vision-based approaches have been developed to reduce the cost of the expensive motion capture systems while ensuring high accuracy result. In this thesis, we introduce our new low-cost gait analysis system that is composed of two low-cost monocular cameras (camcorders) placed on the left and right sides of a treadmill. Each 2D left or right human skeleton model is reconstructed from each view based on dynamic color segmentation, the gait analysis is then performed on these two models. The validation with one state-of-the-art vision-based motion capture system (using the Microsoft Kinect v.1) and one ground-truth (with markers) was done to demonstrate the robustness and efficiency of our system. The average error in human skeleton model estimation compared to ground-truth between our method vs. Kinect are very promising: the joints angles of upper legs (6.29° vs. 9.68°), lower legs (7.68° vs. 11.47°), feet (6.14° vs. 13.63°), stride lengths (6.14cm vs. 13.63cm) were better and more stable than those from the Kinect, while the system could maintain a reasonably close accuracy to the Kinect for upper arms (7.29° vs. 6.12°), lower arms (8.33° vs. 8.04°), and torso (8.69° vs. 6.47°). Based on the skeleton model obtained by each method, we performed a symmetry study on various joints (elbow, knee and ankle) using each method on two different subjects to see which method can distinguish more efficiently the symmetry/asymmetry characteristic of gaits. In our test, our system reported a maximum knee angle of 8.97° and 13.86° for normal and asymmetric walks respectively, while the Kinect gave 10.58° and 11.94°. Compared to the ground-truth, 7.64° and 14.34°, our system showed more accuracy and discriminative power between the two cases.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

APF	Annealed particle filter
DS	Deformable structure
EM	Expectation maximization
FJM	Fixed joint model
GAIMS	Gait measuring system
HOG	Histogram of oriented gradient
ICP	Iterative closest point
IR	Infrared
ISA	Interacting simulated annealing
LED	Light-emitting diode
LLM	Loose limb model
MCMC	Markov chain Monte Carlo
MOCAP	Motion capture
PF	Particle filter
PFICP	Particle filter + Iterative closest point
PS	Pictorial structure
RGB-D	Red-Green-Blue + Depth
RF	Random forest
RRF	Random regression forest
SDK	Software development kit
SVM	Support vector machine

NOTATION

N_s	Predefined number of sectors
s_i	Sector i
w_i	Weighting function for sector i
k	Number of clusters in k-means algorithm
$(\mu_{skin}, \sigma_{skin})$	mean and variance of the skin model
BB_1	Bounding box of the whole silhouette
BB_2	Bounding box of the half lower part of the silhouette
$< L_c, R_c >$	Left and right contours of the torso model
N_c	The total number of points of both L_c and R_c
$\{x_{li}, y_{li}\}_N$	x-component and y-component of point li in L_c
$\{x_{ri}, y_{ri}\}_N$	x-component and y-component of point ri in R_c
$\mathbf{k}$	Frame number $\mathbf{k}$
N_{skin}^{li}	Number of frames where an contour point x_{li}, y_{li} is skin
N_{skin}^{ri}	Number of frames where an contour point x_{ri}, y_{ri} is skin
$P_{forehead}$	The 2D forehead joint location of our 2.5D skeleton model
P_{chin}	The 2D chin joint location of our 2.5D skeleton model
$P_{shoulder}$	The 2D shoulder joint location of our 2.5D skeleton model
P_{hip}	The 2D hip joint location of our 2.5D skeleton model
P_{hand}	The 2D hand joint location of our 2.5D skeleton model
P_{elbow}	The 2D elbow joint location of our 2.5D skeleton model
P_{ankle}	The 2D ankle joint location of our 2.5D skeleton model
P_{ce}	The furthest possible elbow joint position along the forearm orientation
P_O	The origin coordinate of the 3D world space
cl^i	The closest edge point to the i-th point of the new left contour L_p that has the same y-value
ΔL	The distance between two consecutive 3D points

cr^i	The closest edge point to the i-th point of the new right contour R_p that has the same y-value
E_t	The cost function used to evaluate the new torso shape $< L_p, R_p >$
Map_{skin}	A binary image in which the value of skin pixels is 1, and the rest is 0
L_{LA}	Forearm segment length
L_{UA}	Upper arm segment length
$rectlai$	A fixed size rectangle used to capture forearm orientation.
w_{lar}	Width of $rectlai$
$List_{rectla}$	List of rectangle $rectlai$ generated with respect to the hand joint location P_{hand}
$Line_{LA}$	The 2D line represents the forearm orientation
S_{elbow}	List of elbow candidates
S^i_{elbow}	The i-the candidate of S_{elbow}
$S_{shoulder}$	List of shoulder candidates
$S^j_{shoulder}$	The j-the candidate of $S_{shoulder}$
$S_{candidates}$	List of combined candidates for constructing 2D arm
ρ_{ua}	Foreshortening level of upper arm
ρ_{la}	Foreshortening level of forearm
E^i_{UA}	Cost function to evaluate an arm candidate i of $S_{candidates}$
$\mathscr{H}^i_{ua1}$	First term of E^i_{UA}
$\mathscr{H}^i_{ua2}$	Second term of E^i_{UA}
α_{uarm}	Constant to control the importance of $\mathscr{H}^i_{ua1}$ to $\mathscr{H}^i_{ua2}$
N_f	Frame number of 3D foot trajectory
H	Homography matrix
θ_f	The 3D line represent the walking orientation of a foot
P_{min}	The top left point of the 3D foot trajectory when foot is on the ground
P_{max}	The top right point of the 3D foot trajectory when foot is on the ground

$L_{traject}$	The line that connects P_{min} and P_{max}
$L^{2d}_{traject}$	The projection of $L_{traject}$ into image space
$L^{\mathbf{m}}_{stride}$	The stride length of a sub-map **m**
L_{stride}	The final average stride length
T_i	Time that we turn off the light ($i = 0, 2, 4, etc.$)
T_j	Time that we turn on the light ($i = 1, 3, 5, etc.$)
N_{switch}	Number of turn off-turn on the light
Δ	Time delay
c	Camera
r	Reference
$\overline{\varepsilon}$	The minimized mean error of multiple cameras synchronization with respect to Δ
T_{id}	Constant waiting time between two consecutive asymmetry calculation steps
R_a	A curve measured at one of the right joints (knee, elbow, ankle, etc.)
L_a	A curve measured at one of the left joints (knee, elbow, ankle, etc.)
δ	Asymmetry index
$F_{\rho_{ua}}$	Mapping function that maps one value of ρ_{ua} into degrees.
p_{vfoot}	The 2D projection of $P_{shoulder}$ into $L^{2d}_{traject}$
P_{vfoot}	The corresponding 3D point of p_{vfoot}
$H_{shoulder}$	The distance from shoulder to the ground according to [50]
$P^{init}_{shoulder}$	The initial position of 3D shoulder deduced from P_{vfoot}
pt_s	A particle which is an candidate of 3D shoulder
pt_e	A particle which is an candidate of 3D elbow
E_S	Cost function to evaluate an shoulder candidate
N_p	Number of particles
N_l	Number of frames that are used to reconstruct the 3D arm
DP	A $(4 \times N_l)$ table used in finding the most consistent movement of arm
E_{eh}	Cost function value of a cell in DP
$P^{3d}_{shoulder}$	The 3D shoulder joint location of our 3D arm model
P^{3d}_{elbow}	The 3D elbow joint location of our 3D arm model
P^{3d}_{hand}	The 3D hand joint location of our 3D arm model

(X^w, Y^w, Z^w)	The x,y,z coordinates of a 3D point in world space
(X^c, Y^c, Z^c)	The x,y,z coordinates of a 3D point in camera space
(x, y)	The x,y coordinates of a 2D point in image space
f	Focal length
(p_x, p_y)	Horizontal and vertical effective pixel size
(o_x, o_y)	Image centre coordinates

CHAPTER 1

INTRODUCTION

Nowadays, in order to diagnose abnormal gait patterns in medical clinics, gait analysis systems play a crucial role. Among state-of-the-art systems, motion capture (MOCAP) is without a doubt the most popular one due to its very high precision. Each patient is asked to wear several infrared (IR) reflective markers so that multiple IR cameras can observe the signal. Such systems cost thousands of dollards and require complex knowledge from the operator to make it function properly. In recent years, along with the development of computer hardware, video-based human motion capture and analysis has achieved huge advances. Along with the appearance of RGB-D cameras on the market, high-accuracy motion capture can be archived in real time based on this new technology. Most of these systems aim for recognizing human gesture only (based on the upper body), so their application in gait analysis, which mostly focuses on lower body, has not been fully studied and tested yet. In this thesis, we focus on developing a new low-cost video-based gait analysis system that can automatically reconstruct a 2.5D human skeleton over time and then analyze the gait to detect related pathological problems. Combining multiple vision and optimization techniques to create such a system that can outperform vision-based state-of-the-art system in gait analysis, that is the principal contribution of this thesis.

We introduce such a system that is composed of a treadmill associated with two low-cost color camcorders (see Fig 1.1). Our system automatically reconstructs the human skeleton model (see Fig 3.2) of the patient walking on the treadmill from a recorded video sequence. Gait asymmetry problems and other gait disorders are then detected by observing the angle variations between various joints of the skeleton through time. Our ultimate purpose when designing this system is to obtain higher accuracy of skeleton reconstruction than the current low-cost state-of-the-art (Kinect) for efficient gait analysis by adding some necessary but easy-to-realize constraints to the estimating process. The processing on each camera is designed to work independently to ease the system setup in

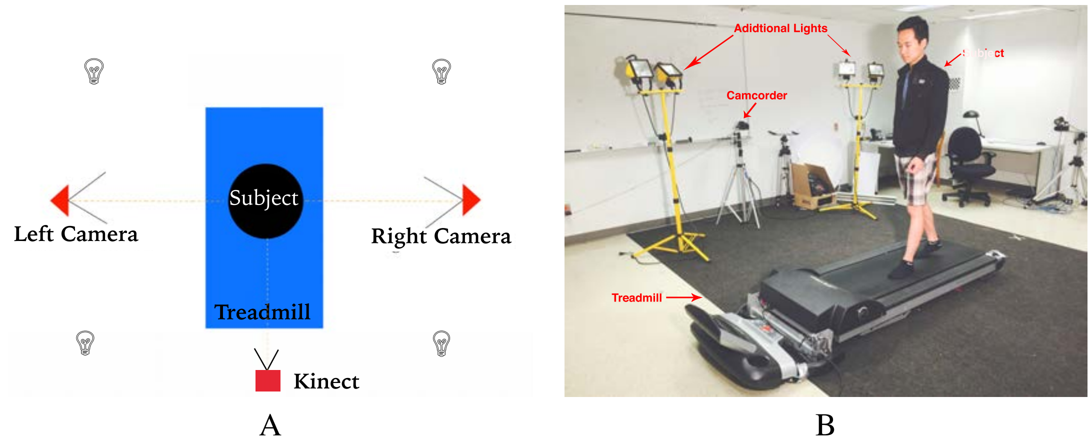


Figure 1.1: (A) Our system consists of one treadmill and two camcorders on the left and right side of the treadmill. Four light sources were placed at the four corners of the room to assure good light diffusion. (B) Our system in real life.

clinics. Furthermore, although the two cameras are not synchronized when constructing skeleton models, the frame correspondance between them can be obtained to give more advanced information to experts.

For validation, we compare the results performed by our method with those from ground truth data obtained with markers manually placed on the subject's body. A comparison with the 3D skeleton reconstructed with the Kinect v.1 [1] is also presented. Finally, we present an application for symmetry/asymmetric gait classification. Our results on different subjects show that although our methodology is simple and low-cost, it is very efficient and provides sufficiently accurate measures for gait assessment.

Our thesis is divided as follows: in chapter 2 we make a literature review about state-of-the-art related techniques. We present our methodology in detail in chapter 3, chapter 4 is for validation and experimentation. Chapter 5 introduces our application on assessing the symmetry/asymmetry of gait. Finally, we conclude this research as well as discuss some future work.

1. The Kinect v.2 was not available at the time we wrote this thesis. The Kinect v.2 has better resolution and better depth map, it can therefore produce better 3D skeleton results compared to Kinect v.1.

CHAPTER 2

LITERATURE REVIEW

Gait is the way in which we move our whole body from one point to another. Most often, this is done by walking, although we may also run, jump, hop etc. Gait analysis is a method used to assess the way we walk or run to highlight biomechanics abnormalities. This study encompasses quantification, i.e. acquiring some helpful gait parameters, as well as interpretation, i.e. drawing some conclusions about individuals from their gaits. Some real life applications of it would be evaluating the recovery from a surgery, or detecting the potential risks that can lead to injury when running or walking, or helping athletes to boost their performance, etc. In gait analysis, we often consider the movement of lower limbs and pelvis within a gait cycle, which is measured from initial heel strike to the ipsilateral (same side) heel strike. All events that happen inside a gait cycle are summarized in Fig. 2.1.

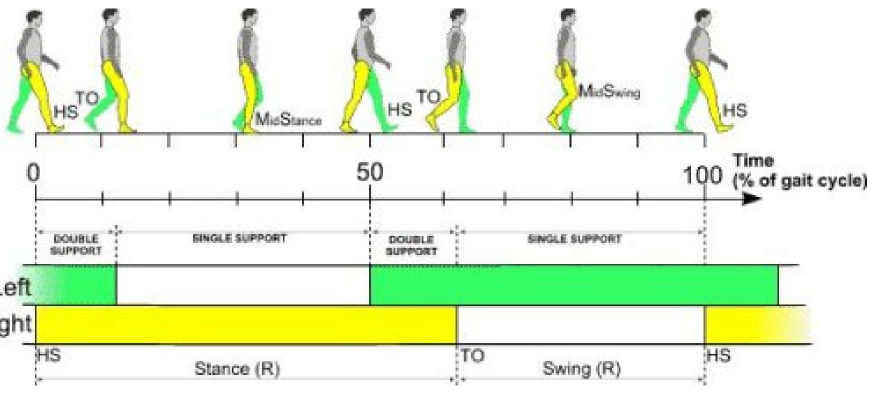


Figure 2.1: A detailed gait cycle description [65].

As we can see in Fig. 2.1, there are two periods in a gait cycle: stance and swing which takes 40% and 60% of total time respectively. The "double support" period indicates the state in which the two feet are on the ground, while "single support" means there is only one foot on the ground. A deep understanding of gait will help us extract more helpful information leading to more important constraints for our algorithm. For a specialist, when assessing gait, observing the patient's walk from all points of view is necessary. Normally, two body planes are considered: sagittal plane and coronal plane (Fig. 2.2). Each plane gives different information and should be used in combination with each other to provide more precise analysis result.

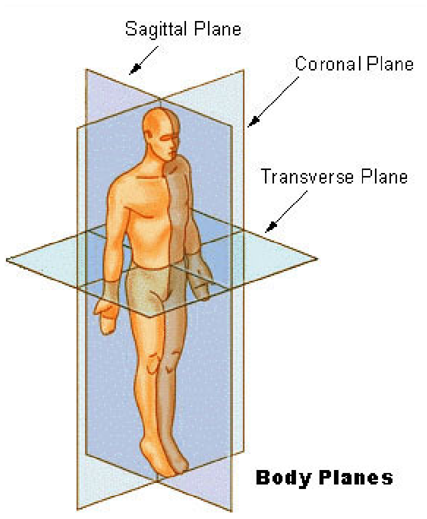


Figure 2.2: Anatomical planes in a human [66].

Current methods for precise measurement of human movement usually involve complex systems requiring a dedicated laboratory and trained specialists. These systems can be subdivided into two main classes: marker-based and marker-less systems. Marker-based systems are still the most favoured methods of human motion assessment and gait analysis. Essentially, markers (acoustic, inertial, LED, magnetic or reflective) are placed on the body and tracked to measure the human motion. Some examples of such systems are the popular motion capture system Vicon [1], see Fig. 2.3, based on infrared (IR) reflective markers put on human body parts of interest; or a very recent one GAIMS [28], in which four special laser sensors (BEA LZR-i100) are placed around the walking trajectory to locate the feet over time. However, these systems require specific equipments and expertise, and are cumbersome and difficult to use. Despite these drawbacks these systems are accurate and mature, and remain the gold standard for human motion

1. http://www.vicon.com/

assessment and gait analysis, and are readily available in the market.

Marker-less systems constitute an interesting alternative but are still in development within research groups. Over the last decade, many computer-vision-based approaches have been made to focus on marker-less systems and the results have been very promising compared to traditional marker-based systems. Our research falls into the domain of marker-less human pose estimation, which is a process of estimating the configuration of the underlying skeletal articulation structure of a person.

In this section, we focus mainly on the review of pose estimation techniques after 2006, readers interested in older surveys before 2006 are referred to [3]. In pose estimation, there are different levels of difficulties that need to be tackled depending on one's particular problem: single vs. multiple cameras vs. depth camera; 2D pose reconstruction vs. full 3D pose reconstruction. Each level has its own advantages as well as its limitations. For easier understanding, we follow the functional taxonomy in [3] in which they separate all algorithms in this area into two categories based on the use of a prior human model. We will at the same time compare these techniques with ours as well as indicate how different our problem is compared to theirs.

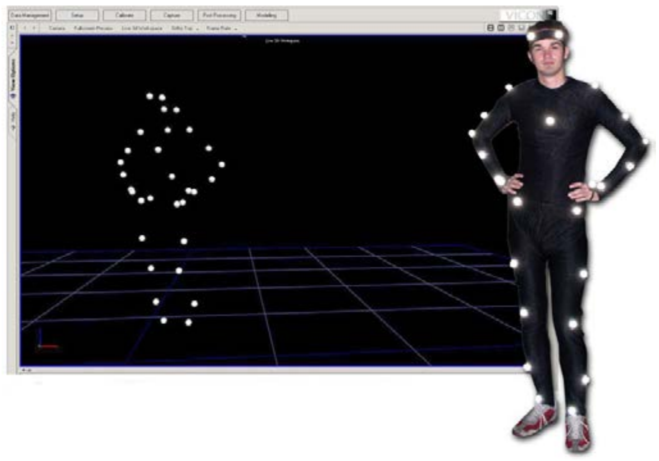


Figure 2.3: The Vicon system makes uses of infrared reflective markers on the subject body parts [67].

2.1 Model free

This category refers to algorithms that directly detect 2D pose from individual images, and often is called the bottom-up method. Because the natural huge variation of human poses, the common point of these techniques is that they all require a large of samples of dataset that can cover this large range of poses. In this category, *Probabilistic assemblies of parts* is the group of techniques dealing with 2D and 3D pose detection problem, while *Example-based methods* is dedicated for 3D pose recovery problem only.

2.1.1 Probabilistic assemblies of parts

Since 2000, 2D object detection in monocular images has achieved huge advances. The techniques in this area focus on detecting likely location of body parts and then assembling them to form the final configuration. More recently, many state-of-the-art techniques [4–8, 10, 68] are based on the Pictorial Structure (PS) model, which was proposed by Felzenszwalb [9]. In PS, objects are represented as a flexible configuration of N distinct parts. In a Bayesian framework [10, 11], the posterior probability of the 2D part configuration L given the single image I and the set of object model's parameters Θ is computed as

$$p(L|I,\Theta) \propto p(I|L,\Theta)p(L|\Theta)$$

This model allows one to learn parameters Θ from training data. The prior term $p(L|\Theta)$ has a tree structure and represents the kinematic dependencies such as relative angle, part-joint locations, and between body parts, see Fig. 2.4.

The likelihood term $p(I|L,\Theta)$ is assumed as a product of individual part likelihoods. This approximation is good if there is no overlapping between parts, but such cases happen often in real life. A good likelihood is critical to obtain a good PS performance. Many attempts to learn the likelihood functions use the normalized score of a classifier trained on rectangular boxes containing parts. Some features used are shape context [4], histogram of orientated gradient (HOG) [12], or raw image pixels [8]. Learning meth-

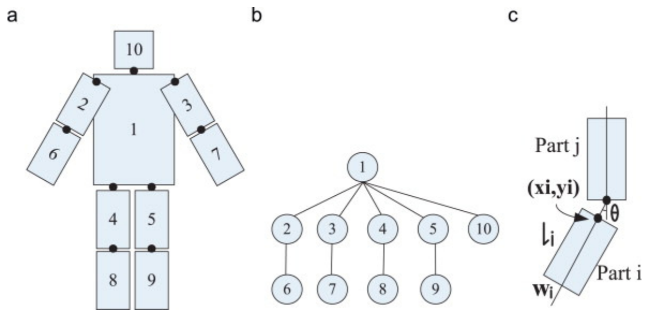


Figure 2.4: The tree structured representation of human body parts used in [9].

ods are typically Boosted classifiers or SVMs but current likelihood still does not attain a good body part detection accuracy. Recent trends focus on learning a more complex likelihood but in addition require more computational cost. Sapp *et al.* [7] use information extracted from shape similarity defined by regions and contours. In [14], they cluster the space of possible poses to learn the discriminative appearance models between different poses. Wang *et al.* [13] argue that the human anatomy representation of "parts" is not necessary and that a combination of multiple limbs in a hierarchical representation based on Poselet [15] works even better. More recently Silvia *et al.* [16, 17] define a new Deformable Structure (DS) model that can capture the pose-dependent body shape and enhance the accuracy on moving limbs by incorporating optical flow to extend the image evidence from adjacent frames, as well as to predict the next pose. One important contribution of this approach is that it provides an efficient 2D pose detection in cluttered natural scenes from a single view. Some impressive results of these techniques allow detecting multiple pedestrians and recovering their 2D [19] or 3D poses [18].

2.1.2 Example-based methods

In this approach, a mapping from image features, such as silhouette, to 3D poses is learnt directly from the database of samples [30, 31]. Due to the very large number of samples, many efforts attempt to reduce the search space by using sophisticated techniques such as SVM, mixture of experts, random forests, etc. Okada *et al.* [31] address the problem of 3D pose estimation in a cluttered scene. Starting from the image, they

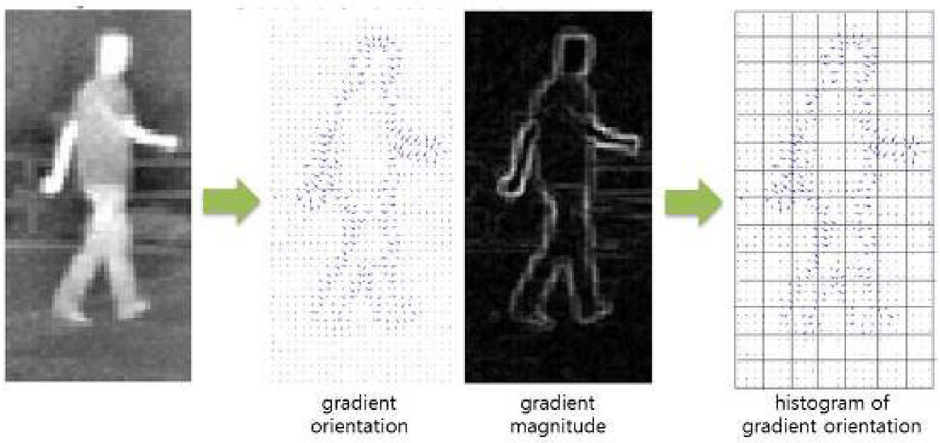


Figure 2.5: The feature histogram of orientated gradient (HOG) is widely used in the human detection area [12].

first calculate a feature vector of gradient orientation histograms, then search for which pose cluster the current pose belongs to using an SVM classifier. Finally, the 3D pose recovery is performed by multiple local regressor of the selected cluster. By grouping poses into clusters, the search space for 3D pose estimation is greatly reduced leading to a less expensive computational cost . The same approach can be found in [32] where they use randomized forest for pose classification. By defining classes on a torus, this technique is insensitive to view points. Furthermore, the hierarchical searching mechanism of random forests in the training set allows one to reject unsuitable candidates at the early stage and thus is less computational demanding. One last advantage of this method is the capability of exploring multiple branches which greatly increases its robustness. Despite the very powerful mechanism for directly estimating 3D pose, the example-based methods require typically large training sets and are limited to the poses or motions used in training. Adding too many types of movements may cause more ambiguities in the mapping.

Some of these techniques were designed to work in very cluttered environments, which is not necessary in our problem in which a controlled environment is available, such as a hospital or a medical clinic, etc. With an efficient background subtraction algorithm, we can easily isolate the person's silhouette so there is no need for a sophisticated body part detector which is trained on a very large dataset. We can however make use of their appearance model to guide our skeleton reconstruction process. Furthermore, as indicated in [30], there is always uncertainty in estimated location of each body part which makes them an inappropriate choice for an accuracy-above-all system like gait analysis system.

Another more recent direction of research has dealt with color-depth cameras (or RGB-D cameras). In a depth image each pixel indicates depth in the scene, rather than only a measure of intensity or color. This depth information can be obtained with different sensors: stereo camera, time-of-flight camera or structured light camera (e.g. Microsoft Kinect). However, to develop a low cost, reliable and easy-to-install system, the Kinect sensor is currently the best solution to obtain depth images. Moreover, the human pose method proposed by Shotton *et al.* [21] represents the state of the art for color-depth cameras and is implemented in the Microsoft Kinect system. It computes the subject's pose using machine learning techniques (Random Forest) with local 3D shape descriptors. The accuracy of the Kinect calculations seems reasonable for gait analysis and was successfully used in [20–22] for instance. The aim of these techniques is to achieve high accuracy 3D pose reconstruction while still assuring the real time performance on the gaming machine Xbox360. The use of Random Forest (RF) is efficient, simple and available for parallelization. Making use of depth information from the Kinect, Shotton *et al.* [21] propose using features based on the difference between just a few depth image pixels. They also introduce a very large dataset containing a million synthetic pairs of images of people of varied shapes in varied poses in which each pair includes a depth image and its corresponding body part label image. The RF is trained using the features above on this dataset by employing the standard entropy minimization. The reconstruction pipeline is shown in the Fig. 2.6.

Each point is assigned to a body part label by traversing the RF. Clustering is then performed to obtain hypotheses about the locations of joints of the body. The last step is to fit the skeleton model using kinematic and temporal constraints. One of the limitations of this technique is that it fails to recognize joints whose body parts are invisible to the camera. An offset regression approach is proposed in [22] in which each pixel votes for

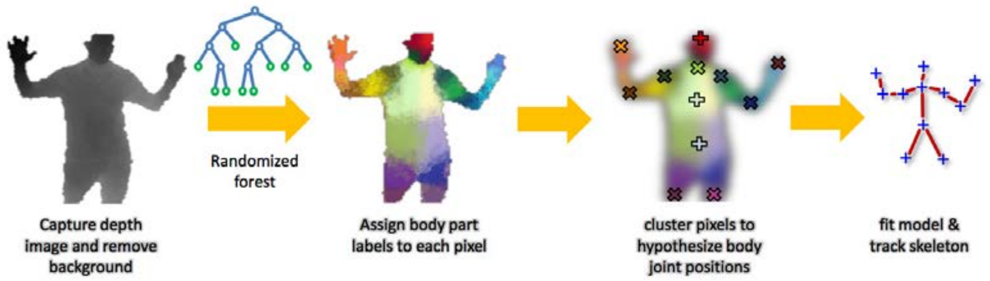

Randomized forest
Capture depth image and remove background
Assign body part labels to each pixel
cluster pixels to hypothesize body joint positions
fit model & track skeleton

Figure 2.6: The fundamental steps of the skeletal detection technique was introduced in [21].

the positions of the different body joints. A random regression forest (RRF) is therefore introduced to realize this. Compared to the original RF, the RRF only differs at the leaf nodes, which capture the distribution over the relative 3D offset to each body joint of interest. This new representation can collect votes from each pixel to predict joints of the body whether they are visible or not. This new improvement significantly outperforms the recognition result in [21]. They continue to improve this model by incorporating a latent variable that encompasses some useful global characteristics of the image, such as the human height, the torso's direction, etc. In [43], we also have experimented with using Kinect camera to construct the basic human skeleton model starting from the depth image, then the angle information at knee is tracked over time to distinguish the normal from the abnormal gait. More recently, Dantone *et al.* [29] successfully combined the PS with the Random Forest, which once was used with great success in [32], producing very promising results. But in the other hand, the Kinect suffers problems due to hardware limitations: very limited field of view (about 57.8°), limited range (about from $0.2m$ to $4m$), low video resolution (640×480), noisy depth image, limited precision (e.g. 40mm at 2m from the sensor) causing not very consistent results, especially on the lower limbs; and finally it captures only 30 fps (frame-per-second) for the video, which is a limitation when capturing fast swing movement walking, running, etc. or gait in general. Furthermore, a disadvantage of the front view provided by the Kinect is the appearance of body part occlusion during a gait cycle that might lead to incorrect pose estimation results. In our system, any low cost camcorder could work in theory since we do not require any special requirements on image quality. By de-interlacing each frame, we obtain 60 fps video allowing 60 human poses per second, this makes our gait assessment more exact. Beside, the Kinect camera itself can not record video and requires a Window-based machine connected and some necessary softwares to operate. In our system [51, 52], a cheap camcorder is more flexible with options to record videos into a memory card or its own hard drive. The processing then takes place offline after the video acquisition. All these Kinect-based approaches are for generic purposes, so they also have limited accuracy requirement compared to specialized gait analysis system.

2.2 Direct model use

In this section, we consider the methods that use an explicit model of a person's kinematics, shape, and appearance to reconstruct human pose. This kind of approach is also referred as top down. The main reason to choose this approach is that it still works in difficult cases such as partial occlusion, lost silhouette cues due to fast movement, etc. According to [3], recent trends focus on using stochastic sampling techniques based on sequential Monte Carlo, as well as the use of constraints on the model. Gall *et al.* [23] summarizes that the strategies for model-based pose estimation can be categorized into global optimization, filtering / smoothing / prediction and local optimization.

Stochastic global optimization, or interacting simulated annealing (ISA) [24], has shown its ability to recover from errors and provides high accuracy estimation. On the other hand, it produces jitter due to its lack of temporal coherence, which is typical, when tracking over time. Beside, the fact that the computational cost greatly increases as the accuracy increases limits its application in practice.

In filtering approaches, we want to estimate an unknown true state x_t of the model from some noisy observations y_t, such as images. This problem is typically solved by Kalman filtering or particle filtering

$$x_{t+1} = f_t(x_t) + v_t,$$

$$y_t = h_t(x_t) + w_t$$

where noise parameters v_t and w_t are known. One of its advantages is that this approach takes temporal coherence into account but its limitation resides on being inaccurate for motion analysis in high dimensional spaces. To deal with that, some heuristics are introduced to combine local optimization with filtering, such as covariance scale sampling [33], smart particle filtering [34], and annealed particle filter [35]. The results have been proved to work well but the lack of evidence that they converge to the optimal solution makes them unreliable.

Local optimization, such as Iterative Closest Point (ICP), shows the highest accuracy if the initial state is close enough to the global optimum. The word "local" implicitly indicates that it only searches for local optimum and it needs an initialization. The tracking process is also unable to recover if the object of interest is lost. Those techniques can be used in both multi-view or monocular view approaches. In our methodology, we apply the local optimization method for 2D torso, 2D upper arm, 3D arm since their initial locations are available via specific context, kinematic knowledge and other image processing techniques.

2.2.1 3D pose estimation from multiple views

To make use of the advantages as well as to limit the drawbacks of these sampling techniques, Juergen *et al.* [23] have created a multi-layer framework in which the first layer produces an initial pose that will be refined by the second layer. Particularly, for the first layer, the ISA estimates a relatively accurate initial pose by using image features such as silhouettes, colour and some weak prior on physical constraints from all views. In the second layer, filtering helps reduce jitter produced from the first layer and the local optimization step increases the accuracy of the final estimation. The comparison between ISA and the original techniques above has been made in their paper. In Fig. 2.7, PF stands for the original particle filter, APF stands for the annealed particle filter, PFICP stands for the combination between particle filter and ICP.

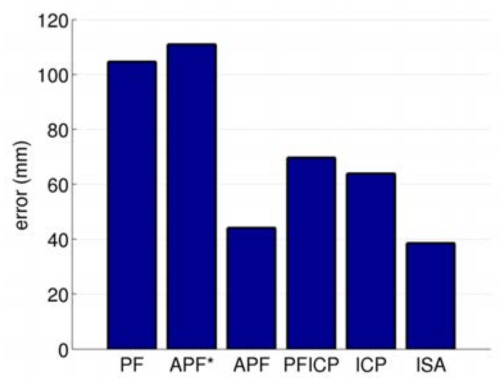


Figure 2.7: Result shows the superior performance of ISA against other techniques [24].

2.2.2 2D/3D pose estimation from monocular view

Compared to the problem of 3D pose estimation from multiple views, the monocular view problem is much more challenging due to its inherent ambiguity. For that reason, many attempts to overcome this by adding more constraints on the skeleton configuration and kinematics. In [25], they introduce a technique that represents the human body as a probabilistic graphical model such as part based model [4–8, 10]. They argue that the original model using a soft connection at the joint between two connected parts, referred as Loose Limb Model (LLM), cannot reflect all the characteristics of the training set. They hence introduce a novel representation called Fixed Joint Model (FJM), in which the connected joint between parts is required to coincide. This model is defined over joints instead of over parts/limbs. The distribution of each joint, which is learnt from the training set, will be used to draw samples in the sampling process. This global sampling process is hierarchical which means each set of child samples is drawn conditioned on those samples generated for the parent node. With this FJM representation, they show that Expectation Maximization (EM) maximizes the posterior probability $p(L|I,\Theta)$ faster, see Fig. 2.8 for a detail comparison.

More recently, another approach [30] retrieves 3D poses from 2D parts locations estimated from state-of-the-art detectors, with probabilistic assemblies of parts (Section 2.1.1). The problem with this approach is that the result from the 2D detector has an associated uncertainty, and that this is an ill-posed problem. To deal with this, they first

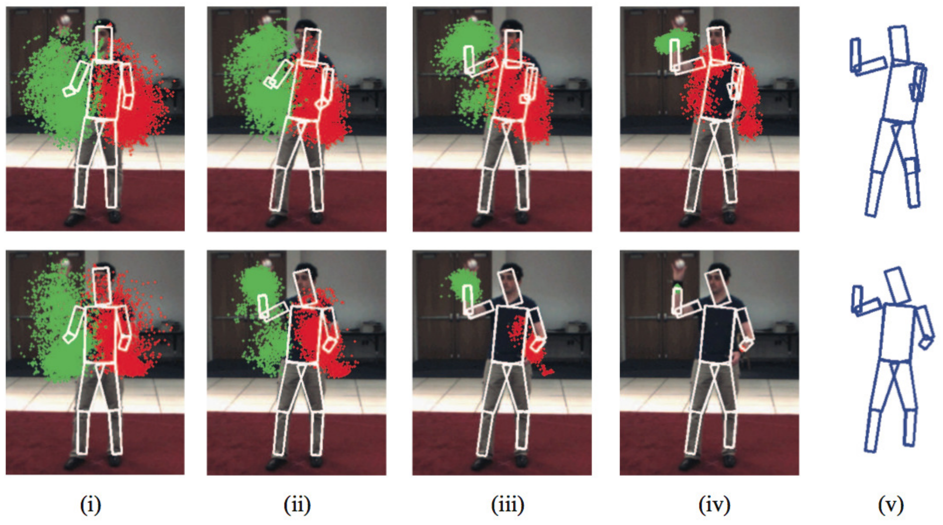


Figure 2.8: The first row shows the result of LLM and the second row shows the result of FJM. (i) to (iv) show results obtained at iteration 1, 3, 5, 10 respectively. (v) shows the final 3D pose. The convergence speed of FJM is clearly superior than LLM (source [25]).

represent the output of 2D detector by a Gaussian distribution and use this as image cues to sample the solution space to obtain a set of initial ambiguous poses. Due to the uncertainty of 2D body parts, they introduce a new strategy of sampling by successively drawing batches. Each batch is sampled from a multivariate Gaussian whose mean and covariance matrix are iteratively updated using the Covariance Matrix Adaptation [36]. Each pose is represented by using a coordinate-free kinematic representation, based on the Euclidean distance matrix. A one-class classifier SVM, which was trained on the training set, helps to determine the most anthropomorphic pose among them. This algorithm is illustrated in Fig. 2.9.

The closest work to ours is from Courtney *et al.* [1], in which they also estimate half body structure from side view. They first describe the human pose as an ellipse-based hierarchical tree structure, then isolate the two lower limbs by performing active contour fitting on the xt side of the 3D space and time block xyt. The 3D block is created by grouping images whose dimensions are (x,y) over time t. With this method, they can efficiently distinguish the left leg from the right leg but can not model the foot,

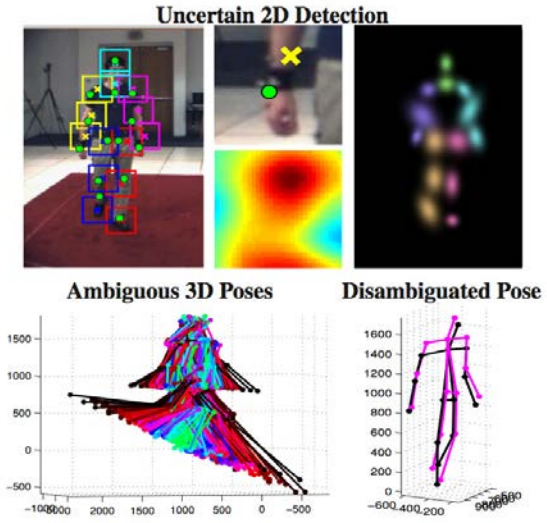
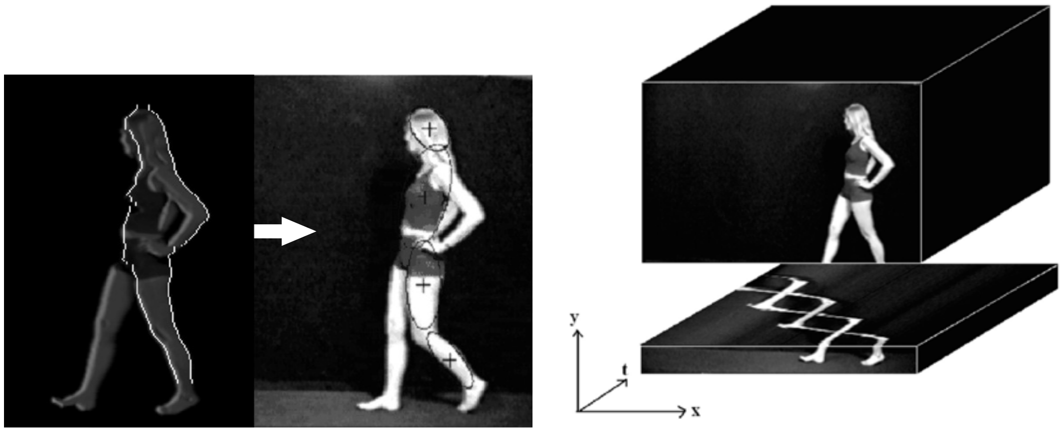


Figure 2.9: The 3D poses disambiguation from uncertainty 2D poses is introduced in [30].

see Fig. 2.10(B). The heart of their system is based on the observation that each limb can be represented by an ellipse, see Fig. 2.10(A). Their estimation accuracy was very sensitive to limb shape changes. Furthermore, they only focused on lower limbs and thus neglected the upper limbs that are also important factors for gait analysis. More important, this system and other one-camera-based side-view systems suffer from one limitation: they can only watch one side at a time, making it impossible to compute the symmetry of a gait.

Generally speaking, in model-based approaches, too many ambiguities in 2D pose inference as well as too many degrees of freedom of the skeleton model are main reasons that make this area more difficult. With sampling technique, the more accuracy we want to obtain, the more computational cost it requires. As the estimation accuracy is paramount, we add some constraints about colour of the foot and make use of skin colour cues to separately detect foot and hand, thus greatly reducing our search space. With this bottom up approach, we obtain the best of both worlds that achieve both accuracy and performance.



A B

Figure 2.10: (A) In [1], the gait is obtained by first finding body contours (left image), then performing ellipse estimation for each body part within these contours. (B). The *xyt* block is used to distinguish left leg from right leg.

2.2.3 Learnt motion models

Another trend in human tracking from multi-views is the use of prior poses or motion pattern learnt from a motion database, which has achieved impressive tracking results in difficult and ambiguous scenarios [37]. In [38, 39], Gaussian process dynamical modes have been introduced for embedding motion in a low-dimensional latent space. These approaches are however limited to specific motion models with relative small variation in motion and fixed transitions. This kind of technique is often used for recognizing different types of actions but rarely for gait analysis. The principal difficulty is that reconstructing such a reliable, well generalized gait database (for symmetry and asymmetry gaits) is costly. This is the main reason that we decided not to depend on a predefined database when designing our methodology.

2.2.4 3D pose estimation from depth sensor

Many real-time human tracking systems introduced recently have obtained promising results [40, 41] with depth sensors. A combination between an accurate generative model with a discriminative model gives data driven evidence about body part locations.

For the generative model, the human model is a collection of 15 rigid body parts which are spatially constrained according to a tree-shaped kinematic chain. At each iteration, they perform a local model-based search. When tracking is lost due to fast movement or occlusion, trained patch classifiers will be used to detect body parts which then allow one to reinitialize the local model-based search. Siddiqui *et al.* [41] focus on tracking human pose from the frontal view. It uses a top down approach that generates samples to find the optimal pose by using the data-driven Markov chain Monte Carlo (MCMC) approach that compares synthesized depth images to the observed depth image. A bottom up detector for head, arm and forearm is used to reduce the search space and thus speed up the search convergence. A Markov chain dynamics process is introduced to combine the top down and bottom up processes. Beside, many old techniques working on 2D images or video sequence can be applied to depth data, such as mixture of experts [45], structure prediction model [46], latent variable model [47]. In general, these techniques proved their robustness in tracking with high accuracy but comparing to those in [20–22], they still suffer from tracking failure and not being able to quickly re-initialize.

Our approach when designing our algorithm firstly focuses on developing techniques that are data-free, which means that there is no need for training anything. By considering a gait system with limited range of movement of people, our method resolves the problem of tracking failure. We will present our methodology in detail in the next chapter.

CHAPTER 3

METHODOLOGY

3.1 System overview

For the best gait assessment, accuracy information of the pose and a sufficiently long time of evaluation are crucial. With that ultimate aim, we propose to use a treadmill, which allows a long and stable walk, and two monocular cameras placed on the left and right sides of the treadmill. Another advantage of letting the subject walk on a treadmill is to keep him relatively stationary with respect to the camera point of view and thus help us to almost be immune to ambiguity issues which are a problem for systems in which the subject walks in a corridor back and forth. Each camera in our system is responsible for capturing the movement of its corresponding half body over time. There is no special hardware requirements for the camera so a camcorder or even a low-cost webcam can work in our system as long as the frame rate remains constant. This is one of our advantages over those using the Kinect which is more limited as an autonomous camera. In essence, we focus on estimating 2.5D pose information (two profile views but each view is 2D only) which will be used to calculate the left and right gait patterns. We are interested in criteria during a cycle of gait such as: the stride length and the angle variation of left body parts (elbow, knee and foot) vs. their right counterparts. To perform calculation on these characteristics, we found that placing the camera on the side of the subject (not in front like the Kinect) has advantages that we have a clear, occlusion-free view of each half body, and sufficient information to perform a thorough analysis. Furthermore, we argue that, if calculating the criteria above is the ultimate aim, there are little differences between a 2D pose and a 3D pose from the side point of view of a gait because most of the walking dynamics take place in the sagittal plane. This observation leads us to only reconstruct 2D human poses and thus much simplifies the process. In the gait analysis domain, the correct location of leg and foot is of top priority. Courtney *et al.* [1] dealt with this problem by creating a 3D block *xyt* to mainly distinguish left leg from right leg, but were not able to model the foot. In our system, we propose a very simple way to quickly and efficiently model the foot, but yet applicable in realistic situations. That is each foot will wear a sock with a different color. This simple constraint guarantees stable and high accuracy detection rates of each foot. Another constraint we require from the subject is that he has to wear a short-sleeve T-shirt to expose the skin color as much as possible.

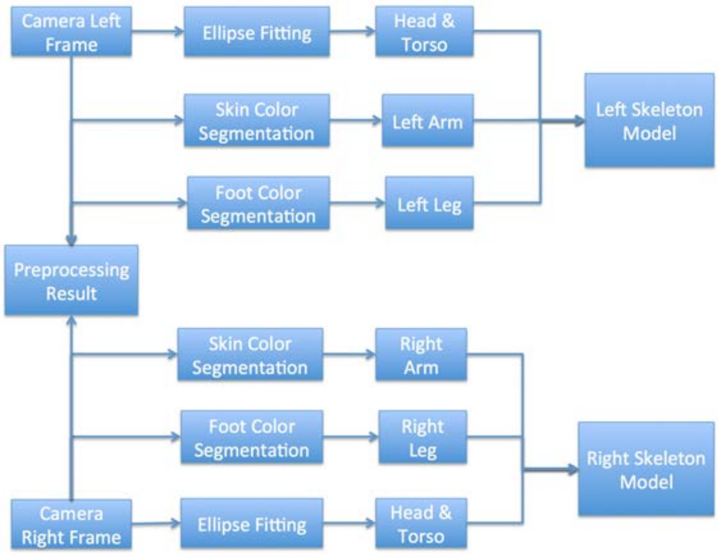


Figure 3.1: Overview of our system: each processing of each camera is treated independently to get its corresponding skeleton model. This architecture encourages parallel computing to speed up the system performance.

As shown in Fig. 3.1, our system starts with a pre-processing step, then we estimate the head and torso joints using an ellipse fitting technique, detect lower and upper arms using a particle filter based on a skin color model that was already predefined in the pre-processing step above, and finally construct leg based on the known foot color information. The whole process is repeated exactly for both left and right cameras resulting in left and right skeleton models (see Fig. 3.2 for more details) respectively. We also propose a method to efficiently synchronize the frame between left and right cameras, and thus allow more advanced gait assessment such as asymmetrical (left-right) walk patterns, etc. For evaluation of our methodology, we aim to compare the gait characteristics (including the angles at interesting joints and the stride lengths) calculated from our 2.5D skeleton models and those from the full 3D skeleton using the Kinect SDK (v1.8 for Kinect v.1 [42]), which we consider the marker-less state-of-the-art until now.

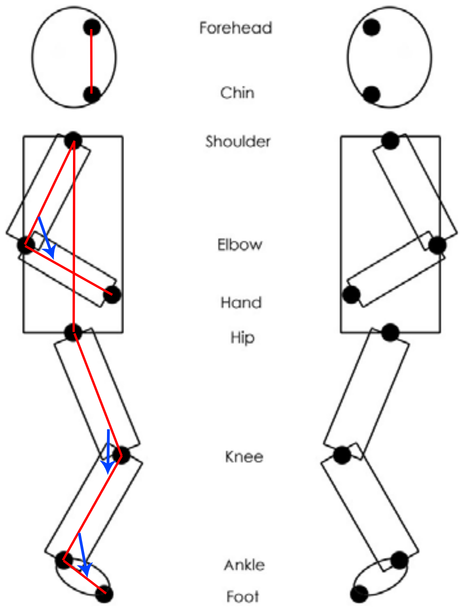


Figure 3.2: The left and right human skeleton models are used in our system. Each model consists of 9 nodes and 8 joints allowing more complicated gait analysis. Red lines connect body joints together. Blue arrows represent our 3 joint angles of interest at elbow, knee and ankle.

3.2 Preprocessing

Firstly, we need to calculate color models in each camera separately for the foot and the skin. These dynamic color models allow the system working under various lighting environments in different clinics, as well give more freedom to the operator when choosing the sock colors. The sock color should be homogeneous and distinguishable from the background color. For modelling skin, the authors [26] introduced a simple but efficient way by first locating the face then extracting skin color within. Furthermore, in order to be more robust to noise as well as blur due to motion, we apply the edge enhancing technique, introduced by Papari *et al.* [27], to make the color more homogeneous and efficiently remove blur at the same time. Another advantage of [27] in our case is that we can completely erase red markers (used for validation in Chapter 4) on the subject's body to make sure that they do not have any effect on our methodology afterward. Although applying the filter proposed in [27] adds more processing time to our system, the aim of our system is not real-time performance but the best possible accuracy in reconstructing the human skeleton model leading to highest quality gait assessment.

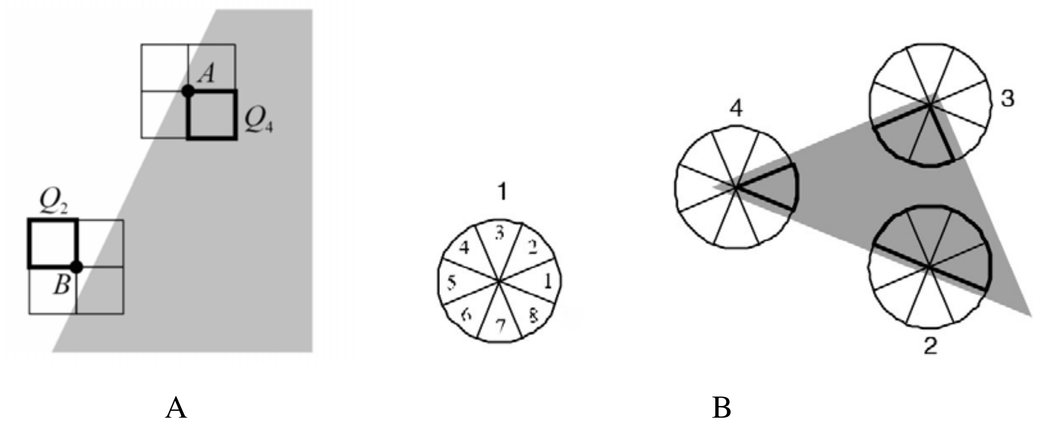


Figure 3.3: (A) The square block: the output of points *A* and *B* are determined by Q_4 and Q_2 respectively. The straight line edge is not preserved.(B) The circular block is divided into 8 sectors in [27]. The sectors (with thick-line) determine the final output.

A brief description of this algorithm follows in [27], they aim to smooth texture details while the sharpness at edges is significantly enhanced at the same time to obtain a painting-like effect on the image. Their contribution lies on the non-linear smoothing operator that empirically preserves better the information at edges and corners compared to approaches such as Kuwahara filter [71]. This method is actually an improvement of Kuwahara filter when resolving Kuwahara filter's problem, i.e. block structure and Gibb phenomenon [1] on the output. The reason of this defect is that, in Kuwahara filter, the final output is determined based on the four sub-square components of a square block, see Fig. 3.3(A). Only the one with smallest standard deviation contributes to the final

1. Gibb phenomenon is the particular manner in which the Fourier series of a piecewise continuously differentiable periodic function behaves at a jump discontinuity (source: Wikipedia)

result, which is clearly not suitable for preserving all edge shapes. In [27], they consider a circular block and divide it into N_s sectors and labeled each sector s_i with a different weight w_i, , see Fig. 3.3(B). All sectors contribute to the final output through a linear combination, but only the ones with significantly large weights play main roles.

This approach was also proved by the authors to provide high precision and efficiency in preserving different types of edges and corners. As illustrated in Fig. 3.4, the blocking effect due to the Gibb phenomenon is eliminated, the edges are well enhanced and the color in homogenous areas are quite identical.

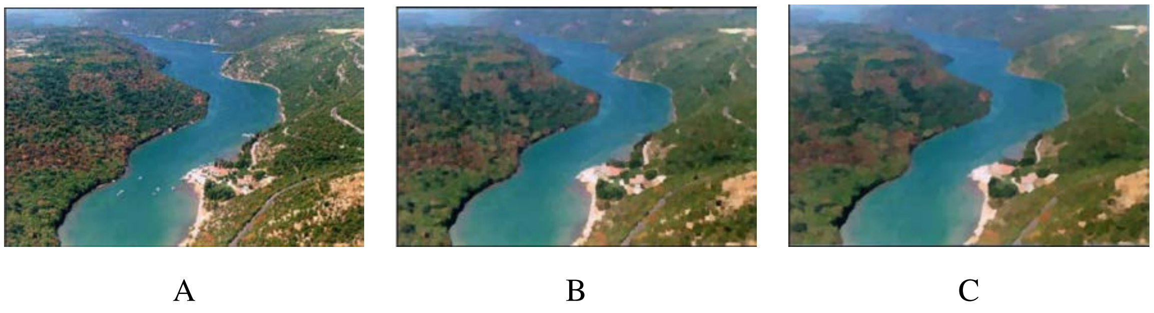


Figure 3.4: (A) The original image. (B) Output of Kuwahara filter with visible blocking effects. (C) Output of [27].

In our system, the first 360 frames (6 seconds) of walk are dedicated to constructing the color models only. After doing background subtraction and calculating the bounding box BB_1 of the person's silhouette, we extract the head and foot areas from the silhouette based on anthropometric ratios of human height [50]. Due to the natural shape of heads, we apply ellipse fitting to locate the head, then the frontal part of the ellipse is used for calculating skin color. We then cluster the color information within the half lower part of that region to reduce the appearance of the hair pixels using the k-means algorithm [2]. We set $k = 2$ since we only consider 2 types of color: skin and hair color. Other minor details such as eyes, mouth, etc. are usually removed after applying [27]. The majority cluster, which represents the skin information, will be used to calculate the skin gaussian model $(\mu_{skin}, \sigma_{skin})$. This skin model is kept updated every 360 frames (6 seconds).

2. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster (source: Wikipedia).

For the foot color modelling, the process is different as we have to deal with leg occlusions. During a cycle of walk, leg occlusion happens when the distance between the two feet is very small. Taking that observation into account, we keep track of the variation of the bounding box BB_2 of the half lower part of the silhouette. Every time the width dimension of BB_2 falls below a threshold, which is fixed empirically as $1/3$ of BB_1 height in our implementation, we apply the k-means clustering with $k = 3$, which corresponds to left foot color, right foot color and pants color. The target foot color is obviously in the majority group. In a similar manner to skin modelling, the foot model is calculated over 360 frames.

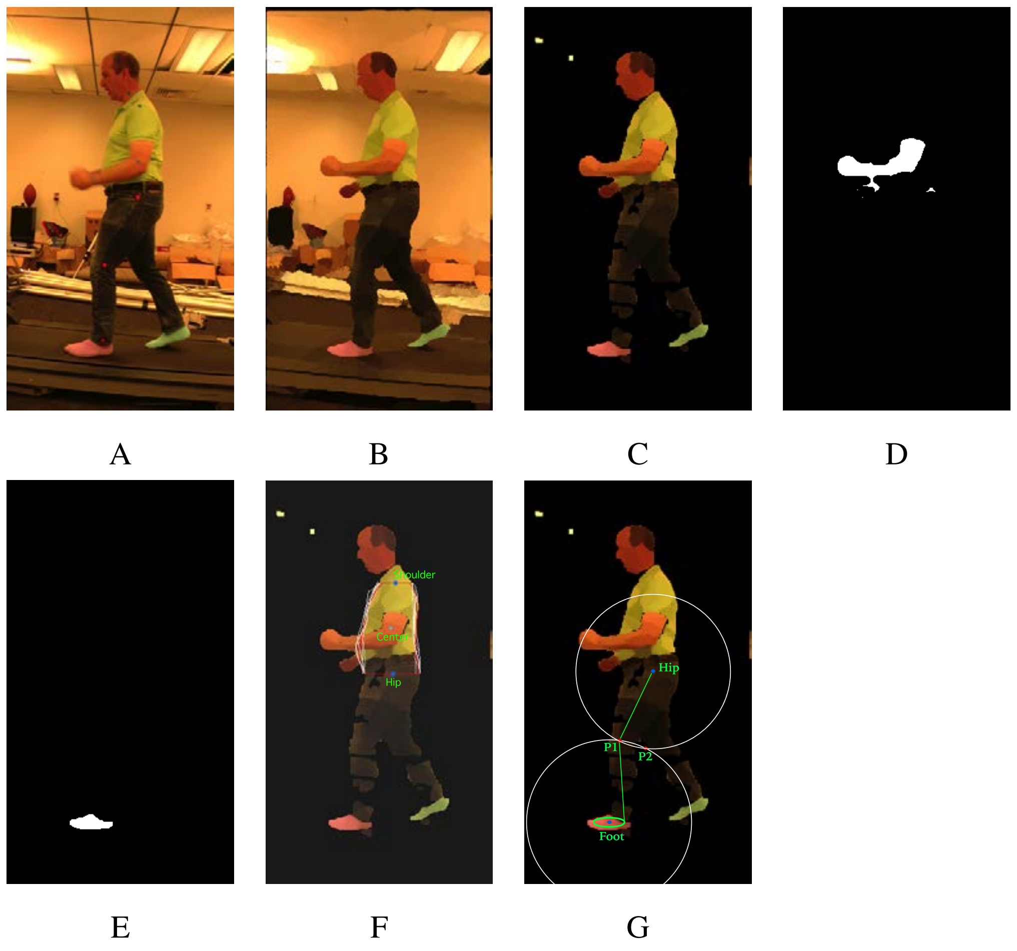


Figure 3.5: (A) The subject walks on a treadmill with red markers placed on his body. (B) After applying [27], the edge information is enhanced, the color becomes more homogeneous and most noise is removed. (C) Background subtraction. (D) The skin extraction after removing face skin exposes the arm skin. (E) Foot color extraction. (F) The averaged torso shape is represented by a red curve, the white curves indicate the torso shape detected in each frame. The three points ("Center", "Shoulder", "Hip") are used to move and rotate the shape. (G) The knee position can be efficiently computed by intersecting two circles at $P1$ and $P2$.

Lastly, since the torso detection (shoulder and hip) plays an essential role in order to have successful arm and leg estimations, we also model the form of the subject's main body in a way that minimizes the effect of his clothes on the final result. To facilitate processing as well as to increase robustness of our system, the bounding boxes are aligned along the horizontal axis, with respect to the first frame, to make sure that the subject stays at the same location in all frames. The torso shape model is defined as <L_c,R_c> in which $L_c = \{x_{li}, y_{li}\}_{N_c}$ and $R_c = \{x_{ri}, y_{ri}\}_{N_c}$ are its left and right contours respectively, N_c is the number of points of both L_c and R_c since they always have the same scanline within the torso height ratio. Algorithm 1 demonstrates how to construct the $L_c^{\mathbf{k}}$ and $R_c^{\mathbf{k}}$ at frame **k**, note that we choose the top-left corner of the image as the origin coordinate.

Algorithm 1: Constructing the left and right contours of the torso model.

```
   Input: Subject silhouette after background subtraction at frame k
   Output: the contour model at frame k
1  Fill any black holes inside the silhouette
2  Define torso range $[y_{shoulder}, y_{hip}]$ based on the torso height ratio
3  $y \leftarrow y_{shoulder}$
4  while $y > y_{hip}$ do
5      Compute the center $C$ of the current torso slice
6      $(x_{li}^{\mathbf{k}}, y_{li}^{\mathbf{k}}) \leftarrow$ first background point on the left side of $C$
7      $(x_{ri}^{\mathbf{k}}, y_{ri}^{\mathbf{k}}) \leftarrow$ first background point on the right side of $C$
8      Add $(x_{li}^{\mathbf{k}}, y_{li}^{\mathbf{k}})$ to $L_c^{\mathbf{k}}$
9      Add $(x_{ri}^{\mathbf{k}}, y_{ri}^{\mathbf{k}})$ to $R_c^{\mathbf{k}}$
10     $y \leftarrow y - 1$
11 return $< L_c^{\mathbf{k}}, R_c^{\mathbf{k}} >$
```

To cope with the movement of clothes during a gait and arm occlusion, the final contour is calculated by averaging all contours $< L_c^{\mathbf{k}}, R_c^{\mathbf{k}} >$ over the training frames, see Fig. 3.5(F). Given that the *y*-component is fixed, the *x*-component of the point at the *i*-th position of the final contour of L_c is computed as follows

$$x_{li}^{final} = \frac{\sum_{\mathbf{k}}^{360}(x_{li}^{\mathbf{k}} * \Psi_{li}^{\mathbf{k}})}{360 - N_{skin}^{li}} \tag{3.1}$$

where $x_{li}^{\mathbf{k}}$ represents the value of *x*-coordinate of the *i*-th point of L_c at frame **k**, $\Psi_{li}^{\mathbf{k}}$ is the binary function that gives 1 if the image point (x_i, y_i) at frame **k** is not skin point, and gives 0 otherwise. N_{skin}^{li} is the number of frames where the point (x_{li}, y_{li}) contains skin color. The final x-component of each element in R_c are deduced by using the same formula. By the very similar manner, the x-component of the point at the i-th position of the final contour of R_c is computed as follows:

$$x_{ri}^{final} = \frac{\sum_{\mathbf{k}}^{360}(x_{ri}^{\mathbf{k}} * \Psi_{ri}^{\mathbf{k}})}{360 - N_{skin}^{ri}} \tag{3.2}$$

In essence, we want to minimize the effect of the swinging arm to the torso shape so whenever the arm intersects with either the left or right contour, the points within the intersection zone are not taken into account, as shown in the formula above. The torso model construction is computed over 360 frames. More details on how we use this model to accurately locate the position of shoulder and hip are explained in Section 3.3.1.

3.3 Pose estimation

In this section, we only explain our methodology for reconstructing the left half skeleton since the same process is executed on the right side. Firstly, the bounding box of the full person silhouette is extracted from the background subtraction result $I_{silhouette}$, the relative height ratio between body parts is applied to divide this bounding box into sub-part bounding boxes obtained from anthropometry data [50]. We then estimate the orientation of arm, leg, head and the torso region. Finally, starting from the torso, the exact locations of head, arm and leg are inferred accordingly. This process ensures that we always obtain the model with stable relative position between body parts, one wrong estimated position of one node does not break the whole model. For example, in case of a bad estimated position of the shoulder, it only affects the angle between torso and upper arm, but not the angle between upper arm and lower arm.

3.3.1 Head and torso

An ellipse-based fitting technique [69], which is described below, is used for locating the head. The orientation of the main axis of the ellipse is the orientation of the head joint in our skeleton model. With known head length, we can easily locate the joints $P_{forehead}$ and P_{chin} along the main axis.

In order to perform the ellipse fitting efficiently on a set of scattered points, we use the well-known technique introduced in [69] which is fast and non-iterative, see Fig.3.6. In [69], a general conic can be represented by an implicit second order polynomial

$$f(\mathbf{a},\mathbf{x}) = a_0x^2 + a_1xy + a_2y^2 + a_3x + a_4y + a_5 = \mathbf{a}\cdot\mathbf{x} \tag{3.3}$$

where $\mathbf{a} = (\begin{array}{cccccc} a_0 & a_1 & a_2 & a_3 & a_4 & a_5\end{array})^T$ and $\mathbf{x} = (\begin{array}{cccccc} x^2 & xy & y^2 & x & y & 1\end{array})^T$.

In the definition above, $f(\mathbf{a},\mathbf{x}) = 0$ is the equation of the conic, and $f(\mathbf{a},\mathbf{x_i})$ is the difference from point $p_i = (x_i, y_i)$ to the curve. The ellipse fitting can be stated as the minimization of the sum of algebraic squared distances

$$d(\mathbf{a}) = \sum_{i=1}^{N} f^2(\mathbf{a},\mathbf{x_i}) \tag{3.4}$$

The N input points are gathered in the form of a *design matrix* $\mathbf{D} = [\begin{array}{cccc} \mathbf{x_1} & \mathbf{x_2} & \cdots & \mathbf{x_N}\end{array}]^T$. The objective function $d(\mathbf{a})$ is then re-written as

$$d(\mathbf{a}) = \mathbf{a}^T\mathbf{S}\mathbf{a} \tag{3.5}$$

where $\mathbf{S}$ is the 6×6 *scatter matrix*. They make use of the equality constraint $4a_0a_2 - a_1^2 = 1$, or, in matrix form

$$\mathbf{a}^T\mathbf{C}\mathbf{a} = 1 \tag{3.6}$$

where $\mathbf{C}$ is the *constraint matrix*. By introducing the Lagrange multiplier [3] λ and differentiating, they have to solve an eigen system subject to the constraint 3.6 as

$$\mathbf{S}\mathbf{a} = \lambda\mathbf{C}\mathbf{a} \tag{3.7}$$

Since $\mathbf{S}$ is positive definite, there is a unique solution for the best fit ellipse. After finding the ellipse parameters, then positions, orientation, etc., are finally deduced. More details and mathematical proofs can be found in [69].

3. In mathematical optimization, the method of Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to equality constraints.

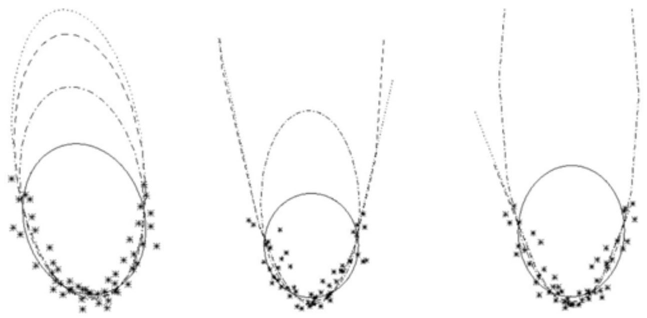


Figure 3.6: The ellipse fitting result from [69] (solid line) compared to other approaches. (Source [69]).

In [1], Courtney *et al.* introduced a torso estimation technique based on ellipse fitting [70] which is an improvement of [69], but we find that it was too sensitive to the overall torso shape which normally includes two arms in motion. The swing of the arms within a gait cycle causes deformation of the silhouette when deciding the torso orientation. In [51], we introduced a method that eliminates the arm section from the torso shape and thus minimizes the effect of swinging arms for the torso orientation estimation. Although this method performs well overall and better than results from Courtney *et al.*, it still suffers from the fact that the ellipse can not represents all torso shapes of humans. In some cases, the subject's belly also causes torso shape distortion. Besides, clothes are empirically another factor that leads to failure estimation due to shape changes in our experimentation. In other cases, the ellipse-shape assumption also fails on persons with bent torso, etc. We hence propose a new method that dynamically captures the subject's torso shape ($<L_c, R_c>$ in Section 3.2) and then use it to locate the shoulder and hip with higher precision. The cost function E_t used to evaluate the new torso shape $< L_p, R_p >$ is defined as follows

$$E_t = \sum_{i=1}^{N_c} \Phi(L_p^i, cl^i) + \sum_{j=1}^{N_c} \Phi(R_p^j, cr^j) \tag{3.8}$$

where the first term represents the error accumulation between every point i of the left contour (L_p^i) and its closest edge point cl^i that has the same y-value (on the same horizontal scanline), the second term represents the error accumulation between every point j of the right contour (R_p^j) and its closest edge point cr^j that has the same y-value (on the same horizontal scanline); and the function Φ is defined as

$$\Phi(p,c) = \begin{cases} ||p_x - c_x|| & \text{if } ||p_x - c_x|| < \tau \\ 2\tau & \text{otherwise} \end{cases} \tag{3.9}$$

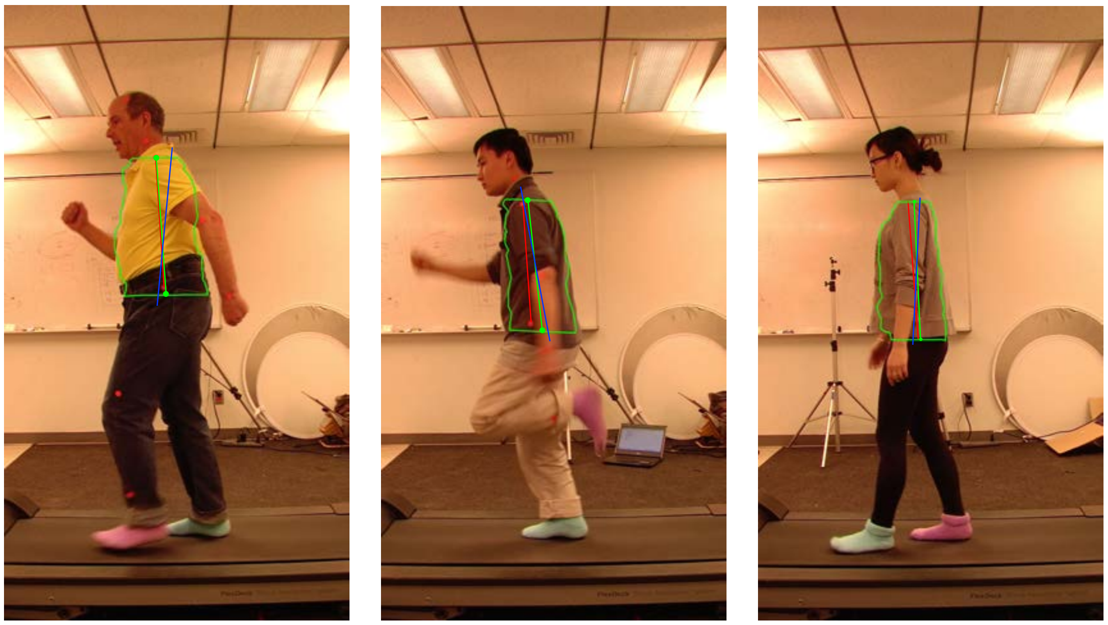
where τ is a parameter to control the accuracy of the estimation. Here, a too small τ can lead to the overfit problem and thus probably could not find the matching. In our system, this value is empirically set as $\tau = 9$ pixels (at resolution 540×960). With the help of a treadmill, the subject is relatively stationary to the camera view. For that reason, the search range for new torso shapes is much reduced and it thus increases the performance a lot. For validation, we also test and compare our technique with [51] (which is the most recent contribution, as the best of our knowledge, on torso estimation) using different subjects' body shapes, and different types of clothes. The torso estimation in [51] has, as expected, larger error range compared to the ground-truth than ours, while their computing time is just a little bit faster. Another advantage of our method is that it is not sensitive to the skin detection error like [51]. If many skin points on the arm are missed, they stays always in the list after the arm removal step and become the input for the ellipse fitting that might lead to a false estimation. In our method, this case is trivial since we already remove noise when constructing the torso shape. If many skin points are mis-considered as the torso border points in a frame, a good chance that they will not be there in the next frames due to the arm swinging. Finally, when we take the average (in practice, we choose the median instead of the average to be more robust to noise), all the mis-classified points are completely eliminated. Then in the recognition step, the skin information does not impair the process. The algorithm that matches the predefined torso shape model to current frame is described in Algorithm 2.

Fig. 3.7 presents our torso estimation results in three different cases with small differences compared to the ground-truth. With the help of the dynamically-predefined torso shape model in the preprocessing step, the results also show that our method has capacity to detect well the torso boundary in case of being partly occluded by the arm

Algorithm 2: Locate shoulder and hip using torso shape matching.

```
   Input: Torso shape model $<L_c, R_c>$ from preprocessing, current frame
   Output: Shoulder and hip positions
 1 $C \leftarrow$ centroid of $<L_c, R_c>$
 2 $CS \leftarrow$ generate new possible positions for $C$ following a 2D Gaussian distribution
   with center $C$ and standard deviation $\sigma = \frac{1}{10}$ torso height
 3 $Arr_\alpha \leftarrow$ angle variations from $[-10°, 10°]$
 4 for each position $C_p$ in $CS$ do
 5     for each angle $\alpha$ in $Arr_\alpha$ do
 6         $(R, T) \leftarrow$ forms the affine transformation matrix with respect to the
           Euclidean distance (between $C_p$ and $C$) and $\alpha$
 7         $<L_p, R_p> \leftarrow$ applies the transform matrix $(R, T)$ to $<L_c, R_c>$
 8         $E_t \leftarrow$ computes cost function using Equation 3.8
 9 $< L_n, R_n > \leftarrow$ the contours having the minimum $E_t^*$
10 $P_{shoulder} \leftarrow$ the mid-point between $L_n^{first}$ and $R_n^{first}$
11 $P_{hip} \leftarrow$ the mid-point between $L_n^{last}$ and $R_n^{last}$
12 Update $< L_c, R_c > \leftarrow < L_n, R_n >$
13 return $P_{shoulder}, P_{hip}$
```

(see Fig. 3.7A). In Fig. 3.7, our result (green line) compared to ground-truth (red line) showed significant less error than [51] (blue line). The numerical error quantization is also presented in Table 4.II. Once the shoulder and hip are well estimated, we continue to reconstruct the arm in the next section, and the leg afterward.



A B C

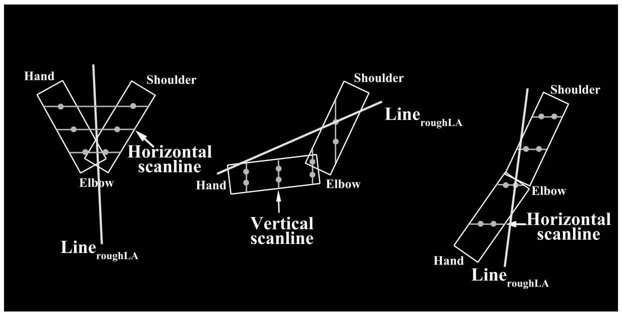
Figure 3.7: Torso estimation comparison between [51] (blue line), ours (green line) and ground-truth (red line). Our method is very close to the ground-truth in all three cases, especially better than [51] in (A) where the subject's body shape does not have ellipse-form, and (C) where the clothes distort the body shape.

3.3.2 Lower arm

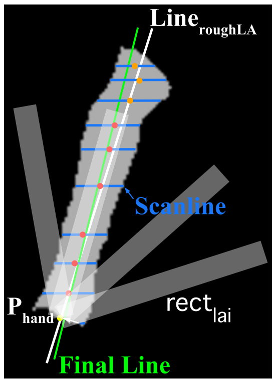
With the skin color model built during the preprocessing step (Section 3.2), we filter each frame using this model to produce the skin map Map_{skin} (a binary image in which the value of skin pixels is 1, and the rest is 0), see Fig. 3.5(D). Combining this with the body part height ratio provided by anthropometry data [50], we can eliminate easily the head and neck areas from the skin map, thus leaving alone the skin of the left arm and probably a small portion of the right arm if it is not occluded by the torso. To deal with the noise and this unwanted right arm, we only preserve the skin pixels within the bounding box having the closest distance to that from previous frames. With the short-sleeve T-shirt constraint, as mentioned in Section 3.2, the lower arm will be fully exposed while only a small part (or none) of the upper arm is visible. This condition gives us enough information to correctly estimate these two body parts.

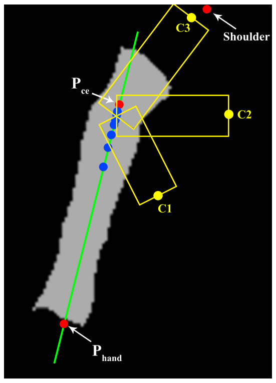
In order to correctly estimate the orientation of the upper arm, due to its different orientation with the lower arm, we need to separate the lower arm from the upper arm by removing the pixels belonging to upper arm in the skin map when performing line fitting. Although the number of upper arm points compared to lower arm points is small (or zero in case of no visible upper arm), the risk of still having some effects on the lower arm estimation is still high for our gait analysis system. To simplify the skin map of the arm without losing directional information, we apply a scanline (line of pixels) technique [43] which has already proved its efficiency in preserving body parts orientation over a 2D/3D cloud of points.

Fig. 3.8(A,B) presents the overview of this process. In summary, we firstly estimate a rough direction of the lower arm using linear least squares on all arm skin pixels to obtain a line $Line_{roughLA}$, this line is for comparison purpose only. Based on the size of the bounding box of skin points, we generate a list of vertical (from left to right) or horizontal (from up to down) scanlines in a way that has a maximum number of scanlines. We then cluster each scanline using k-means with $k = 2$ (i.e., two center points for each scanline). The Euclidian distance between these two points is used to distinguish the lower arm from the upper arm within a scanline: if the distance is large enough, that means each cluster centre belongs to one part of the arm (we keep two points); otherwise, they all belong to one part (we only keep the mid-point of these). By sampling this way, we always have two extreme points: one represents the hand and the other one represents the top point of the upper arm. Generally, the extreme point which is farther to the shoulder implies the hand position. In some special poses of gait, in which the hand is very close to the shoulder (first case in Fig. 3.8(A), not usual for a normal walk), kinematic constraints are needed to choose the correct one. The number of scanlines generated is a trade-off between accuracy and computational time. The more scanlines we have, the more exact the linear fitting is, but also the more computational work for the CPU. In practice, we set empirically a distance of 4 pixels (at resolution 540×960) between two consecutive scanlines for best performance.



A





B C

Figure 3.8: (A) Scanline type decision (horizontal or vertical) depends on $Line_{roughLA}$ orientation and hand location. (B) Lower arm estimation using linear least squares on scanline clustering. The white line represents the rough line $Line_{roughLA}$, red dots are scanline centres that belong to the lower arm, while the orange dots do not. The final green line is estimated by performing linear least squares on the red dot set. (C) Sampling technique for estimating the upper arm: blue points follow a Gaussian distribution whose centre is located at the known elbow. Many yellow rectangles, such as C_1, C_2, C_3 representing three candidates of upper arm, were generated.

Next, the task is to obtain the points that belong to lower arm. Due to the rise of foreshortening [4] problem when swinging the arm, we can not rely solely on the lower arm joint length L_{LA} to remove the upper arm from the set. As shown in Fig. 3.8(B), taking the hand joint P_h as the center, we generate a list of rectangles $List_{rectla}$, in which each $rect_{lai}$ has size (L_{LA}, w_{lar}) where w_{lar} controls how close the points fall within the rectangle, by

4. Foreshortening is the visual effect or optical illusion that causes an object or distance to appear shorter than it actually is because it is angled toward the viewer (source: Wikipedia).

following a circle distribution with the angle from 0° to 360° . The best rectangle, that is the one containing most of cluster central points, is calculated producing red points in Fig. 3.8(B). The drawback of this approach is that it could not work correctly if upper arm and lower arm have a similar orientation under foreshortening issue (see Section B.3 for more discussion). By using linear least squares on these red points, we can estimate the line $Line_{LA}$ representing the lower arm direction (green line in Fig. 3.8(B)). The index of the extreme scanline belonging to the hand is then located. Finally, we locate the hand joint P_h which is the projection of the hand scanline on $Line_{LA}$. The detail algorithm can be found in Algorithm 3.

Algorithm 3: Lower arm orientation estimation.

```
   Input: Skin map of the arm $Map_{skin}$
   Output: Hand position $P_{hand}$ and the orientation of the lower arm $Line_{LA}$
 1 $Line_{roughLA} \leftarrow$ compute line fitting on $Map_{skin}$
 2 $Rect_{bb} \leftarrow$ compute the bounding box on $Map_{skin}$
 3 $(width_b, height_b) \leftarrow$ calculate size of the bounding box $Rect_{bb}$
 4 if $width_b > height_b$ then
 5     $List_{scan} \leftarrow$ sample vertical scanlines along $x$ axis
 6 else
 7     $List_{scan} \leftarrow$ sample horizontal scanlines along $y$ axis
 8 $List_{cps} \leftarrow$ apply k-means ($k = 2$) clustering on each scanline of $List_{scan}$
 9 $P_{hand} \leftarrow$ locate two extreme points of $List_{cps}$, then choose the hand joint
10 $List_{LA} \leftarrow \varnothing$
11 $List_{rectla} \leftarrow$ generate rectangles of size $(L_{LA}, w_{lar})$ with center at $P_{hand}$
12 $rect_{best} \leftarrow \varnothing$
13 for $rect_{lai}$ in $List_{rectla}$ do
14     if Number of center points in $rect_{lai}$ > Number of center points in $rect_{best}$ then
15         $rect_{best} \leftarrow rect_{lai}$
16 $List_{LA} \leftarrow$ points within $rect_{best}$
17 if $List_{LA} \neq \varnothing$ then
18     $Line_{LA} \leftarrow$ compute line fitting on $List_{LA}$
19 return <$P_{hand}, Line_{LA}$>
```

3.3.3 Upper arm

After estimating the lower arm orientation $Line_{LA}$ and locating the hand, this section aims at locating the last missing joint of the arm: the elbow. Due to foreshortening issues, the accumulated information until now is not sufficient to allow us to directly calculate the exact location of the elbow, but is enough for evaluating where an elbow joint should be with the following contraints

1. Reasonable distance to the hand joint P_{hand}
2. Must lie on the lower arm line $Line_{LA}$
3. Reasonable distance to the shoulder joint $P_{shoulder}$
4. Must be a skin point of upper arm (orange points in Fig. 3.8(B))

In such a case, a filtering technique, such as particle filter, is the best fit technique to solve our problem. The first two constraints can be used for limiting the range of newly generated samples, while the last two give us clues to calculate the cost function.

Starting from P_{hand}, we limit the search range by locating the farthest possible position of elbow P_{ce} along the line with the known length of lower arm L_{LA}. However, the real location of the elbow can be closer due to foreshortening. Therefore, from P_{ce} to P_{hand}, we sample candidate points S_{elbow} (blue points in Fig. 3.8(C)) on L_{LA} following a 1D Gaussian distribution whose centre is located at P_{ce} and σ is the forearm segment length. For each candidate S^{i}_{elbow}, using the walking direction knowledge, we generate a new set of candidates $S_{shoulder}$ for shoulder following a semi-circle-distribution (yellow rectangles in Fig. 3.8(C)). Each shoulder candidate $S^{j}_{shoulder}$ has a distance

$$dist(S^{j}_{shoulder}, S^{i}_{elbow}) = L_{UA} * \rho_{ua} \tag{3.10}$$

where L_{UA} is the upper arm length and ρ_{ua} is a random scale parameter (which is empirically set from 0.7 to 1 in our experimentation) to deal with foreshortening of the upper arm. We will discuss ρ_{ua} more in the next chapter. In summary, we combine S_{elbow} and $S_{shoulder}$ into a final list of candidates $S_{candidates}$, each candidate is a pair $(S^{i}_{elbow}, S^{i}_{shoulder})$, where i is the index of the candidate in the list. The next step is to define a cost function on which each candidate will be evaluated.

In order to verify the skin coverage of each candidate, as many other Pictorial Structure approaches, we use a rectangle-form to represent the upper arm. We therefore create a rectangle from S^i_{elbow} to $S^j_{shoulder}$ and calculate the skin pixels that fall inside this rectangle. The cost function E^i_{UA} for evaluating candidates i of $S_{candidates}$ consists of two terms which are defined below:

$$E^i_{UA} = \alpha_{uarm} * \mathscr{H}^i_{ua1} + (1 - \alpha_{uarm}) * \mathscr{H}^i_{ua2} \tag{3.11}$$

in which α_{uarm} represents the importance of one term to the other, the first term $\mathscr{H}^i_1$ controls how close $S^j_{shoulder}$ is to $P_{shoulder}$ estimated in Section 3.3.1. Note that each term is normalized to the range $[0,1]$.

The first term is defined as follows

$$\mathscr{H}^i_{ua1} = \min(\frac{dist(S^i_{shoulder}, P_{shoulder})}{M}, 1) \tag{3.12}$$

where $dist(S^i_{shoulder}, P_{shoulder})$ is the distance between a candidate and $P_{shoulder}$. In our implementation, we set M_{ua} equal to the head height because this value is small enough to discard bad candidates while it is still large enough to cover the maximum movement range of upper arm during walk.

The second term $\mathscr{H}^i_{ua2}$ controls how much skin pixels are represented by the rectangle. Taking our short-sleeve T-shirt constraint into account, the skin pixels of the upper arm only cover a portion at the bottom of the rectangle (or not at all), the ratio between this portion and the whole rectangle is indicated by a threshold r_{shirt}, which is a value between $[0,1]$. This parameter is calculated approximately based on the ratios of how much the skin map covers the upper arm area. In our experimentation, we also tested our system in cases such that no upper skin arm were exposed with success. Although the upper arm orientation error is a little bit larger than that in normal cases, the shoulder constraint always keeps the process from estimating bad results. $\mathscr{H}^i_{ua2}$ is described as follows:

$$\mathscr{H}^{i}_{ua2} = r_{shirt} - \frac{\sum LH - \sum UH}{\sum \text{total pixels}} \tag{3.13}$$

where LH are skin pixels in lower half, UH are skin pixels in upper half. After many experimentations on different subjects, we empirically found that setting $\alpha_{uarm} = 0.35$ gives the best results. The candidate giving the lowest score of the cost function will contain the positions of the elbow P_{elbow} and shoulder $P_{shoulder}$ that we are looking for. Fig. 3.8 demonstrates that our method gives relative accurate value of the angle between upper arm and lower arm and is capable of dealing with the foreshortening problem. In Algorithm 4, we present the main steps of our process.

Algorithm 4: Main steps of the elbow estimation process by Monte-Carlo sampling technique.

```
Input: Hand position $P_{hand}$, shoulder $P_{shoulder}$ and lower arm orientation $Line_{LA}$
Output: Elbow position $P_{elbow}$
1  $S_{candidates} \leftarrow \varnothing$
2  $S_{elbow} \leftarrow$ initialize set of elbow candidates
3  for $S^{i}_{elbow}$ in $S_{elbow}$ do
4      $S_{shoulder} \leftarrow$ initialize set of shoulder candidates
5      for $S^{j}_{shoulder}$ in $S_{shoulder}$ do
6          Add $< S^{i}_{elbow}, S^{j}_{shoulder} >$ to $S_{candidates}$
7  for candidate $l$ in $S_{candidates}$ do
8      $E^{l}_{UA} \leftarrow$ calculate cost function using Equation 3.11
9  $P_{elbow} \leftarrow$ elbow position having minimum cost function $E^{*}_{UA}$
10 return $P_{elbow}$
```

In term of performance, thanks to the known lower arm orientation (see Section 3.3.2), we can reduce the search space for elbow (using only a 1D Gaussian distribution), thus greatly reducing the final number of candidates in $S_{candidates}$. We ran our system on a machine with configuration: CPU Intel core i5 (2.6GHz), 4GB RAM, and using Matlab environment. The upper arm processing was the longest computational time step in the whole process, it took 1.8-2.4s per frame (640×480). Since Matlab is slow in running heavy loop computing, we believe that we can get faster results if using other programming languages (such as C++).

3.3.4 Leg reconstruction

The issue of reconstructing the lower part is easier since we have these advantages:

1. Negligible foreshortening of the legs due to the nature of gait on a treadmill.
2. Known left foot color obtained after a preprocessing step, see Section 3.2.

Filtering the frame by the left foot color model gives us a binary image, we then apply the ellipse fitting on it to acquire the orientation of the left foot, which is represented by the small green ellipse in Fig. 3.5(G). Combined with the known foot length by anthropometry data [50], the position of the ankle P_{ankle} is located at the extreme left point of the main axis of the estimated ellipse. In Section 3.3.1, we also located P_{hip}. The only point missing is the location of the knee. With the known length of both upper leg and lower legs, the first advantage allows us to directly locate the knee at the intersection between two circles whose centres are at P_{hip} and P_{ankle} respectively. This intersection results in two possible candidates during a stance phase, and only one during the swing phase, see Fig. 3.5(G). In the former case, we choose the intersection point on the right based on kinematic knowledge. In practice, the latter case never happens so we only take the mid-point between P_{hip} and P_{ankle} if there is no intersection.

Besides from being simple, another advantage of this method is that it does not require any prior information of the lower-half body parts, such as pants color, leg shape, etc. as long as body part ratios can be applied. This condition allows the system to work with patients with artificial lower limb which is usually hard or sometimes impossible for other shape-based methods.

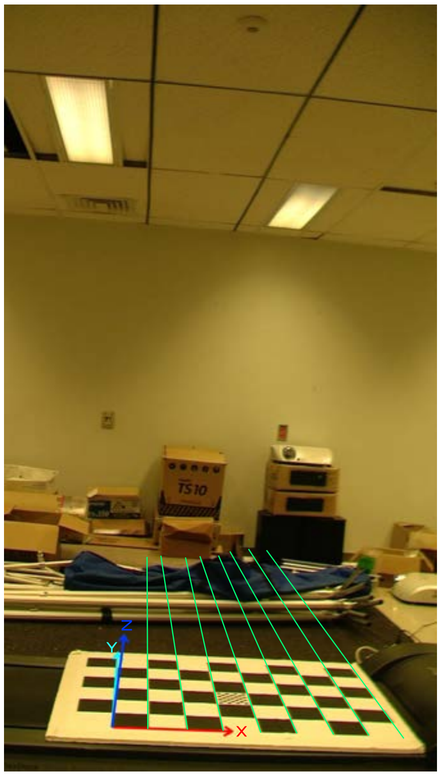
After successfully reconstructing the left skeleton model, the right one is computed in the same way.

3.3.5 3D foot estimation and automatic gait cycle segmentation

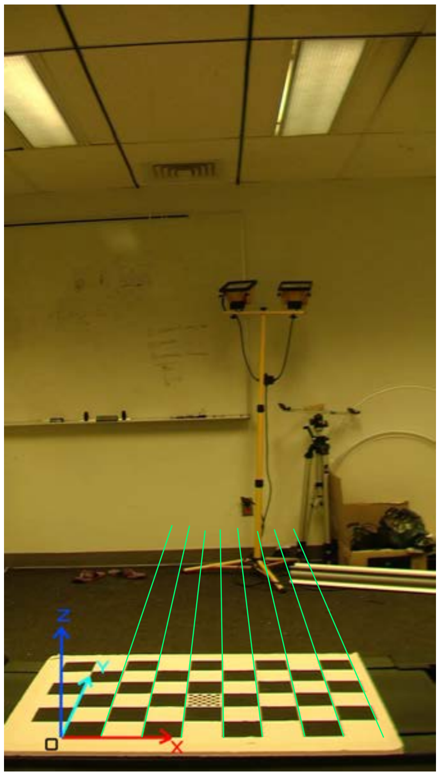
In gait analysis, the stride length of each foot is also one of the key factors. With the subject walking on the treadmill, the stride length can be measured by the Euclidean distance between the most left and most right positions of a foot when it is on the ground. In our system, we can correctly locate the foot (or more exactly, the heel) at each frame and easily calculate the stride length in pixels, which is not very useful in real applications. We thus propose a method to convert it into real measure units (ex. centimetres) inspired from [48]. As an easy way to solve it, we can simply measure the stride length in pixels first, then convert pixels to centimetres. This is true in theory since the subject is walking parallel to the camera plane. But we found that, in our experimentation, the foot trajectory within a cycle of walk is not always parallel to the walking direction. Due to this nature, the precision of the simple method is limited. In order to achieve higher precision, we propose another method that is able to estimate the foot trajectory direction as well as calculate the stride length directly in centimetres.

This process is started once the video recording is done from either left or right camera and all skeleton models for every frame are reconstructed. This step can be considered as a post-process to improve one gait characteristic, i.e, the stride length. In [48], we calculate the homography matrix to transform a 2D point in the image plane to 3D world coordinates on the ground plane. The limitation of this method is that it can only correctly locate the 3D position of the foot if it is on the ground. During a gait cycle, there are times that the foot swings in the air (called swing phase) and times that it stays relatively stationary on the ground (stationary phase). The duration of each phase varies but normally each one occupies approximately 50% of the time of a gait cycle. In order to calculate a stride length, since the subject is walking on the treadmill, we need to know exactly where the extreme left and extreme right 3D positions of the foot on the ground are. In this section, we only introduce the process for the left camera since the same and independent process is done for the right camera.

First of all, we compute the homography matrix by placing a checkerboard on the treadmill surface. We chose the bottom left corner of the chessboard as the origin co-



Left Camera



Right Camera

Figure 3.9: A chessboard is placed on the treadmill to compute the homography H between chessboard plane and image plane. The world origin is locate at O; red, cyan and blue arrows represent the X, Y and Z axis respectively. All green lines, which are parallel to each other in real world, now intersect at one vanishing point in image plane.

ordinate P_O of the 3D coordinate system, as shown in Fig. 3.9, along with the 3-axis represented by red, green and blue lines. Starting from P_O, we sample a list of 12 points located at different corners of the black squares in the image plane (p) and in real world 3D coordinates (P). The 3×3 homography matrix H is then estimated by solving the linear system

$$\begin{bmatrix} X^w \\ Y^w \\ 1 \end{bmatrix} = P = Hp = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \\ 1 \end{bmatrix} \tag{3.14}$$

where (X^w, Y^w) are the $x-$ and y-coordinate of a pixel in the 3D plane; $(\mathbf{u}, \mathbf{v})$ are the $x-$

and y-coordinate of a pixel in the image plane.

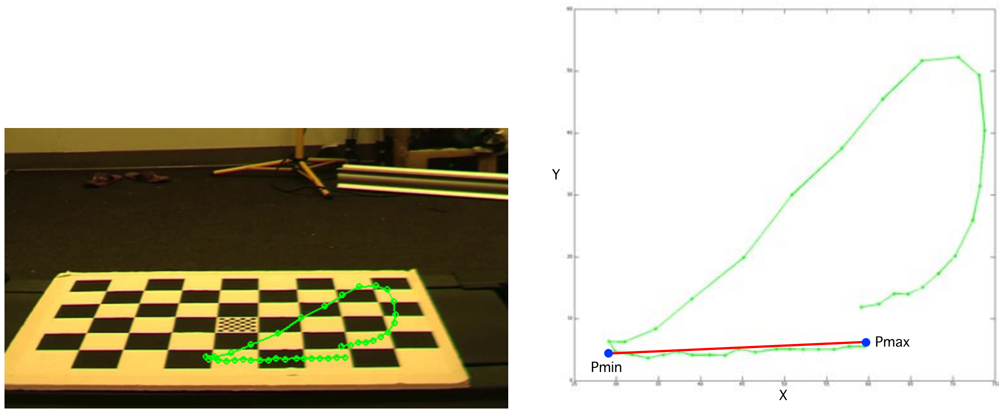
Second, regardless of the swing or stationary phase, we build the 3D foot trajectory $<X^w, Y^w, N_f>$, where N_f is the frame number, using the homography matrix H calculated before. This trajectory is then subdivided into sub-trajectory according to time frame N_f in such as way that each sub-trajectory represents a complete gait cycle. By dividing this way, we avoid the confusion between cycles when estimating the walking range in the next step. The gait cycle segmentation is described as follows: by tracking the 2D position of the foot overtime, every time its x-component reaches local minimum (extreme left point in the image plane) then we mark this time as the end of one cycle and the beginning of the new one. This $< starting, ending >$ information of each cycle is used in the next chapter when we performed the validation between our method and others over a fixed number of cycles.

In the stationary phase, since the foot is relatively fixed to the treadmill panel and the panel is moving, its (X^w, Y^w) information through time forms a linear trajectory observed by the left camera. This line orientation is close but not identical to the O_y axis in our system which represents the walking orientation. In the other hand, the foot locations in the swing phase generate an unknown curve because we actually observe only their projection on the ground from the camera point of view, see Fig. 3.10. The higher the foot is, the farther its projection is on the ground and thus is not the real 3D foot positions. The idea therefore is how to extract only a set of the foot positions while the foot stands idly on the ground, other foot positions are discarded since they are just noise. This observation leads us to an efficient way to determine the walking range, and furthermore to calculate the stride length.

As mentioned above, the stationary phase occupies about 50% of the total time of a gait cycle. Taking this information into account, based on the foot position trajectory, we repeat the process, as described in Algorithm 5, for each sub-trajectory to obtain the walking range in x-axis $[X^w_{min}, X^w_{max}]$ as well as the estimated line orientation θ_f. The trajectory L_{trajec} is the line connecting two points $P_{min} = (X^w_{min}, Y^w_{min}, 0)$ and $P_{max} = (X^w_{max}, Y^w_{max}, 0)$. The stride length for each gait cycle ($L^{\mathbf{m}}_{stride}$) is then easily deduced by computing the distance between P_{min} and P_{max}. The final stride length (L_{stride}) is the average over all stride lengths for each sub-map $L^{\mathbf{m}}_{stride}$. Fig. 3.10 visually demonstrates a sub-map and how we refine points to estimate the walking range.

Algorithm 5: Points refining to calculate the stride length

```
Input: foot position sub-map $\mathbf{m}$
Output: Walking range $P_{min}$ and $P_{max}$, the stride length $L^{\mathbf{m}}_{stride}$
1  $SM_{sorted} \leftarrow$ sort sub-map with the order of ascending value of $y$ component
2  $y_{min} \leftarrow$ the $y$ component of the first element of $SM_{sorted}$
3  $y_{max} \leftarrow$ the $y$ component of the last element of $SM_{sorted}$
4  $y_{init} \leftarrow$ the $y$ component of the middle element of $SM_{sorted}$
5  $y_{step} \leftarrow \frac{size(SM_{sorted})}{10}$ (the step between each iteration)
6  $L_{pre}$ is the line estimation in the previous iteration
7  $L_{now}$ is the line estimation in the current iteration
8  $y \leftarrow y_{init}$
9  while $y > y_{min}$ do
10     $M \leftarrow$ points fall within $[y, y - y_{step}]$
11     $L_{now} \leftarrow$ 3D line estimation over M
12     if Angle between $L_{pre}$ and $L_{now}$ is small then
13         break;
14     else
15         $L_{pre} \leftarrow L_{now}$
16         $y \leftarrow y - y_{step}$
17 $P_{min} \leftarrow$ the first element of $M$
18 $P_{max} \leftarrow$ the last element of $M$
19 $L^{\mathbf{m}}_{stride} \leftarrow$ Euclidean distance between $P_{min}, P_{max}$
20 return $P_{min}, P_{max}$
```



A B

Figure 3.10: The overview of our 3D foot trajectory estimation. (A) 2D foot locations (green line) in one gait cycle in image space. (B) 3D trajectory (red line) retrieved in world space in which P_{min}, P_{max} represent the minimum point and maximum point respectively along the horizontal axis.

3.4 Synchronization between two cameras

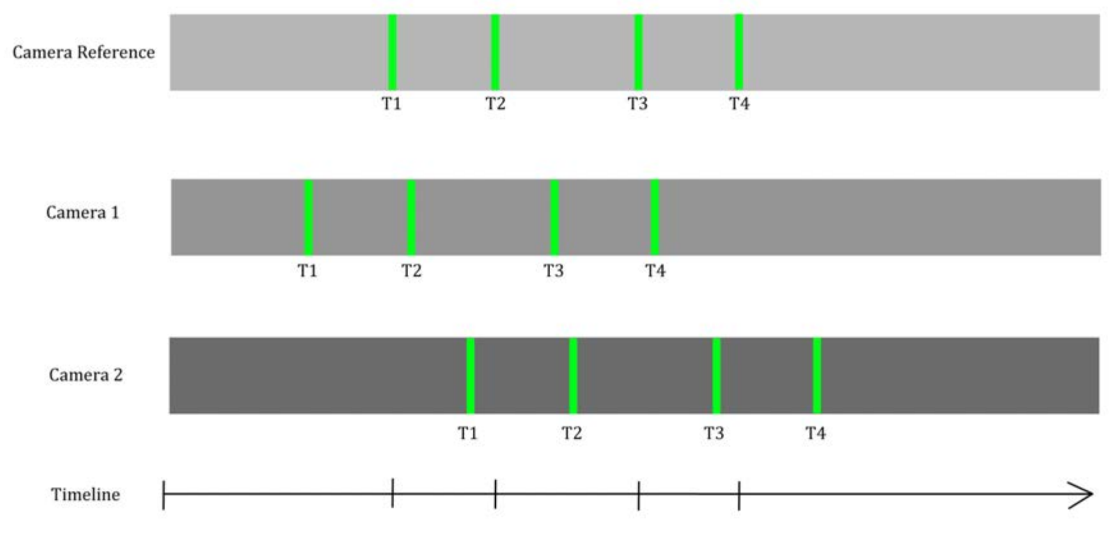
Since each camera only tracks body halves of the subject, the synchronization between two cameras is essential to establish the relation between the left and right skeleton models, thus allowing more advanced gait assessment (which will be introduced in the next chapter). There are many ways to do the synchronization, either manually or automatically. In our thesis, we only focus on automatic synchronization methods. In [44], the authors use flashes produced by a still-photo device to cause a sudden sharp brightness change in the frame instantaneously. The flash duration must be long enough to be captured by all the involved cameras in one (or two) frames. Finally, taking one camera's frame as a reference, all other cameras's frames are aligned accordingly. This approach, however, only works in a small and quite dim light environment to have the maximum brightness difference before and after the flash occurs. Fig. 3.11 visually demonstrates the overview of our method.

In our system, we have to use four additional light sources (placed at the four corners of the room, see Section 3.1), in addition to the standard room lighting to make sure each camera records well-lit videos and to reduce shadows. Flashes therefore are not useful in our case since they can only produce small (almost negligible) light changing effect, thus it is not easy to locate in which frames they happen. To cope with this situation, we make use of the on- and off-state of the standard room lighting in our experimentation room. The detail of the process is described as follows

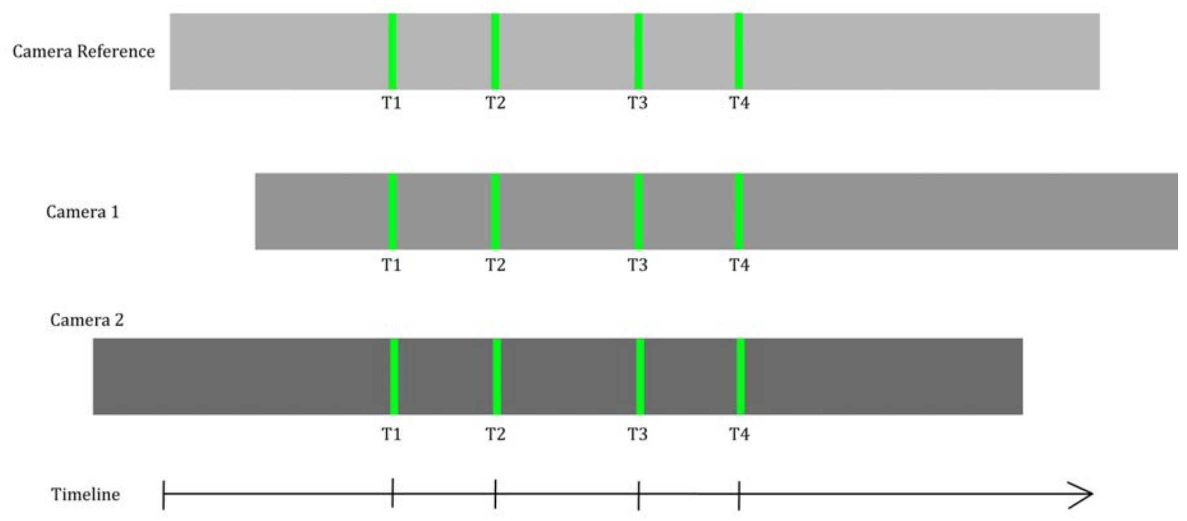
1. First, we turn the light off at time T_1 then turn it back on at time T_2. T_2 is set as 1 second after T_1 in our implementation.
2. Repeat the same process above N_{switch} times. Each time is after a short amount of time (about 2 seconds) from the previous time. For simplicity, we only explain for the case $N_{switch} = 2$, producing T_3 and T_4.
3. For each camera c, we locate their times $T_{c1}, T_{c2}, T_{c3}, T_{c4}$ based on thresholding the overall luminance changes, see Fig. 3.11.
4. Keeping one camera as a reference, denoted by r, for camera c, the aim is to minimize an error function with respect to a time delay Δ:

$$\Delta^* = \operatorname*{argmin}_{\theta_{min} \leq \Delta \leq \theta_{max}} \sum_{m=1}^{4} \frac{|T_{cm} + \Delta - T_{rm}|}{4} \tag{3.15}$$

   where $\theta^*_{min} = \min(0, T_{m1} - T_{c1})$, $\theta_{max} = \max(0, T_{m1} - T_{c1})$.
5. Δ^* is the time difference between camera c and camera r that we are looking for.



A



B

Figure 3.11: The synchronization process. (A) We first locate T_1, T_2, T_3, T_4 in each video. (B) Camera 1 and 2 are aligned with respect to the camera reference by aplying the optimized time delay Δ^* found in Eq. 3.15.

Using 4 points of time T_1, T_2, T_3, T_4 (or more depending on N_{switch}) to compute the synchronization is necessary to have reliable constraints when aligning frames, and thus minimizing the risk of having bad synchronization, see Fig. 4.1. Depending on the customizeable N_{switch}, the system will do the synchronization accordingly. Our method above can be used for multiple (more than two) cameras.

We have also investigated the possibility to extract the 3D arm location from its 2D positions (see Annex B). The results are quite promising but the technique is still complex and has some limitations. Future work is needed to improve performance and accuracy.

CHAPTER 4

EXPERIMENTATION AND VALIDATION

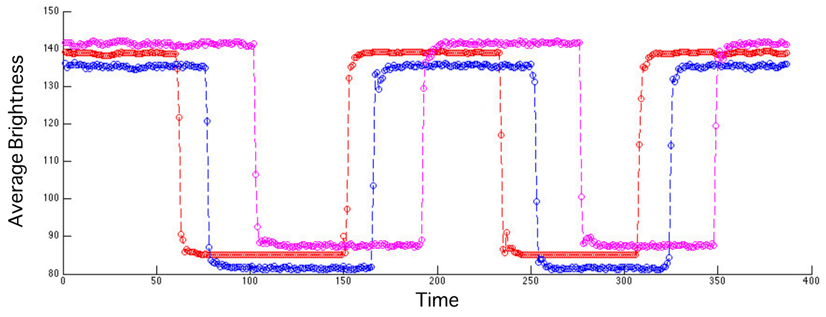
In this chapter we present our 2.5D skeleton reconstruction on two different subjects. For validation, the comparison between our method vs. ground-truth and the Kinect are also performed. Furthermore, we calculate the gait characteristics (joint angles: knee, elbow, ankle; stride length) and demonstrate that our results can lead to an efficient asymmetry assessment that will be presented in the next chapter.

4.1 Synchronization between cameras

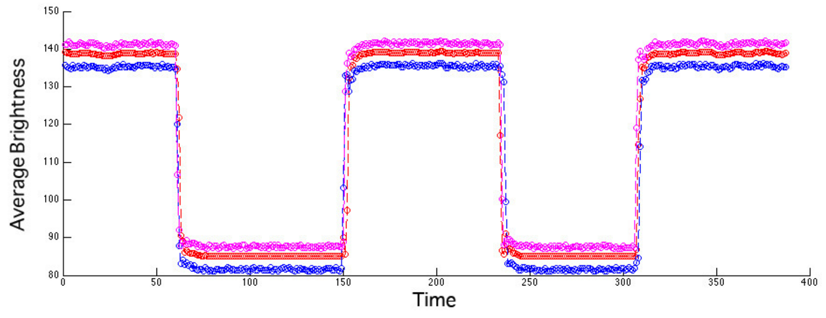
In our experimental study, we have to do the synchronization between 3 cameras: our left and right camcorder and the Kinect used for comparison. For testing the synchronisation method proposed in Section 3.4, we recorded 10 different videos for each camera with various values of N_{switch} for comparison purposes. In summary, the higher the value of N_{switch} (Section 3.4), the less error we have. Some of the minimized mean error $\overline{\varepsilon}$ over 10 test cases are shown in Table 4.I. The relation between N_{switch} and the mean error $\overline{\varepsilon}$ is shown in Fig. 4.2. The mean error $\overline{\varepsilon}$ is defined as (see also Equation 3.15):

$$\overline{\varepsilon} = \min_{\Delta} \sum_{m=1}^{2\times N_{switch}} \frac{|T_{cm} + \Delta - T_{rm}|}{2 \times N_{switch}} \tag{4.1}$$

To avoid potential errors caused by automatic light compensation of the camera (i.e., white balancing), we purposely turn off this functionality both on our cameras and on the Kinect before doing this experimentation. By using ceiling lights, there are normally about 3 frames affected by a light switching. This duration affects the accuracy when aligning frames between cameras and thus causes a little disturbance of the $\overline{\varepsilon}$ in Table 4.I for N_{switch} larger than 3. Further improvement is needed to reduce this duration to 1 frame to ensure better synchronization (e.g. by putting a flashing device at a predefined area of the image and only observing the light changing in that area).



A



B

Figure 4.1: Our synchronization result for 3 cameras: left camcorder (red line), right camcorder (blue line) and Kinect (magenta line). (A) Before doing synchronization. (B) After doing synchronization with the left camera as the reference.

Table 4.I: Synchronization error $\overline{\varepsilon}$ (in number of frames) between our left, right camcorders and the Kinect.

N_{switch}	Left Camcorder (reference)	Right Camcorder	Kinect at 30 fps
1	0	0.62	0.85
3	0	0.38	0.47
5	0	0.31	0.37
7	0	0.26	0.25
9	0	0.24	0.29
10	0	0.20	0.26

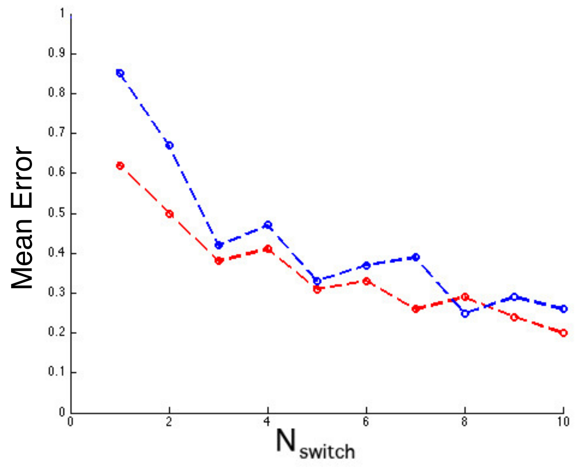


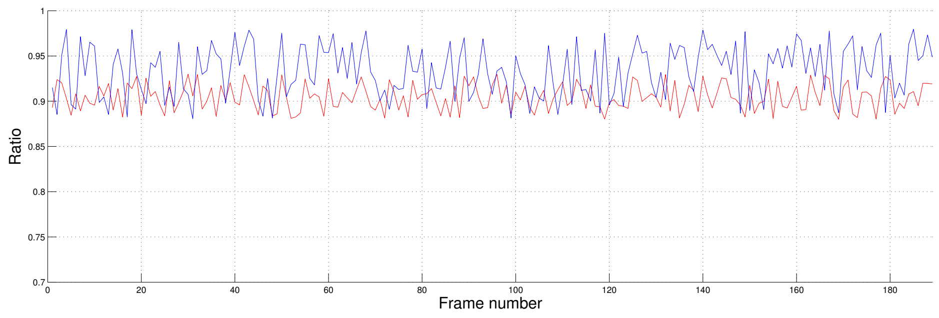
Figure 4.2: Graphs representing the relation between N_{switch} and error $\overline{\varepsilon}$ for the right camera (red line) and for Kinect (blue line). The error significantly dropped until $N_{switch} = 3$, and less significantly after that.

4.2 Validation with the Kinect and ground-truth measurements

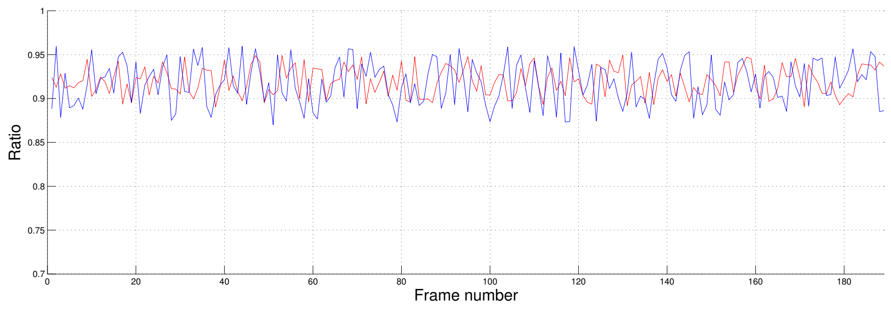
To perform our tests, we chose two JVC camcorders (model GZ-HD6U) as our monocular cameras (60 fps with de-interlacing [2]) in our system because they are readily available in our laboratory and they also produce good image quality. The main issue with this type of camera is that it has a fixed shutter speed that produces motion blur on moving objects. We addressed this problem by both limiting the maximum speed of walking and applying the edge-enhancing technique (see Section 3.2). Additionally, since our experimentation room ceiling lights cause a lot of shadow on the subject's body, we also used four additional light sources placed at the four corners of the room to make sure the light is well diffused all around and to reduce shadows on the subject's body. To perform a fair test with the Kinect, we used the latest version v1.8 of the Kinect for Window software development kit (SDK) [42] and placed the Kinect in front of the treadmill since the SDK works best with this frontal point of view. We also tested placing the Kinect on the side (same position as our camcorders), but the reconstruction results were poor when legs crossed in a walk cycle. We thus eliminated this solution for side view reconstruction. Since our system has capacity of processing 60 frames-per-second (fps) (with de-interlacing), we had to scale down the frame numbers to 30 fps to match that of the Kinect and of the ground-truth (frame-rate of the camera is already 30 fps). This is only to facilitate the comparison step between the 3 methods afterward.

The ground-truth was obtained by tracking red stickers put on subject's joints, as shown in Fig. 4.4 and Fig. 4.6. We extracted the centroids of circular red areas corresponding to the markers by performing a predefined-red-color filtering on the input frame producing a binary image. The Hough transform was then executed on the binary image to detect circles, each circle found indicating a joint of the skeleton. To avoid the effect of noise or occlusions due to moving limbs, we manually verified each frame and corrected wrong joints if necessary to make sure we obtained 100% accuracy.

In the previous chapter, we mentioned the parameter ρ_{ua} which represents the foreshortening level of the upper arm (see Section 3.3.3). This information comes in handy to reconstruct the arm fully in a 3D environment (see Annex B). However, due to the nature of the swinging arm movement during a walk, the upper arm does not suffer much from foreshortening. That is why we only set the range of ρ_{ua} from 0.7 to 1 in our experimentation. We also measured ρ_{la} which represents the foreshortening level of the forearm and illustrated the variance of both ρ_{ua} and ρ_{la} for subjects 1 and 2 in Fig. 4.3. This result proved our assumption that the foreshortening has little effect on a walk and can usually be neglected. Furthermore, we found that the variation margin of ρ_{ua} was more consistent than that of ρ_{la} because (1) the forearm estimation precision (see Section 3.3.2) depends on the skin extraction result (explained in Section 3.2), and (2) the upper arm estimation process took into account the upper arm segment length (see Section 3.3.3).



A



B

Figure 4.3: Variation of foreshortening level ρ_{ua} of upper arm (red line) and foreshortening level ρ_{la} of forearm (blue line) for subject 1 (A) and subject 2 (B).

Once the pose was estimated with our system, the Kinect SDK and the ground-truth, we started calculating the gait characteristics, as explained in Section 3.1, on these three skeletons. Since we only focused on calculating angles between joints and the stride lengths, all the comparisons between our system with the Kinect SDK and ground-truth were performed directly without any additional processing. There was no need to make some spatial transformation on the skeleton coordinates from the Kinect space into our camera space and vice versa. The angle can be calculated directly from the body joints in each skeleton model. Furthermore, we also can get both walking orientation as well as the walking range. These informations could be interesting to human recognition based on his gait in the surveillance domain. All these experimentations, in Fig. 4.7, Fig. 4.8, and Fig. 4.9, were performed over 20 cycles because we only chose a short consecutive period at which the subject walks most stable (relatively stationary at the center of the treadmill, stable speed) so that each method (ours and the Kinect) performs best. In these 20 cycles, their gait cycle durations only differed by 2 or 3 frames (vary from 38 to 41 frames in my experimentation). We therefore always cut them at 38 frames for convenience when calculating average curves.

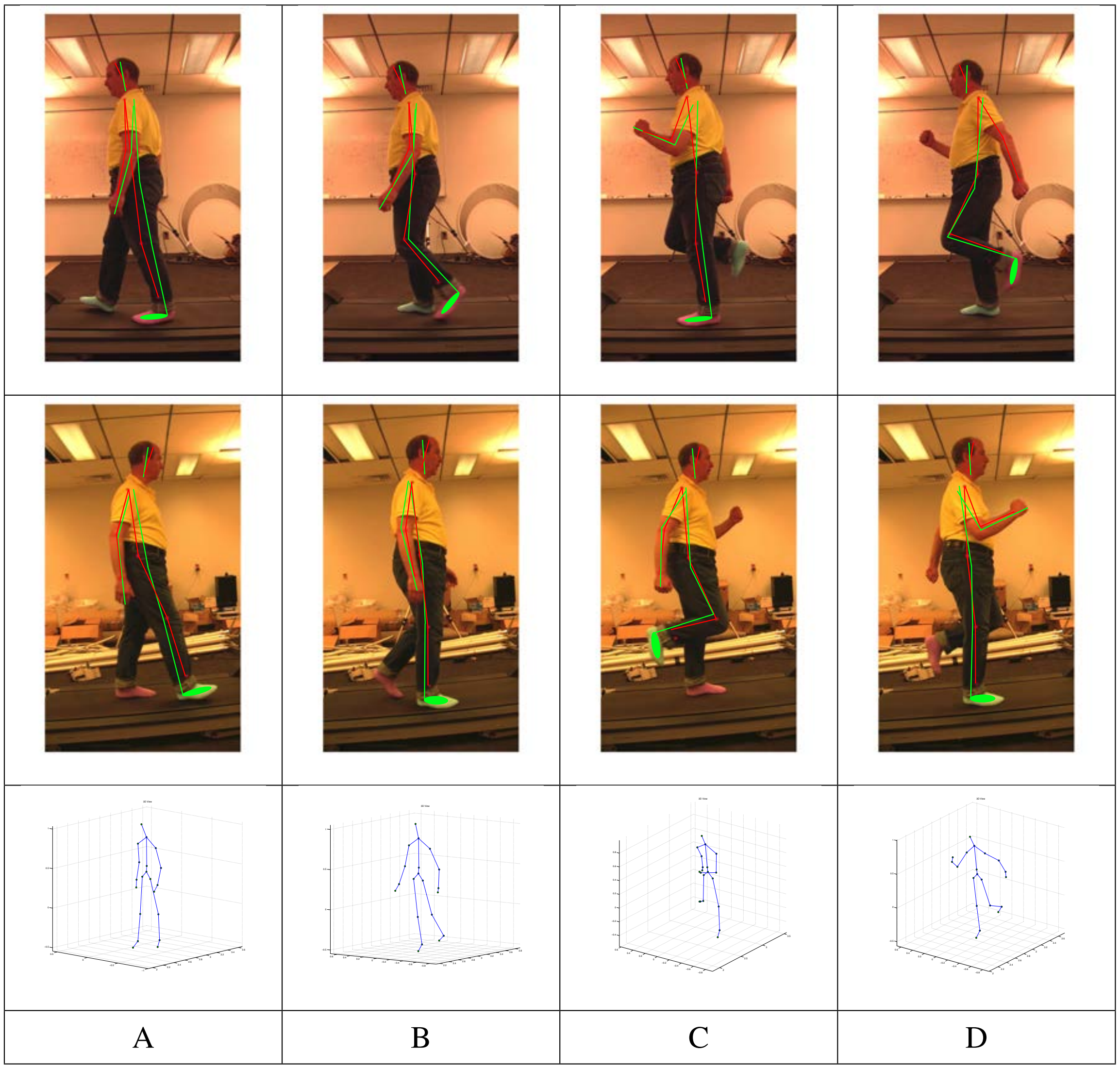


Figure 4.4: Our skeleton reconstruction on subject 1. The first row shows some of our testing frames captured from the left camera, the second row shows their corresponding frames from the right camera. Red lines indicate the ground-truth formed by red markers placed on the subject's body, our pose estimation produced the green lines and green ellipses. The third row shows the 3D skeleton from the Kinect SDK. The Kinect tends to fail in cases C,D where it could not see the lower leg due to occlusions.

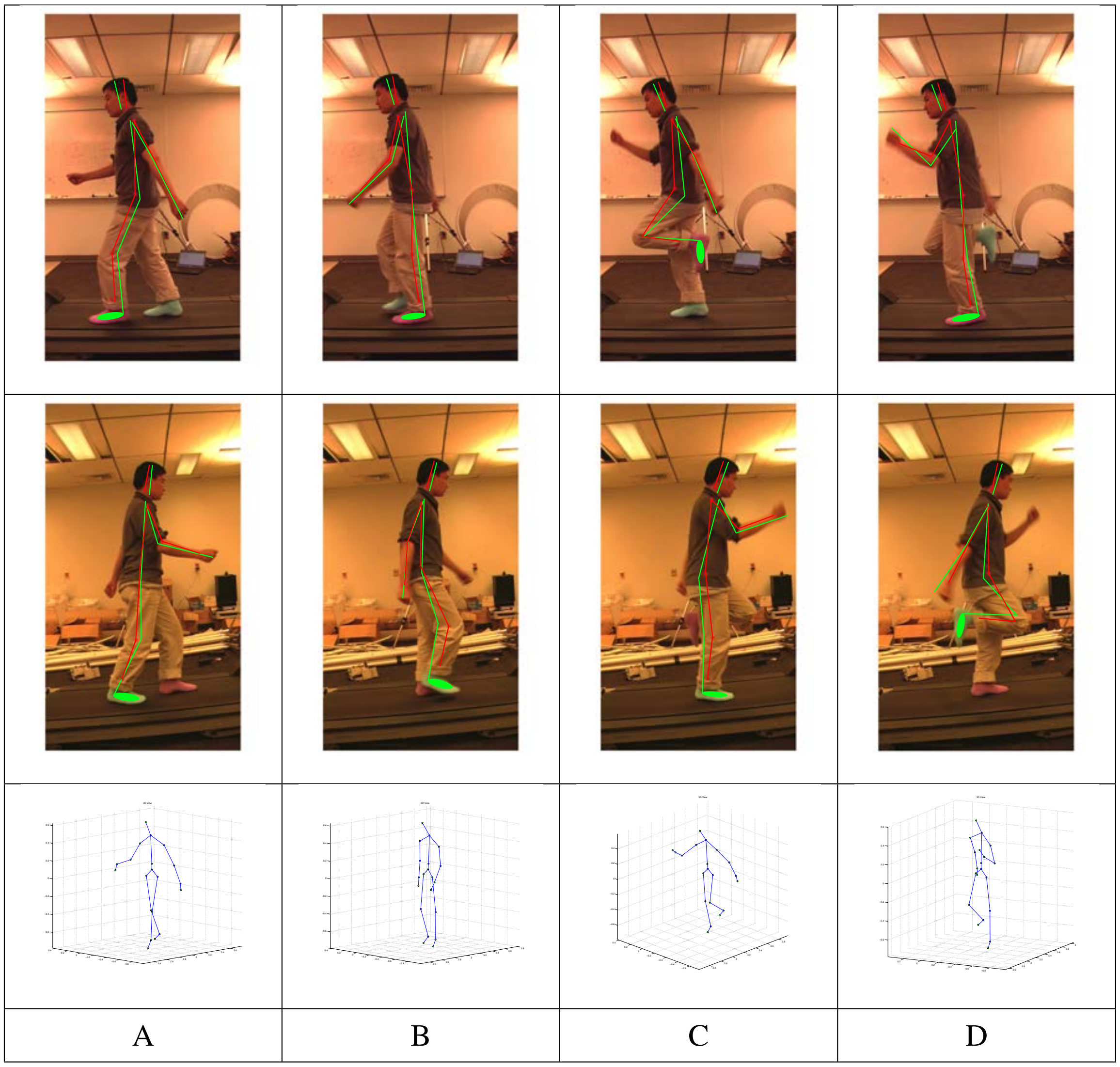


Figure 4.5: Our skeleton reconstruction on subject 2. Just like subject 1, the Kinect failed in cases C,D where it could not see the lower leg due to occlusions.

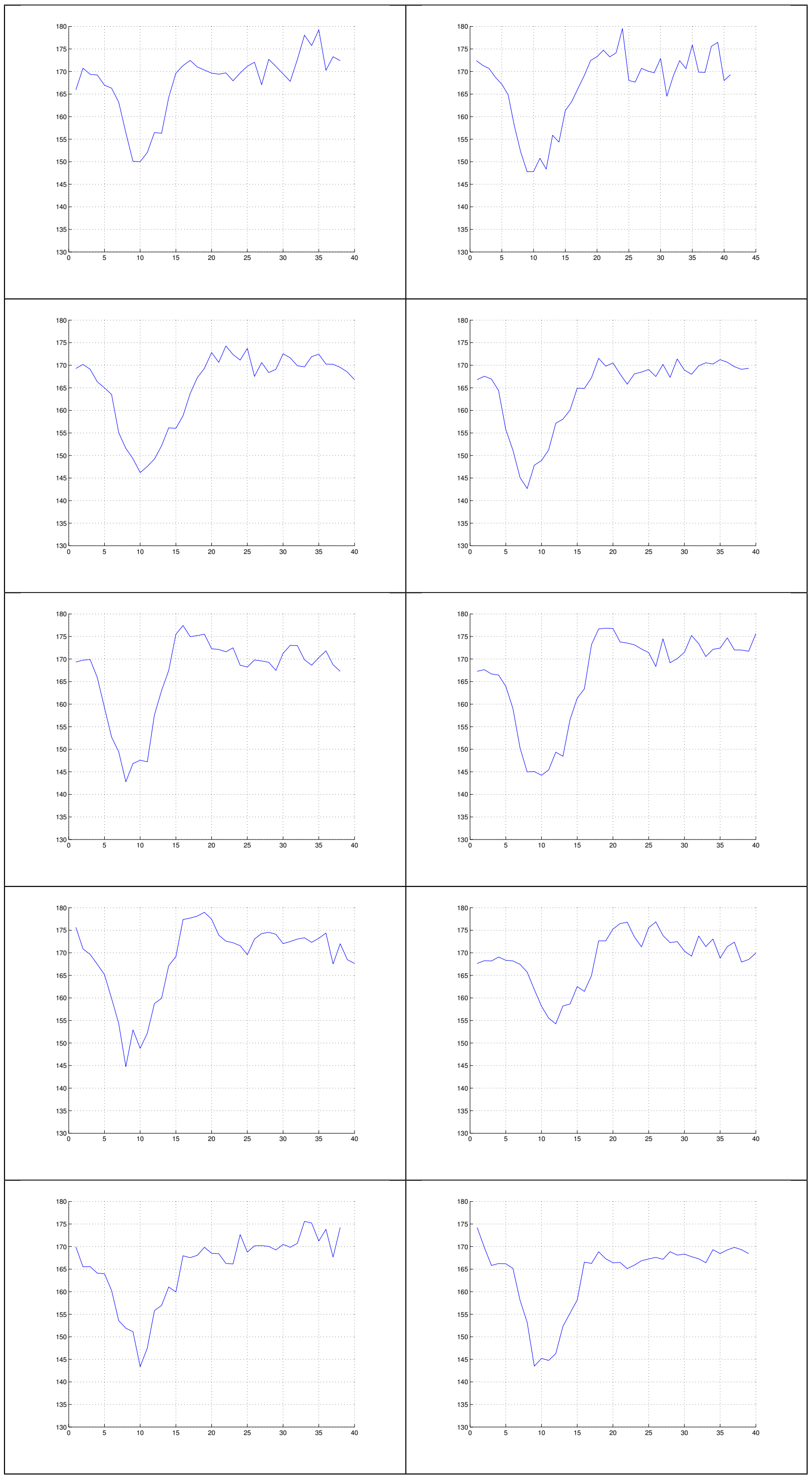


Figure 4.6: Ten typical curves which correspond to 10 consecutive cycles produced by our method. Each gait cycle duration varies very lightly from 38 to 41 frames. For convenience, we simply cut all cycles at frame 38 when calculating the average curve over 20 cycles.

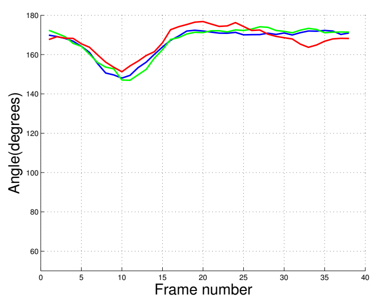
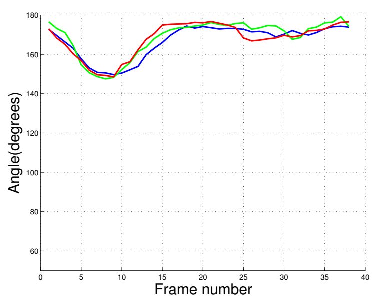
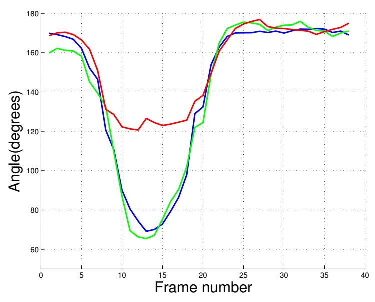
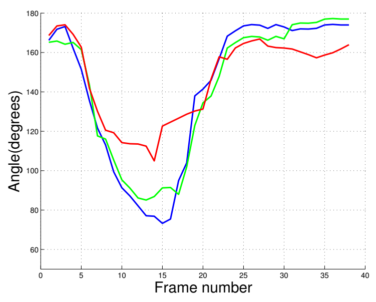
	
	
Knee (Subject 1)	Knee (Subject 2)

Figure 4.7: Average curve representing the angle variation at the left knee estimated from ground-truth (blue line), Kinect SDK (red line) and our method (green line) within a gait cycle. The horizontal axis is the frame number in a walking cycle. The first row is the result observed from Test 1, the second row is from Test 2. Our method outperformed the Kinect SDK in Test 2 and successfully located the minimum knee joint angle while the Kinect SDK failed due to lower leg occlusions. The curves were obtained by averaging 20 cycles.

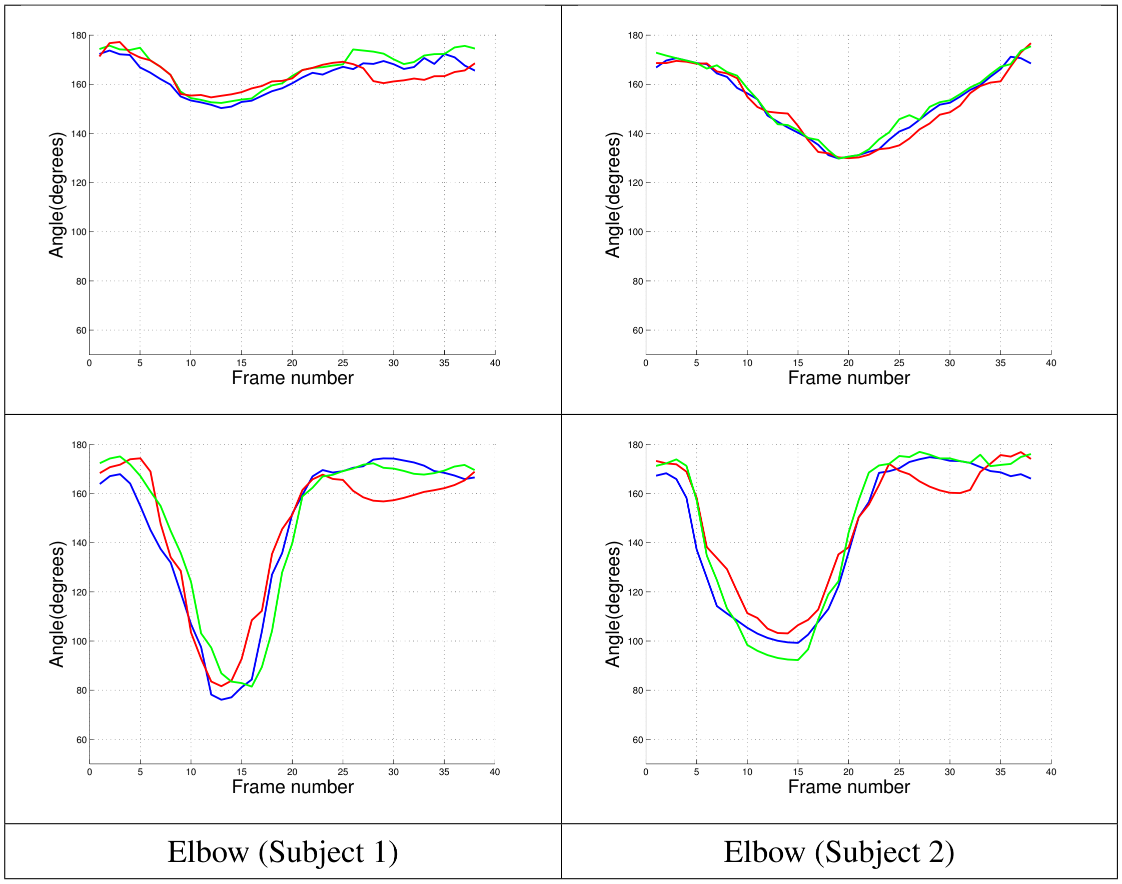


Figure 4.8: Average curves representing the angle variation at the left elbow estimated from ground-truth (blue line), Kinect SDK (red line) and our method (green line) within a gait cycle. The curves were obtained by averaging 20 cycles. The results from the three methods were quite similar in this case.

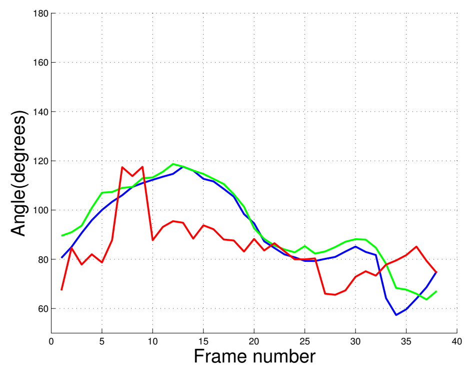
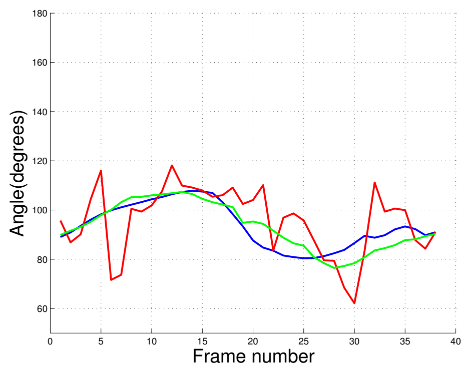
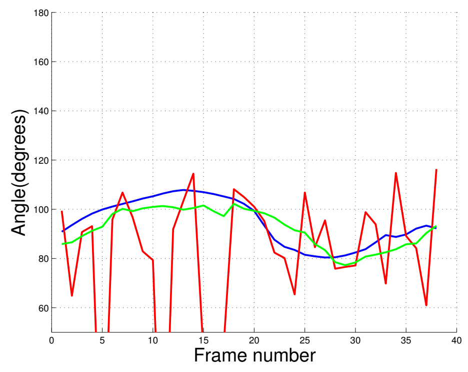
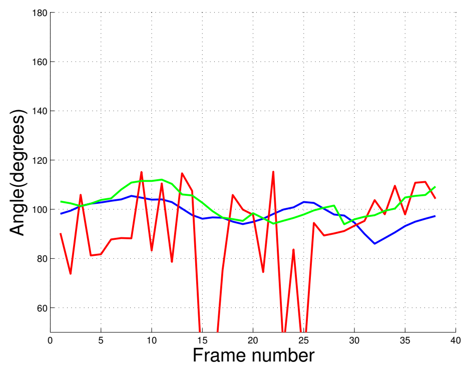
	
	
Ankle (Subject 1)	Ankle (Subject 2)

Figure 4.9: Average curves representing the angle variation at the left ankle estimated from ground-truth (blue line), Kinect SDK (red line) and our method (green line) within a gait cycle. The curves were obtained by averaging 20 cycles. The ankle angle estimated by the Kinect is inconsistent and produces much larger bias errors than ours.

To compare both the accuracy and the robustness between our method and that from the Kinect SDK, we firstly performed a normal walk (with limited movement of limbs), named Test 1, and then an exaggerated walk (with large rotation amplitude of the joints similar to running), named Test 2, on two different subjects. After getting the skeleton from each method, we monitored the joint angle changes at knee and elbow in order to evaluate the capacity to analyze gait for each skeleton model. Examples of these changes are represented by the curves in Fig. 4.7, Fig. 4.8, and Fig. 4.9. The closer to the ground-truth the curve is, the more correct the result is. In Test 1, both our method and the Kinect SDK gave very little differences compared to the ground-truth curves at knee and elbow. In Test 2, the Kinect suffered heavily from occlusions due to its frontal view, and as a result, it cannot reach the correct minimum angles at knees. At the elbow, the Kinect SDK produced noisy but still acceptable results. Our method, in the other hand, gave similar curves as the ground-truth at the knee, and acceptable results at the elbows like the Kinect. In some cases in which the lower arm moves too fast, the arm estimation becomes harder and this is the reason our result at arm had more noise. But we were still capable of successfully reaching the minimum angles just like ground-truth. At torso, our torso shape model performed very well when giving similar torso orientation as the ground-truth's. At the foot in both tests, our method performed better than the Kinect SDK since the Kinect sometimes produced unstable foot estimations. At other joints like torso, head, etc., the estimation from the Kinect were more accurate and consistent than ours. This can be predictable since we only perform a simple ellipse fitting on these joints. In Tables 4.II and 4.III, we show the detailed average errors in degrees for each segment (4.II) and joint (4.III) estimated by our method, and the Kinect compared with those from ground-truth red markers.

As described in Fig. 1.1(A), we require that each camera view direction is always perpendicular to the walking direction. We therefore also experimented the case, in which the left camera view direction is at 80° with respect to the walking direction, see Fig. 4.10, to evaluate the robustness of our system. In this test, with 10° error on camera position, we reported average angle differences of 4.41°, 5.11° and 3.93° compared to the original perpendicular camera position at knee, elbow and ankle joint respectively. The differences are acceptable considering that such a large camera positioning error is unlikely with a careful setup of the system and that the angle errors will therefore be much lower in practice.

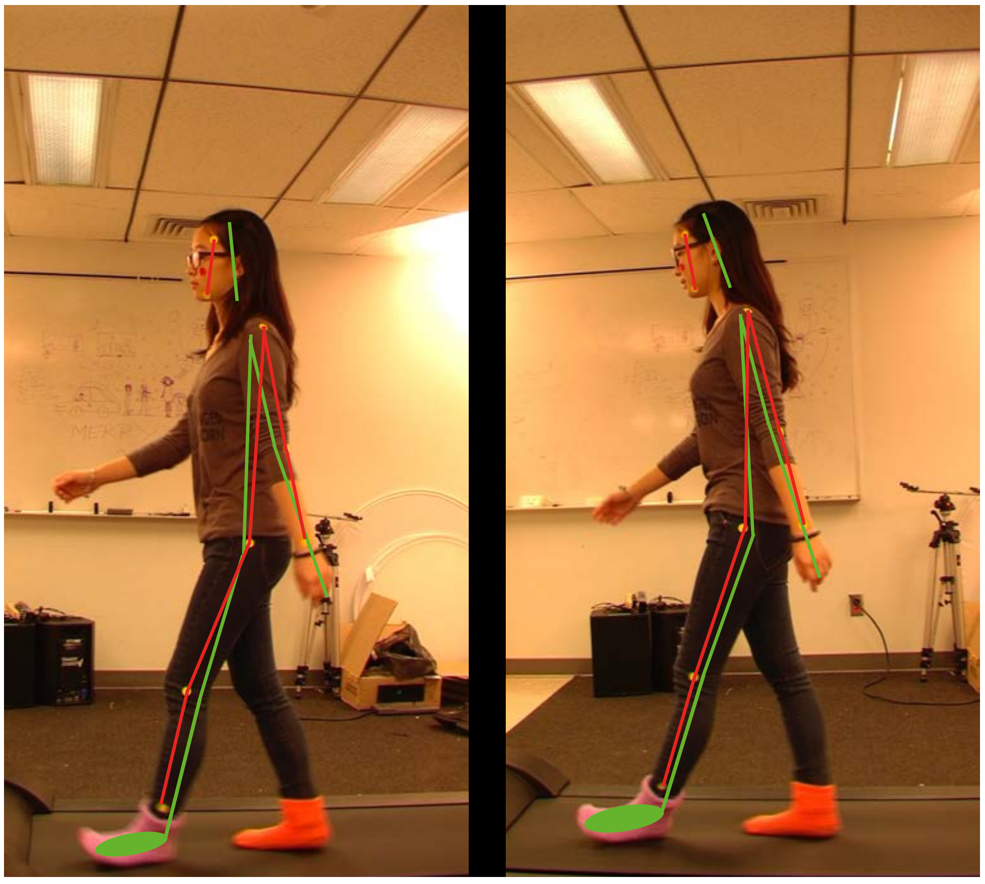
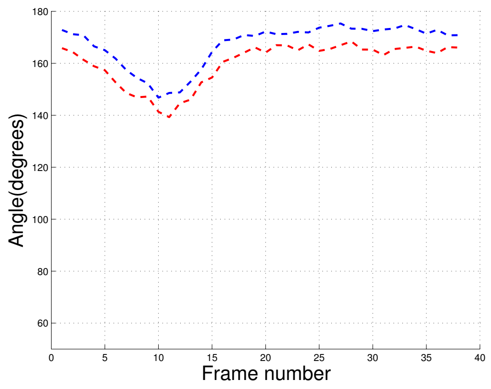


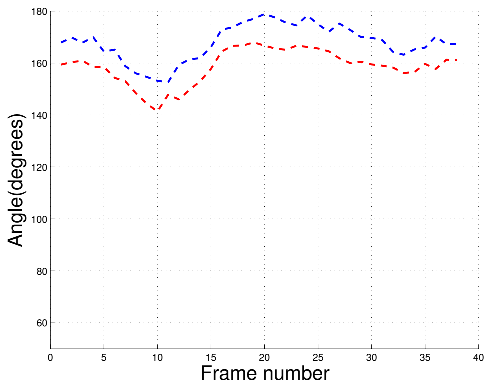
Figure 4.10: (A) The left camera forms an angle of 80° to the walking direction (wrong). (B) The left camera forms an angle of 90° to the walking direction (correct). The effects of (A) to the final skeleton estimation were empirically negligible.

We also studied the reliability of our method by illustrating the variability around the average curve of a series of gait cycle curves (as shown in Fig. 4.7, Fig. 4.8, and Fig. 4.9) in Fig. 4.11, Fig. 4.12 and Fig. 4.13. The variability is represented by the upper (blue dash lines) and lower (red dash lines) curves which are computed by adding and subtracting the standard deviation, which is for each frame and not the entire curve, to the average curve respectively. The closer these two lines are, the more consistent the estimation is. For an objective comparison, we only performed the consistency test on the normal walk of subject 2 in which the Kinect did not suffer from view occlusions.

The consistency result seemed equal at both knee (Fig. 4.11) and elbow (Fig. 4.12), but extremely noisy for the Kinect at ankle (Fig. 4.13).

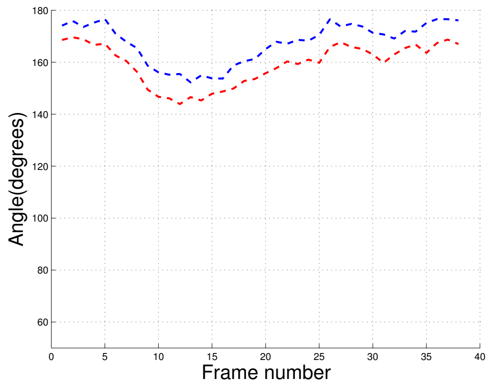


A

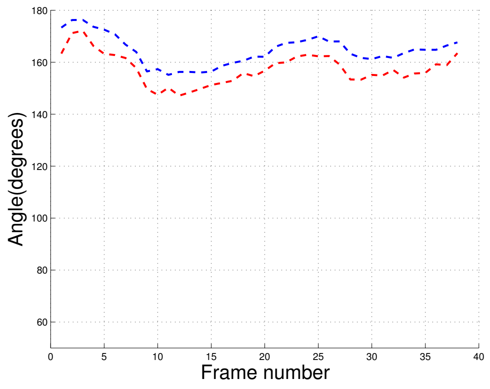


B

Figure 4.11: The upper (blue dash line) and lower boundaries (red dash lines) calculated at knee around the average curve for subject 1. The upper curve is obtained by adding the standard deviation to the average curve. The lower curve is obtained by subtracting the standard deviation to the average curve. The standard deviations were calculated over 20 cycles. (A) Results from our method. (B) Results from the Kinect.

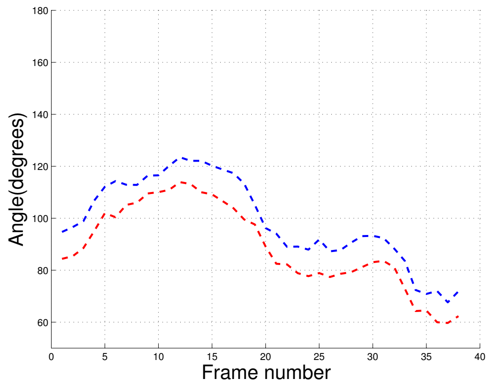


A

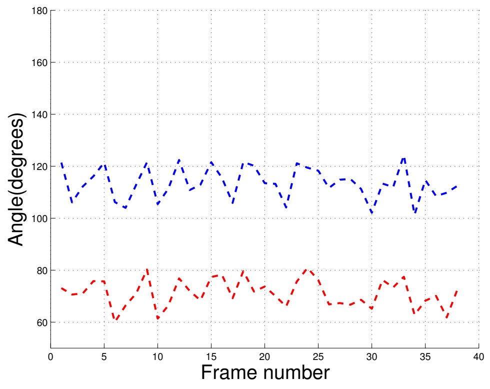


B

Figure 4.12: The upper (blue dash line) and lower boundaries (red dash lines) calculated at elbow around the average curve for subject 1. The upper curve is obtained by adding the standard deviation to the average curve. The lower curve is obtained by subtracting the standard deviation to the average curve. The standard deviations were calculated over 20 cycles. (A) Results from our method. (B) Results from the Kinect.



A



B

Figure 4.13: The upper (blue dash line) and lower boundaries (red dash lines) calculated at ankle around the average curve for subject 1. The upper curve is obtained by adding the standard deviation to the average curve. The lower curve is obtained by subtracting the standard deviation to the average curve. The standard deviations were calculated over 20 cycles. (A) Results from our method. (B) Results from the Kinect.

Table 4.II: Average of absolute differences of each human **segment** (see Fig. 3.2) from our method (columns 1 and 3) and from the Kinect SDK (columns 2 and 4) compared to the ground-truth skeleton model of subject 1 in two tests. The average stride length error (in both absolute value in centimetre and in percent compared to mean stride length) are also shown. These results were calculated over 20 cycles. The values in bold indicate which method (our method or Kinect) produced the largest error.

Body part	Our method (Test 1)	Kinect (Test 1)	Our method (Test 2)	Kinect (Test 2)
Head	**9.13**°	6.22°	**9.97**°	7.72°
Torso	**8.58**°	7.12°	**8.49**°	8.18°
Upper Arm	6.76°	**7.19**°	**14.08**°	12.49°
Lower Arm	**8.01**°	6.66°	**11.72**°	9.61°
Upper Leg	7.49°	**8.68**°	6.84°	**18.62**°
Lower Leg	6.85	**9.02**°	8.02°	**21.51**°
Foot	6.39°	**12.73**°	9.25°	**19.59**°
Left Stride	8.06 cm 9.01 %	**10.29** cm **11.50 %**	13.26 cm 12.96 %	**27.47** cm **26.84 %**
Right Stride	5.32 cm 6.83 %	**13.67** cm **17.55 %**	7.23 cm 9.26 %	**24.92** cm **31.91 %**

Table 4.III: Average of absolute differences at three **joints of interest** (knee, elbow and ankle, see Fig. 3.2 blue arrows) from our method (columns 1 and 3) and from the Kinect SDK (columns 2 and 4) compared to the ground-truth skeleton model of subject 1 in two tests. These results were calculated over 20 cycles. The values in bold indicate which method (our method or Kinect) produced the largest error.

Body Joint	Our method (Test 1)	Kinect (Test 1)	Our method (Test 2)	Kinect (Test 2)
Knee	7.24°	**8.75**°	7.22°	**20.91**°
Elbow	**7.39**°	7.08°	**12.79**°	11.31°
Ankle	6.71°	**11.16**°	8.62°	**22.41**°

With these promising performances with respect to the Kinect at different joints in both test cases, we have demonstrated the potential of our methodology for gait analysis. In the next chapter, we use our method to distinguish normal from abnormal gaits in particular for quantifying the asymmetrical (left-right) walk patterns.

CHAPTER 5

APPLICATION TO GAIT ASYMMETRY ASSESSMENT

The definition of a normal walk is as follows

A method of locomotion involving the use of the two legs, alternately, to provide both support and propulsion, with at least one foot being in contact with the ground at all times.[64]

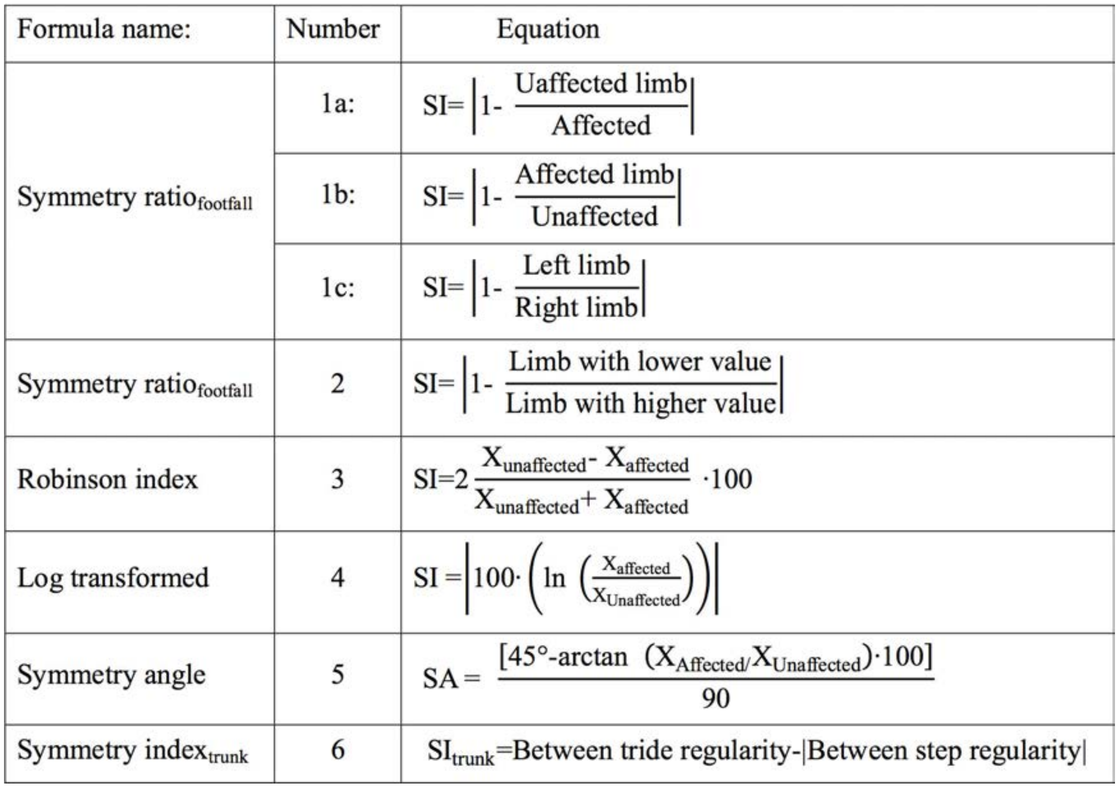
Gait analysis often involves symmetry/asymmetry assessments since gait symmetry is considered as a characteristic of a normal walk. In Encyclopedia Britannica, symmetry is defined as the correspondence of body parts in size, shape, and relative position, on opposite sides of a dividing line or distributed around a central point or axis [53]. Asymmetry, in the other hand, has no mathematically official measurement method. It generally indicates cases that are simply not symmetric. Asymmetry therefore becomes an important factor to assess if a gait is normal or not. To that end, there has been many approaches tried that to quantify symmetry in gait such as footfall symmetry ratios [54–58, 61] (Equation 1a, 1b, 1c and 2, Fig 5.1), Robinson index [60] (Equation 3, Fig 5.1), log transformed index [62] (Equation 4, Fig 5.1), symmetry angle [63] (Equation 5, Fig 5.1), symmetry index of the trunk [54] (Equation 6, Fig 5.1), etc. The definition of each equation, as shown in Fig 5.1, has its own advantages and limitations. This field is still an active research area that attracts many scientists from different fields.

In this chapter, we introduce a new simple asymmetry measure based on the successfully reconstructed skeleton model, presented in the previous chapter, that can effectively distinguish the symmetry and asymmetry of a gait.

5.1 Methodology

5.1.1 Overview

For the best gait assessment, accuracy information of the pose and a sufficiently long time of evaluation are crucial. With that ultimate aim, we propose to use a treadmill and



Formula name:	Number	Equation
Symmetry ratio$_{\text{footfall}}$	1a:	$SI = \left\lvert 1 - \frac{\text{Uaffected limb}}{\text{Affected}} \right\rvert$
	1b:	$SI = \left\lvert 1 - \frac{\text{Affected limb}}{\text{Unaffected}} \right\rvert$
	1c:	$SI = \left\lvert 1 - \frac{\text{Left limb}}{\text{Right limb}} \right\rvert$
Symmetry ratio$_{\text{footfall}}$	2	$SI = \left\lvert 1 - \frac{\text{Limb with lower value}}{\text{Limb with higher value}} \right\rvert$
Robinson index	3	$SI = 2 \frac{X_{\text{unaffected}} - X_{\text{affected}}}{X_{\text{unaffected}} + X_{\text{affected}}} \cdot 100$
Log transformed	4	$SI = \left\lvert 100 \cdot \left(\ln \left(\frac{X_{\text{affected}}}{X_{\text{Unaffected}}} \right) \right) \right\rvert$
Symmetry angle	5	$SA = \frac{[45^\circ - \arctan\ (X_{\text{Affected}}/X_{\text{Unaffected}}) \cdot 100]}{90}$
Symmetry index$_{\text{trunk}}$	6	SI_{trunk}=Between tride regularity-|Between step regularity|

Figure 5.1: The authors in [53] summarized popular equations to quantify gait symmetry. (Source: [53])

two monocular cameras placed on the left and right sides of that treadmill as shown in Fig 5.2 and compare the results with a Kinect placed in front. In the camcorder system, in order to obtain the most accuracy on foot estimation, we propose a very simple way to quickly and efficiently detect and track each foot by requiring to wear left and right socks of different colors, see Section 3.1.

Fig 3.1 demonstrates the workflow of our system: in the preprocessing step, we calculate and store the global skin and each foot color models. Then the following process is performed independently on each camcorder (left and right): we estimate the orientation of head by using ellipse fitting and of torso by dynamically computing the left and right boundaries of torso with respect to the current context (Section 3.3.1). The forehead, neck, shoulder and hip are deduced using anthropometry data [50]. The location of each foot is known based on its special color information, combining with the

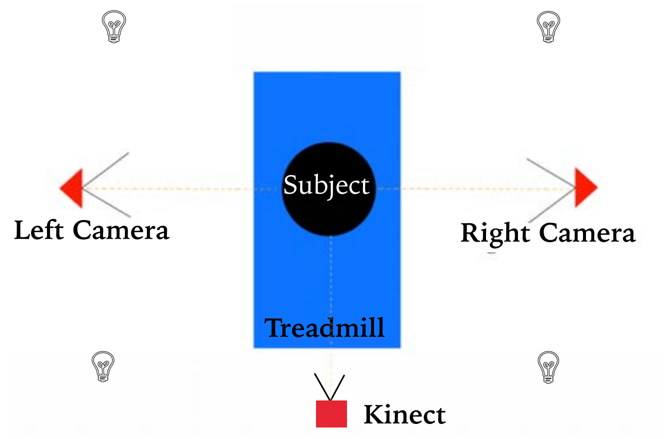


Figure 5.2: Our system consists of one treadmill and two cameras on the left and right sides of the treadmill. Four light sources are placed at the four corners of the room to ensure good light diffusion.

known hip joint and leg joint lengths gives us the correct location of the knee (Section 3.3.4). In a similar manner, the skin information helps us isolate the lower arm and a small portion of the upper arm. We then sample points along the lower arm and apply line fitting to locate the hand and elbow. The upper arm orientation is finally estimated using a particle filter (Section 3.3.2 and 3.3.3). For evaluation, we aim to compare the gait characteristics calculated from our 2.5D skeleton models and the full 3D skeleton using the Kinect SDK, which we consider the marker-less state-of-the-art until now

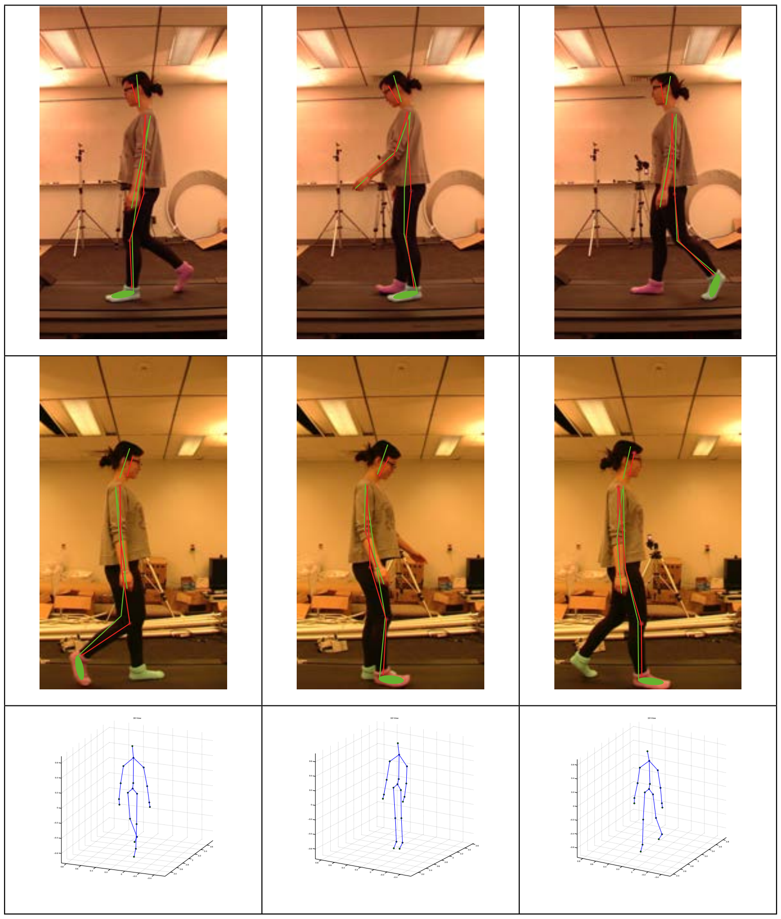


Figure 5.3: The first row shows some of our testing frames captured from the left camera, the second row shows their corresponding frames from the right camera. Red lines indicate the ground-truth formed by red markers placed on the subject's body, our pose estimation produced the green lines and green ellipses. The third row shows the 3D skeleton from the Kinect SDK.

5.1.2 Asymmetry measurements

Our ultimate aim is to clearly quantify the asymmetry between the left and right body parts in order to facilitate gait assessment afterward. We experimentally observed that the curves which reflect the angle changes at some joints of interest (such as knee, elbow, ankle, etc.) look similar (but out of phase) between the left side and their right counterpart, in case of normal walk. But that is not the case if the subject walks abnormally in which left and right joints produce different patterns. We hence propose a measure that allows quantifying this non symmetry value, we called it asymmetry measure. This value will be updated every T_{id} seconds, which is experimentally equivalent to one gait cycle duration because we are considering the symmetry between left and right leg movements within a cycle of walk. Every T_{id} seconds, since the left and right steps are out of phase, we firstly need to align the left and right curves measured at left and right joints (knee, elbow, ankle, etc.) to minimize the mean error δ which is the asymmetry measure as follows:

$$\delta = \min_{0 \leq \Delta \leq T_{id}} \sum_{i=t_0}^{t_0+T_{id}} \frac{R_a(i+\Delta) - L_a(i)}{T_{id}} \tag{5.1}$$

where R_a represents the right curve, L_a represents the left curve and t_0 is the current time. Keeping L_a stationary, we start moving R_a along the timeline, using Equation 5.1 to obtain the minimum error δ which is the asymmetry measure of the gait for this period.

In Section 3.3.5, we had the $< starting, ending >$ time information of each cycle for the whole gait duration. Taking this into account, for a particular cycle $cycle_i$, Equation 5.1 can be resolved by setting $T_{id} = ending_{cycle_i} - starting_{cycle_i}$, and $t_0 = starting_{cycle_i}$. These cues help us overcome the difficulty of the variation of gait cycle duration due to the walking speed. Fig 5.4 illustrates this minimization process for the knee joint.

5.2 Experimentation

The aim of our test is to perform the asymmetry assessment on two different subjects using our system versus Kinect. We chose JVC camcorders (model GZ-HD6U) as our two monocular cameras. We also placed the Kinect in front of the treadmill (distance about 2 meters) to acquire the 3D pose, see Fig 5.2. The frame synchronization of our cameras with Kinect was done by turning off and on the lights (repeat twice i.e., $N_{switch} = 2$) to produce a sudden image brightness drop and rise. The videos from the Kinect (RGB data) and left/right cameras were then aligned temporally according to the image mean brightness changes (see Section 3.4). The ground-truth was obtained by tracking red stickers put on subject's joints as shown in Fig 3.5(A). Each subject was asked to walk three different manners: normal walk, shorter-stride-length of the left leg and shorter-stride-length of the right leg. Once the pose was estimated from our system, from the Kinect SDK [6] and from markers, we started calculating the gait characteristics, explained in Section 3.1, on these three skeletons. In Figs 5.5 to 5.7, we present the joint angle curves at knee, elbow and ankle respectively, acquired from the left and right parts of the three skeletons described above in both normal and abnormal conditions. Our camcorder method outperformed Kinect at knee and ankle while was competitive with the Kinect at the elbow joint.

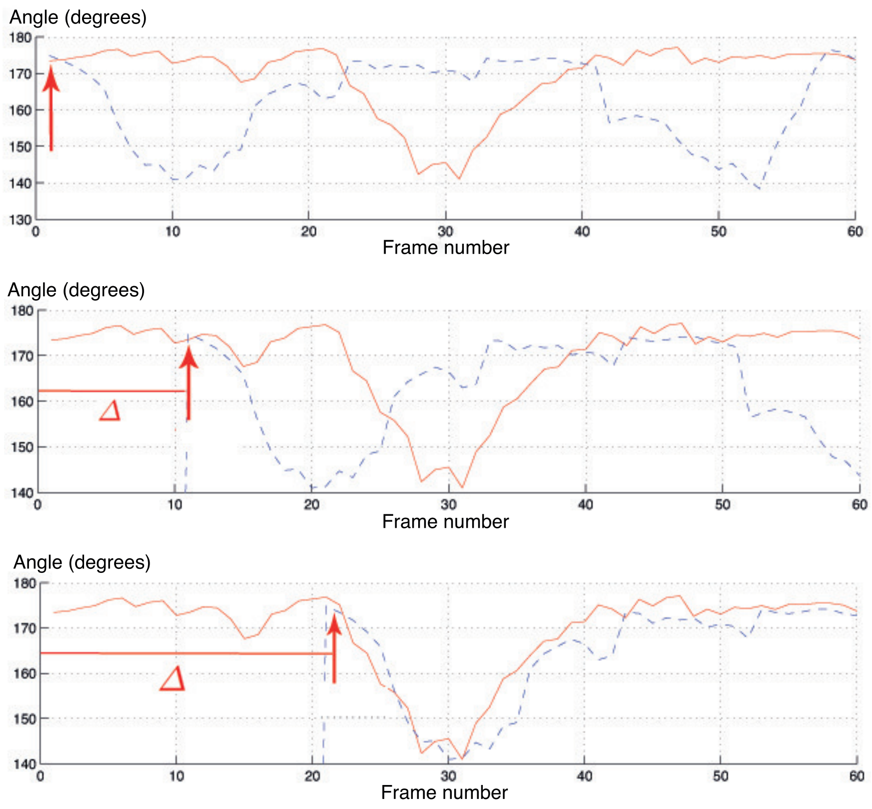


Figure 5.4: From top to bottom, we shift the right knee angle curve (dash blue curve) while the left knee (red curve) remains stationary by increasing Δ. The bottom graph gives the lowest error δ used as asymmetry measure.

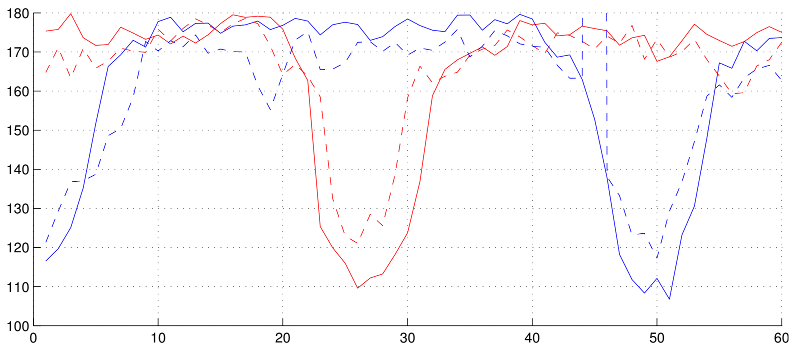
In our experimentation on both subjects, the asymmetry measures on knee, elbow, ankle from normal cases were significantly smaller than that from the abnormal cases (see

Table 5.I, Table 5.II (Camcorders)). Our asymmetric quantizations were consistently close to that from ground-truth (shown in Table 5.I, Table 5.II (Markers)) at all three joints while the Kinect failed at knee and ankle (shown in Table 5.I, Table 5.II (Kinect)) since there were no obvious difference between normal and abnormal gaits. This result leads us to an interesting asymmetry measure, see Section 5.1.2, when classifying normal and abnormal gaits. This measure is deduced relatively to the gait characteristics of each individual, and thus allows an easy intra-subject assessment of gait. In Table 5.III, we measured the difference (in centimetres) between the left and right stride lengths. With results close to ground-truth's and clearer difference between symmetric vs. asymmetric walks (compared to the Kinect's), we demonstrated that our system is capable of giving another reliable factor (stride length) for gait analysis experts.

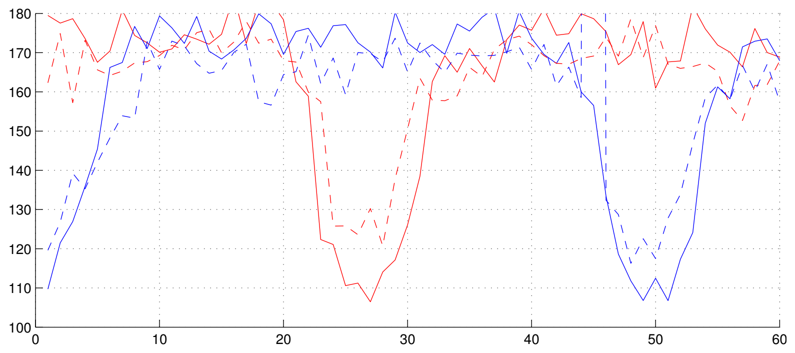
5.3 Discussion

We plan to further test our method on a sufficiently large number of subjects so that we can more precisely define which gait is normal with the help of appropriate angle thresholds at knee, elbow and ankle joints. In this study, a treadmill was used in order to have a one set of relatively constant position of the subject relative to the camcorders/Kinect to facilitate measurements of several gait cycles. Thus, it restricts the use of the method to treadmill walking compared to other approaches based on a walkway. One possible solution for this case is that we will divide the whole walkway into small segments and use our left and right cameras to observe subject in each segment.

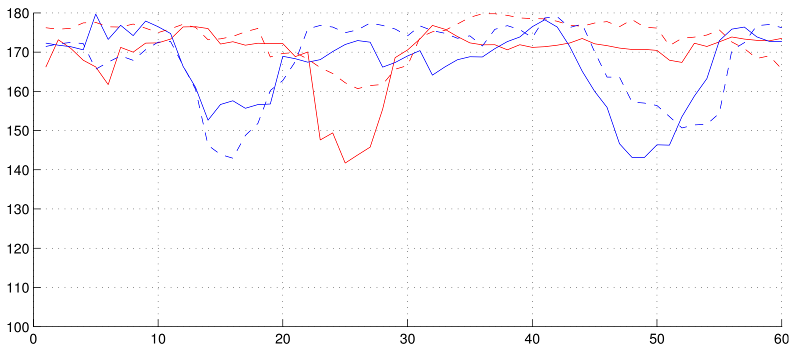
Notice that a full 3D skeleton reconstruction could also be possible with the camcorder system at the cost of a more complex strategy based on known limb length, camera calibration and a pose estimation methodology such as in [48] (see also annexes A and B).



Markers

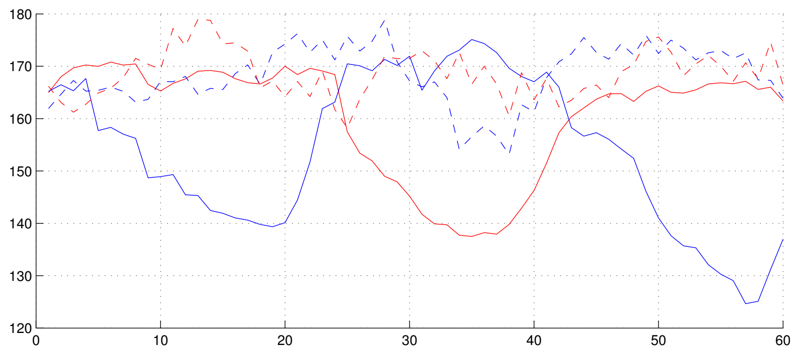


Camcorders

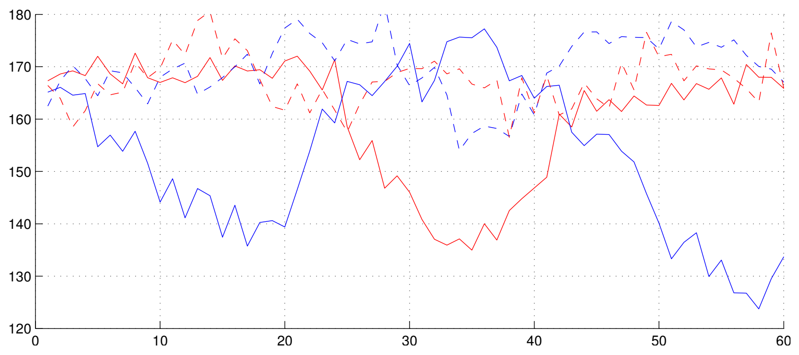


Kinect

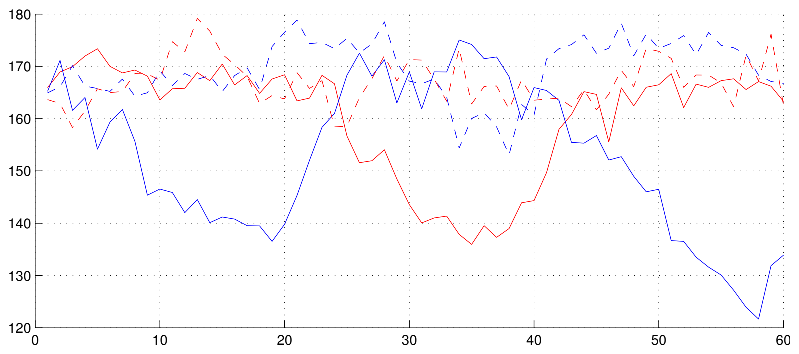
Figure 5.5: Angle changes at knee. Solid and dash red lines represent the angle changes at the left knee joint for normal and abnormal cases respectively within T_{id} seconds. Solid and dash blue lines represent the angle changes at the right knee joint for normal and abnormal cases respectively within T_{id} seconds. In the case of abnormal walk, the dash lines demonstrate clearly the broken symmetry between left and right parts.



Markers

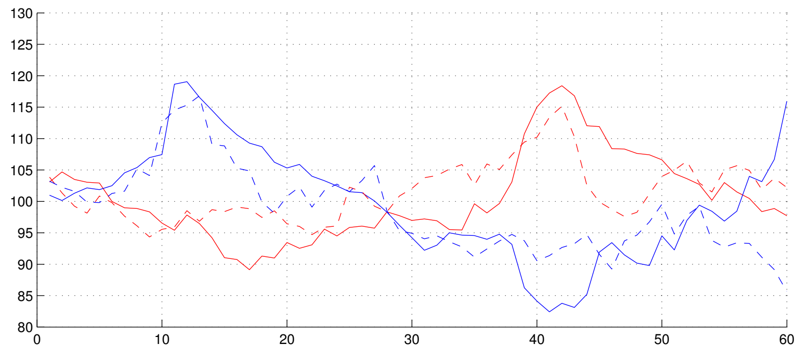


Camcorders

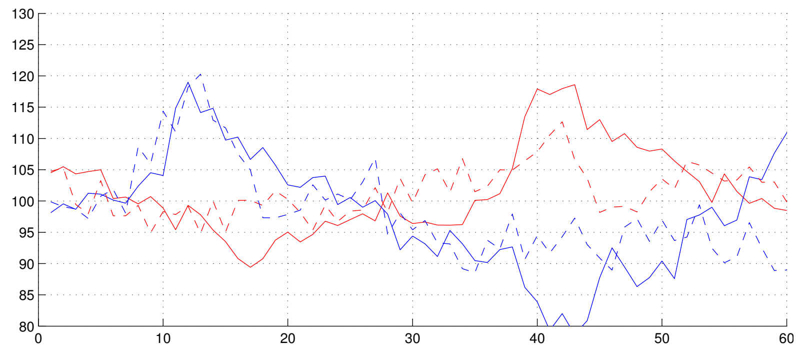


Kinect

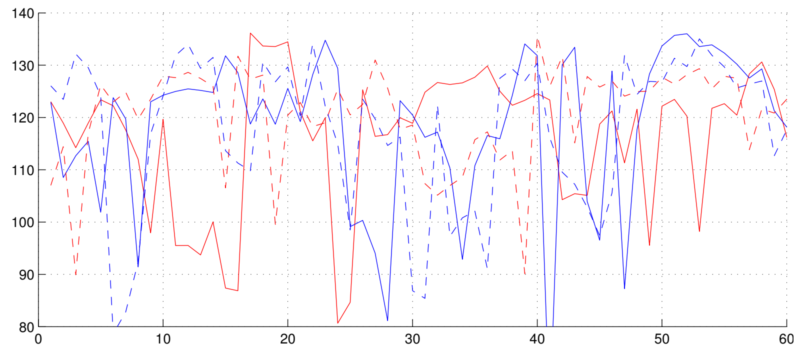
Figure 5.6: Angle changes at elbow, same color configuration as Fig 5.5.



Markers



Camcorders



Kinect

Figure 5.7: Angle changes at ankle, same color configuration as Fig 5.5.

Table 5.I: Asymmetry measures (degrees) compared with the markers (ground-truth) by the camcorders and the Kinect on subject 1. These results were calculated over 10 cycles. (A) Normal walk. (B) Left asymmetry walk. (C) Right asymmetry walk.

	Markers			Camcorders			Kinect		
	Knee	Elbow	Ankle	Knee	Elbow	Ankle	Knee	Elbow	Ankle
A	7.64	6.29	5.29	8.97	7.95	7.33	10.58	7.43	26.76
B	14.34	26.06	9.76	13.86	24.76	13.30	11.94	27.44	27.76
C	12.51	28.99	11.05	13.48	29.50	12.62	13.61	27.70	22.76

Table 5.II: Asymmetry measures (degrees) compared with the markers (ground-truth) by the camcorders and the Kinect on subject 2. These results were calculated over 10 cycles. (A) Normal walk. (B) Left asymmetry walk. (C) Right asymmetry walk.

	Markers			Camcorders			Kinect		
	Knee	Elbow	Ankle	Knee	Elbow	Ankle	Knee	Elbow	Ankle
A	5.99	4.31	5.32	6.63	7.38	5.04	11.12	5.94	22.45
B	12.13	18.54	10.66	15.24	20.85	13.87	14.57	19.69	19.38
C	10.72	32.03	9.93	12.04	23.26	14.07	15.68	23.82	24.03

Table 5.III: The average of absolute difference (centimetre) between left and right stride length calculated by the markers (ground-truth), the camcorders and the Kinect on subjects 1 and subject 2. These results were calculated over 10 cycles. (A) Normal walk. (B) Left asymmetry walk. (C) Right asymmetry walk.

	Subject 1			Subject 2		
	Markers	Camcorders	Kinect	Markers	Camcorders	Kinect
A	4.46	9.12	24.89	3.61	8.57	19.23
B	23.78	19.95	22.90	21.50	26.38	18.85
C	27.05	23.55	31.94	25.23	19.68	22.21

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, we have aimed to create a low-cost, easy-to-use system for gait analysis that offers a promising tool for a wide range of applications in a clinical protocol. To that end, our main contribution is a novel gait analysis vision-based system consisting of two cameras placed on the left and right sides of a treadmill. Compared to other state-of-the-art vision-based human pose estimation methods that are not specifically designed for gait analysis, our context is more focused and tighter with some constraints on the subject's body to increase the final reconstruction accuracy. Those constraints are, firstly, different sock colors on each foot, and secondly, a short-sleeve T-shirt required to expose skin color. The first constraint allows us to correctly estimate the foot position and orientation which, according to our knowledge, has still been impossible for all vision-based systems without any special conditions like ours. Although our ankle computation gave much higher accuracy than that of the Kinect, the errors compared to ground-truth were still large to be considered a reliable information for gait analysis purpose. The second constraint gives us important features to estimate the arm in both 2D and 3D. Unlike traditional gait analysis systems that focus on lower limbs only (mainly on knees), our system considers also the upper limbs and the feet. The results were very promising since the 2.5D skeleton model (9 joints for each half body part) was reconstructed with good precision and stability compared to the Kinect. These results however have not been fully validated by an expert yet. This work needs to be done before concluding our results can be applied on real life application.

Another related contribution of this thesis presented in the annexes is to propose a novel approach that can directly infer 3D skeletons or 3D arm location from its 2D location using a monocular camera. This problem is hard and still is an active research field, we therefore decided to focus on our 2.5D skeletons for the main part of this thesis. In addition, our novel deformable torso estimation is an interesting alternative besides the traditional ellipse fitting technique to efficiently deal with the torso orientation regardless of the body shape. Furthermore, a chain of various computer vision techniques are introduced to create a complete workflow for a fully automatic, vision-based gait analysis system. As ultimately designed for gait analysis, we also have contributed a method to efficiently quantify gait asymmetry with a simple asymmetry index. For validation, we demonstrated the quality of our results by performing tests on two different subjects with both normal and abnormal walks with comparisons with the Kinect and a marker-based method used as ground truth. For a more careful validation, a larger number of subjects and more extensive tests need to be done in the future.

On the other hand, our system still has some shortcomings that need future improvements, such as reducing the number of manually set parameters, more robustly detecting arm skin (especially with significant foreshortening), more accurately locating the arm in 2D leading to more accurate 3D arm reconstruction. Since we made use of a sampling technique for almost each body part estimation independently, we believe that a new method to combine all those estimations into a whole could be very interesting to reduce the sub-level searching space. In terms of gait analysis applications, the asymmetry measure gives an efficient tool to distinguish a normal vs abnormal walk by performing the asymmetry measures calculation on one or more joints. More experimentations might be needed for parameterizing the normal walk (i.e. a threshold at each joint, which joints are worth calculating the index, etc.).

BIBLIOGRAPHY

[1] Jane Courtney, A.M. de Paor, *A Monocular Marker-Free Gait Measurement System*, IEEE Transactions on Neural systems and rehabilitation engineering, vol 18, No.4, August 2010.

[2] P.-Y. Chen and Y.-H. Lai, *A low-complexity interpolation method for deinterlacing* IEICE Trans. on Inf. Syst., vol. E90-D, no. 2, pp. 606-608, Feb. 2007.

[3] Thomas B.Moeslund, Adrian Hilton, Volker Kruger, *A survey of advances in vision-based human motion capture and analysis*, Computer Vision and Image Understanding, 104 (2006) 90-126.

[4] M. Andrilukam, S. Roth, and B. Schiele, *Pictorial structures revisited: People detection and articulated pose estimation* Computer Vision and Pattern Recognition (CVPR), pp. 1014-1021, 2009.

[5] M. Eichner, V. Ferrari, *We are family:Joint pose estimation of multiple persons* International Conference on Computer Vision (ICCV), pp. 228-242, 2010.

[6] V. Ferrari, M. Marin-Jiminez and A. Zisserman, *Progressive search space reduction for human pose estimation* Computer Vision and Pattern Recognition (CVPR), pp. 18, 2008.

[7] B. Sapp, C. Jordan, and B. Taskar, *Adaptive pose priors for pictorial structures* Computer Vision and Pattern Recognition, pp. 422-429, 2010.

[8] B. Sapp, A. Toshev, and B. Taskar, *Cascaded models for articulated pose estimation* European Conference on Computer Vision (ECCV), pp. 406-420, 2010.

[9] P. Felzenszwalb and D. Huttenlocher, *Efficient matching of pictorial structures* In Proc. Computer Vision and Pattern Recognition (CVPR), 2000.

[10] P. Felzenszwalb and D. Huttenlocher, *Pictorial structures for object recognition* In International Journal of Computer Vision, 61(1):55-79, 2005.

[11] M. Fischler and R. Elschlager., *The representation and matching of pictorial structures*, IEEE Trans. Computers, 22(1):67-92, Jan. 1973.

[12] N. Dalal and B. Triggs, *Histograms of oriented gradients for human detection*, Computer Vision and Pattern Recognition (CVPR), pp. 886-893, 2005.

[13] Y. Wang, D. Tran, and Z. Liao, *Learning hierarchical poselets for human parsing*, Computer Vision and Pattern Recognition (CVPR), pp. 1705-1712, 2011.

[14] S. Johnson and M. Everingham, *Clustered pose and nonlinear appearance models for human pose estimation*, BMVC, pp. 12.1-11, 2010.

[15] L. Bourdev and J. Malik, *Poselets: Body part detectors trained using 3D human pose annotations*, International Conference on Computer Vision (ICCV), pp. 1365-1372, 2009.

[16] S. Zuffi, O. Freifeld, M. J. Black, *From pictorial structures to deformable structures*, Computer Vision and Pattern Recognition (CVPR), June 2012.

[17] S. Zuffi, J. Romero, C. Schmid, M.J. Black, *Estimating Human Pose with Flowing Puppets*, International Conference on Computer Vision, 2013.

[18] M. Andriluka, S. Roth, B. Schiele, *Monocular 3D pose estimation and tracking by detection*, Computer Vision and Pattern Recognition (CVPR), 2010.

[19] M. Andriluka, S. Roth, B. Schiele, *People-tracking-by-detection and people-detection-by-tracking*, Computer Vision and Pattern Recognition (CVPR), 2008.

[20] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, *Efficient regression of general activity human poses from depth images*, Computer Vision and Pattern Recognition (CVPR), 2011.

[21] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, *Real-time human pose recognition in parts from a single depth image*, Computer Vision and Pattern Recognition (CVPR), 2011.

[22] M. Sun, P. Kohli, J. Shotton, *Conditional regression forests for human pose estimation*, Computer Vision and Pattern Recognition (CVPR), 2012.

[23] J. Gall, B. Rosenhahn, T. Brox, H-P. Seidel, *Optimization and Filtering for Human Motion Capture*, Int J. Computer Vision, 87(1):75-92, 2010.

[24] J. Gall, J. Potthoff, C. Schnoerr, B. Rosenhahn, H.P. Seidel, *Interacting and annealing particle filters: Mathematics and a recipe for applications*, J. of Mathematical Imaging and Vision, 28(1), 118 (2007).

[25] B. Daubney, X. Xie, *Estimating 3D pose via stochastic search and expectation maximization*, Articulated Motion and Deformable Objects (AMDO), 2010.

[26] C. C. Hsiel, D. H. Liou, W. R. Lai, *Enhanced Face-Based Adaptive Skin Color Model*, Journal of Applied Science and Engineering, Vol. 15, No. 2, pp. 167E176 (2012).

[27] G. Papari, N. Petkov, P. Campisi, *Artistic Edge and Corner Enhancing Smoothing*, Journal IEEE Transactions on Image Processing, Vol. 16 Issue 10, pp. 2449-2462, October 2007

[28] S. Piérard, R. Phan-Ba, V. Delvaux, P. Maquet, and M. Van Droogenbroeck. *GAIMS: a powerful gait analysis system satisfying the constraints of clinical routine. Congress of the European Committee for Treatment and Research in Multiple Sclerosis (ECTRIMS)*, Copenhagen, Denmark, October 2013.

[29] Dantone M., Gall J., Leistner C., and van Gool L. *Human Pose Estimation using Body Parts Dependent Joint Regressors.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[30] E. Simo-Serra, A. Ramisa, G. Alenya, C. Torras, F. Moreno-Noguer, *Single image 3D human pose estimation from noisy observations.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] R. Okada, S. Soatto, *Relevant feature selection for human pose estimation and localization in cluttered images.* In *European Conference on Computer Vision*, 2008.

[32] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, P. Torr, *Randomized trees for human pose detection.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[33] C. Sminchisescu, B. Triggs, *Estimating articulated human motion with covariance scaled sampling.* In Int. J. of Robotics Research, 22(6), 371-391, 2003.

[34] M. Bray, E. Koller-Meier, L.V. Gool, *Smart particle filtering for high dimensional tracking.* In Computer Vision Image Underst, 106(1), 116-129, 2007.

[35] J. Deutscher, I. Reid, *Articulated body motion capture by stochastic search.* Int. J. of Computer Vision, 61(2), 185-205, 2005.

[36] N. Hansen, *The CMA evolution strategy: a comparing review.* In *Towards a new evolutionary computation. Adv. on estimation of distribution alg*, pp 75-102. Springer, 2006.

[37] B. Rosenhahn, T. Brox, H.P. Seidel, *Scaled motion dynamics for markerless motion capture.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[38] R. Urtasun, D.J. Fleet, P. Fua, *3d people tracking with gaussian process dynamical models.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[39] K. Moon, V. Pavlovic, *Impact of dynamics on sub space embedding and tracking of sequences.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[40] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, *Real time motion capture using a single time-of-flight camera*, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[41] M. Siddiqui and G. Medioni, *Human pose estimation from a single view point, real-time range sensor*, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[42] http://www.microsoft.com/en-ca/download/details.aspx?id=40278

[43] Hoang Anh Nguyen, E. Auvinet, M. Mignotte, J.A de Guise, J. Meunier, *Analyzing gait pathologies using a depth camera*, in *Engineering in Medicine and Biology Society (EMBC)*, 2012.

[44] P. Shrestha, H. Weda, M. Barbieri and D. Sekulovski, *Synchronization of multiple video recordings based on still camera flashes*, Proceedings of the 14th annual ACM international conference on Multimedia Pages 137-140, 2006.

[45] L. Bo, C. Sminchisescu, A. Kanaujia, and D. Metaxas, *Fast algorithms for large scale conditional 3D prediction*, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[46] L. Bo and C. Sminchisescu, *Structured output-associative regression*, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[47] Z. Lu, M. A. Carreira-Perpinan, and C. Sminchisescu, *People tracking with laplacian eigenmaps latent variable models*, In *Neural Information Processing Systems*, 2009.

[48] Hoang Anh Nguyen, Jean Meunier, *Reconstructing 3D human poses from monocular image*, International Conference on Information Science, Signal Processing and their Applications, 2012, Canada.

[49] F. Jean, R. Bergevin, A. Branzan-Albu, *Human gait characteristics from unconstrained walks and viewpoints*, In Proceedings of the International Conference on Computer Vision Workshops, (Barcelona, Spain), pp. 1883-1888, November 6-13 2011.

[50] E. Ceriez et R. Motmans, *Anthropometry table*, Ergonomie RC, Leuven, Belgium, 2005.

[51] Hoang Anh Nguyen and Jean Meunier, *Video-based analysis of gait with side views*, International Conference on Image Processing Theory, Tools and Applications, Paris, France, 2014.

[52] Hoang Anh Nguyen and Jean Meunier, *Gait analysis from video: Camcorders vs. Kinect*, International Conference on Image Analysis and Recognition, Vilamoura, Algarve, Portugal, 2014.

[53] C. Hodt-Billin, *Measures of symmetry in gait: Methodological principles and clinical choices*, PhD Thesis, 2014.

[54] C. Hodt-Billington, J.L. Helbostad, R. Moe-Nilssen, *Should trunk movement or footfall parameters quantify gait asymmetry in chronic stroke patients*, Gait Posture 27, 552Ð55810.1016/j.gaitpost.2007.07.015, 2008.

[55] K.K. Patterson, W.H. Gage, D. Brooks, S.E. Black, W.E. McIlroy, *Evaluation of gait symmetry after stroke: A comparison of current methods and recommendations for standardization*, Gait Posture 2010b; 31(2): 241-6, 2010.

[56] M.S. Madsen, M.A. Ritter, H.H Morris, J.B. Meding, M.E. Berend, P.M Faris, V.G Vardaxis, *The effect of total hip arthroplasty surgical approach on gait*, Journal Orthop Res. 2004; 22(1): 44-50.

[57] C.K. Balasubramanian, M.G Bowden, R.R Neptune, S.A Kautz, *Relationship between step length asymmetry and walking performance in subjects with chronic hemiparesis*, Arch Physic Medical Rehabilation. 2007; 88(1): 43-9.

[58] C. Hodt-Billington, M. Almkvist, R. Moe-Nilssen, *Reliability and classification ablity of gait asymmetry parameters in subjects with hemiplegia*, Parkinsonism & Related Disorders 2010; 16(Suppl 1): S67.

[59] S. Hesse, C. Werner, H. Seibel, F.S. Von, E.M Kappel, S. Kirker, M. Kading, *Treadmill training with partial body-weight support after total hip arthroplasty: a randomized controlled trial*, Arch Phys Med Rehabil. 2003; 84(12): 1767-73.

[60] L. Vogt, W. Banzer, I. Bayer, D. Schmidtbleicher, F. Kerschbaumer, *Overground and walkway ambulation with unilateral hip osteoarthritis: comparison of step length asymmetries and reproducibility of treadmill mounted force plate readings*, Physiother Theory Pract. 2006; 22(2): 73-82.

[61] M. Roerdink, P.J. Beek, *Understanding inconsistent step-length asymmetries across hemiplegic stroke patients: impairments and compensatory gait*, Neurorehabil Neural Repair 2011; 25(3): 253-8.

[62] M. Plotnik, N. Giladi, J.M. Hausdorff, *A new measure for quantifying the bilateral coordination of human gait: effects of aging and Parkinson's disease*, Exp Brain Res. 2007; 181(4): 561-70.

[63] R.A. Zifchock, I. Davis, J. Higginson, T. Royer, *The symmetry angle: a novel, robust method of quantifying asymmetry*, Gait Posture 2008; 27(4): 622-7.

[64] M.W. Whittle, *Gait analysis an introduction*, Elsevier, New York 2007.

[65] http://www.gla.ac.uk/t4/ fbls/files/fab/tutorial/anatomy/hfgait.html

[66] https://home.comcast.net/ pegglestoncbsd/human_body_orientation.htm

[67] http://www-personal.umich.edu/ hamms/portfolio/motioncapture/index.html

[68] M. Eichner, M. Marin-Jimenez, A. Zisserman, *2D Articulated Human Pose Estimation and Retrieval in (Almost) Unconstrained Still Images*, Intenational Journal Computer Vision, 99:190-214, 2012.

[69] A. Fitzgibbon, M. Pilu, R. B. Fisher, *Direct Least Square Fitting of Ellipses*, Pattern analysis and machine intelligence, vol. 21, no. 5, May 1999

[70] J. Courtney and A. M. de Paor, *Direct Least Square Ellipses Fitting*, Proceeding of the Seventh IASTED International Conference Computer Graphics and Imaging, USA 2004.

[71] M. Kuwahara. K. Hachimura, S. Eiho, and M. Kinoshita, *Processing of RI-angiocardiographic images*, Digital Processing of Biomedical Images, pp.187-202, 1976.

[72] X. K. Wei, J. Chai, *Modeling 3DHuman Poses from Uncalibrated Monocular Images*, Proceeding of IEEE Conference on Computer Vision and Pattern Recognition, 2009.

[73] C. Barron and I. A. Kakadiaris, *Estimating anthropometry and pose form a single image*, Proceeding of IEEE Conference on Computer Vision and Pattern Recognition, 2000.

[74] C. J. Taylor, *Reconstruction of articulated objects from point correspondences in a single uncalibrated image*, In Computer Vision and Image Understanding, 80(3): 349-363, 2000.

[75] V. Parameswaran and R. Chellappa, *View Independent Human Body Pose Estimation from a Single Perspective Image*, In Proceeding of IEEE Confer- ence on Computer Vision and Pattern Recognition, 2: 16-22, 2004.

[76] R. Rosales and S. Sclaroff, *Inferring body pose without tracking body part*, In Proceeding of IEEE Conference on Computer Vision and Pattern Recognition, 2: 506-511, 2000.

[77] M. W. Lee and I. Cohen, *Proposal Maps Driven MCMC for Estimating human body pose in static images*, In Proceeding of IEEE Conference on Com- puter Vision and Pattern Recognition, 2: 334- 341, 2004.

APPENDIX A

Reconstructing 3D human poses from monocular image. Hoang Anh Nguyen, Jean Meunier. *International Conference on Information Science, Signal Processing and their Applications*, 2012, Canada.

A.1 Context

At the time that we wrote this article, our goal was to resolve a hard computer vision problem: how to correctly locate the 3D position of a known 2D skeleton joints using only a monocular camera. Since this is an ill-posed problem, we took into account the gait analysis context and applied some acceptable constraints on the limb movements. With manual 2D skeleton inputs, our 3D skeleton outputs were promising on simple poses. In our current system, we exploit the side view of the subject to acquire good 2D skeleton model for further gait analysis. As in gait analysis context, from this view, there is no need for fully reconstructing the whole 3D skeleton since the lower limbs do not suffer significant foreshortening. Since the 2D skeleton model alone is enough for a good gait analysis, we therefore did not further apply this approach in our system. The following is an extended version of the original paper.

A.2 Abstract

This project aims to reconstruct 3D human poses from a small number of 2D point correspondences obtained from a calibrated monocular image. This problem is not trivial because the 2D image constraints are often not enough to determine 3D poses of an articulated objects. Due to the variation of human poses, we made some assumptions for the poses to be estimated. The key idea of this project is to identify the location of two image points in 3D space using homography transformation, and then estimate the other points based on these with some constraints. With camera calibration, we can eliminate most 3D pose reconstruction ambiguities but not all of them. In order to choose the

best solution, we use additional constraints. We believe that our methodology will be effective for our future work on musculoskeletal assessment from 3D poses of patients walking or running on a treadmill.

A.3 Introduction

Lately human motion capture, or pose estimation, has received much attention due to its large number of potential applications such as movies, surveillance, video games, virtual reality environments... This project presents an algorithm to reconstruct a 3D human pose from a calibrated image. The analysis of [72] shows that, under perspective projection, we need at least five images to accurately reconstruct 3D human poses. Because the input of our project is just one image, we have to define some assumptions and constraints to eliminate pose ambiguities. Our work builds on the success of previous works in reconstructing 3D articulated objects from a single image [73], [74], [75]. Because modelling articulated 3D objects from a single 2D image is an "ill-posed" problem, many previous approaches often rely on known skeleton sizes or strong anthropometric prior to reduce reconstruction ambiguity. For example, Taylor [74] assumed a known skeleton size and presented an analytical solution to recover 3D orientation of the bone segments up to an undetermined weak perspective camera scale. Baron and Kakadiaris [73] extended the idea to estimate both anthropometric parameters and human body pose from a single image. They formulated the problem as a nonlinear optimization problem and imposed a multivariate Gaussian prior on bone lengths to constrain the solution space. Parameswram *et al.* [75] solved the same problem with known skeleton sizes, accounting for projective foreshortening effects of a simplified skeleton model. Our approach is a little different from previous work because we use the calibrated image to reconstruct 3D human poses with some constraints between the bones of the estimated skeleton. Another approach is to use data-driven techniques to reduce reconstruction ambiguity. Previous work in this direction either learns the mapping between 2D images features (e.g., silhouettes) and 3D poses [76], or constructs pose priors to constrain the solution space [77]. This approach, however, has not been demonstrated that it can accurately reconstruct 3D poses with unknown skeleton sizes and camera parameters. Another limitation of the data-driven approach is that it can only model poses that are similar to the training data. Furthermore, the acquisition of human pose data requires lots of effort for generalization. Our approach does not have this limitation, and can model human poses from a monocular image. We need the skeleton size and calibrated camera to get all the parameters needed for space transformation. To estimate the 3D model, we compute the initial point with a homography transformation, which is a very suitable technique to do that. Then, the whole process consists of three stages: we first build the leg and pelvic, then build the torso and head based on the location of pelvic. The final step is to estimate the shoulder and arm locations. The quality of the reconstruction results produced by our system depends on the accuracy of 2D joint locations specified by the user. We also evaluate the robustness of the algorithm in term of different input noise.

A.4 Overview

Our algorithm can be summarized as follows. To estimate the 3D model, first, we compute two initial points corresponding to the feet with a homography transformation. Then, knowing the skeleton size and the camera calibration, we reconstruct the legs, pelvic, torso, head, shoulders and arms in that order. With this technique, we get a list of possible solutions for human pose that are then filtered with some constraints to obtain the final 3D model. These steps are now described in detail in the next sections.

A.5 Full perspective technique

This section presents a technique which allows estimating a new 3D world point based on the known previous 3D world point and the known distance ΔL between them. This distance can be obtained from anthropometric data (motion capture databases or literature). Two sets of lengths are illustrated in Table A.I. The first set of relative distances between human joints is derived from a motion capture database [Table A.I, central]. The second set is more general and follows the studies performed by Leonardo Da Vinci and Michenlangelo on the human body [Table A.I, right]. In this project, we used the first set based on the motion capture database.

Segments	Relative Length (MC) [cm]	Relative Length (L) [unit]
Height	175	8i
Lower arm	35	2i
Upper arm	25	$1\frac{1}{2}$i
Neck-Head	25	$1\frac{1}{4}$i
Shoulder Girdle	44	2i
Torso	53	$2\frac{1}{2}$i
Pelvic Girdle	30	$1\frac{1}{2}$i
Upper leg	46	2i
Lower leg	52	2i
Foot	22	1i

Table A.I: Two different sets of relative lengths of segments used in the computation of human model. MC = Motion Capture. L = Literature. The coefficient i has been added in the last column to consider variation of the human size.

Assuming that radial distortion can be neglected, the relation between a point (x,y) in the image reference and a point (X^c,Y^c,Z^c) in the camera reference is:

$$x = \frac{f}{p_x}\frac{X^c}{Z^c} \tag{A.1}$$

$$y = \frac{f}{p_y}\frac{Y^c}{Z^c} \tag{A.2}$$

where the intrinsic parameters are: focal length f, horizontal and vertical effective pixel size (p_x, p_y) and image centre coordinates (o_x, o_y).

Suppose we have a previous point $[X^w_{i-1}, Y^w_{i-1}, Z^w_{i-1}]$ and we want to estimate the new point $[X^w_i, Y^w_i, Z^w_i]$, so first of all, we have to transform the previous world point into camera coordinates $[X^c_{i-1}, Y^c_{i-1}, Z^c_{i-1}]$ because all of the calculations below are computed in camera space. The distance ΔL between these two points in camera coordinates is the same as in world coordinates. Calculating this ΔL with A.1 and A.2 gives a quadratic equation with only one unknown variable Z as:

$$Z^2(1+\frac{x^2}{f^2}+\frac{y^2}{f^2})-2Z(\frac{xX^c_{i-1}}{f}+\frac{yY^c_{i-1}}{f}+Z^c_{i-1})$$
$$+(X^c_{i-1})^2+(Y^c_{i-1})^2+(Z^c_{i-1})^2-\Delta L^2=0 \quad (A.3)$$

Resolving A.3 gives two possible values for Z, we therefore compute X and Y from A.1 and A.2 with each value of Z so that we get two points $[X^c_1,Y^c_1,Z^c_1]$ and $[X^c_2,Y^c_2,Z^c_2]$:

$$X^c_{i'}=\frac{(x-o_x)p_xZ^c_{i'}}{f} \quad (A.4)$$
$$Y^c_{i'}=\frac{(y-o_y)p_yZ^c_{i'}}{f} \quad (A.5)$$

where i' =1,2.

Finally, we can obtain the new 3D world points by transforming those points into world coordinates if the extrinsic matrix is known.

$$\begin{pmatrix} X^w_i \\ Y^w_i \\ Z^w_i \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} X^c_i \\ Y^c_i \\ Z^c_i \\ 1 \end{pmatrix}$$

A.6 Skeleton reconstruction algorithm

Because reconstructing 3D human poses from only one calibrated image is an "ill-posed" problem, making some assumptions and constraints are needed to eliminate the ambiguities. In our work, we divide the skeleton into three parts according to the different estimation method between them, each part is built separately but not independently of each others.

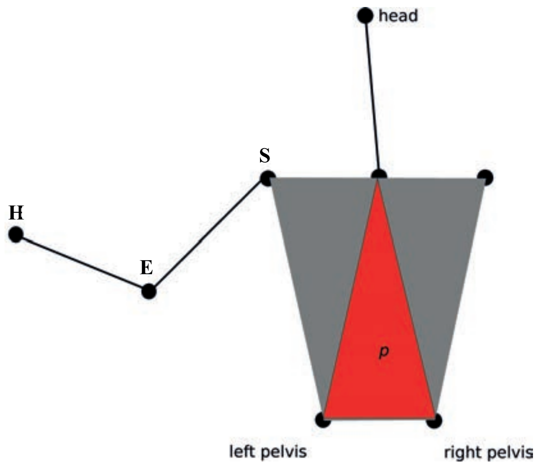


Figure A.1: The 3D model used to estimate the arm. Plane p, represented by the red triangle, is created with 3 points: neck, left and right hips. S, E, H indicate the left shoulder, left elbow, left hand respectfully.

A.6.1 Estimating foot, knee and pelvic points

The first assumption is that both feet must be on the ground, and then both foot positions can be successfully located by homography transformation. From each foot position found, thanks to the full perspective technique (section A.5), we have two possible values of knee, and four possible positions of pelvic.

We begin to filter the solutions with a simple constraint for the knee. This constraint only works if the human pose is photographed from a frontal view. This constraint is acceptable for our future investigation of musculoskeletal disorders with patients walking or running on a treadmill with a frontal camera. Other constraints could be developed for a side view. Among the two possible candidates, we choose the one that is closer to the camera than the other. With this constraint, we only have two possible values for pelvic, instead of four, and thus cut off half of the solution space.

In order to choose the pelvic, we project the two positions into the X^wY^w plane, which can be easily obtained by setting their $Z^w = 0$, and simply choose the one with shortest distance to the foot. We do the same process for both left and right legs, finally, we connect left pelvic and right pelvic points together and the middle of this line will be the point from which we reconstruct the torso and head.

A.6.2 Estimating torso and head

During the experimental period, we observed that the angle between the torso and the head is usually small and the well-estimated torso is mostly the one that points toward the camera. Based on these observations, we only keep the most suitable solution and obtain the positions for both torso and head. This result will be used as initial points for reconstructing the arms in the next step.

A.6.3 Estimating shoulders, elbows and hands

This estimation is a real challenge due to the wide range of movements of the arms. It is not easy to define a set of effective constraints to eliminate the ambiguities. In this project, we have to predict the human poses with just one image, so we created four constraints which are not always true in reality but satisfactorily for our purpose.

To do this, first, we create a plane by using three successive points: left pelvic, right pelvic and torso, see Fig. A.1 for how to create this plane. Then, we choose the shoulder position which has the lowest distance to this plane. This constraint is not correct for all cases but the final result is usually acceptable. By doing this, we have reduced half of the solution space which now has only two solutions for each elbow and four solutions for each hand. Next we apply these simple heuristics to eliminate unreasonable solutions.

First, the distance l_{hs} between hand (H) and shoulder (S) must be inferior to the total distance between hand-to-elbow (E) l_{he} and elbow-to-shoulder l_{es}.

$$l_{hs} < l_{es} + l_{he}$$

In fact, this constraint was not much effective because the solutions produced by the full perspective technique usually satisfy this constraint.

Second, we limit l_{hs} to be superior to half of l_{es} which means the minimum angle between SE and HE can not be too small. This constraint is a little bit more helpful than the previous one.

$$l_{hs} > \frac{1}{2} l_{es}$$

The next constraint is for hand behind the head. This constraint is very effective to eliminate some illogical solutions. We limit the range of the hand by eliminating all the hand positions having a distance to plane (p) superior to the predefined length of the upper arm. It still can be wrong in some special cases but in general, this constraint is valid.

The final constraint checks the relative position between H, S and E to ensure that the distance from E to the plane p must not be larger than the distance from H to p.

$$d(E \to p) < d(H \to p)$$

Finally, after filtering all the solutions with these constraints, the selected solutions are all reasonable enough to be chosen. To choose only one of them, first, we project all the points into the XY plane, which represents the view from the top. Then, for each solution, we compute the sum of the distance from shoulder to elbow and the distance from elbow to hand. The chosen one will be the one having this value minimum. This intuitive method produced a good final solution but not always the best one.

A.7 Result

To test our algorithm in real situations, we have built a set of images with different poses and view directions. This set contains 10 images of both frontal views and side views. As expected, the results from the frontal view were more accurate than those from the side view. Fig. A.2 shows some poses we have tested.

All the skeleton reconstructions are visually accurate, except for C and F with the wrong position for the right elbow and right shoulder respectively. In C, the result is wrong due to the final distance constraint for elbow. In F, when the human pose is seen from a side view, the right shoulder position is wrongly estimated.

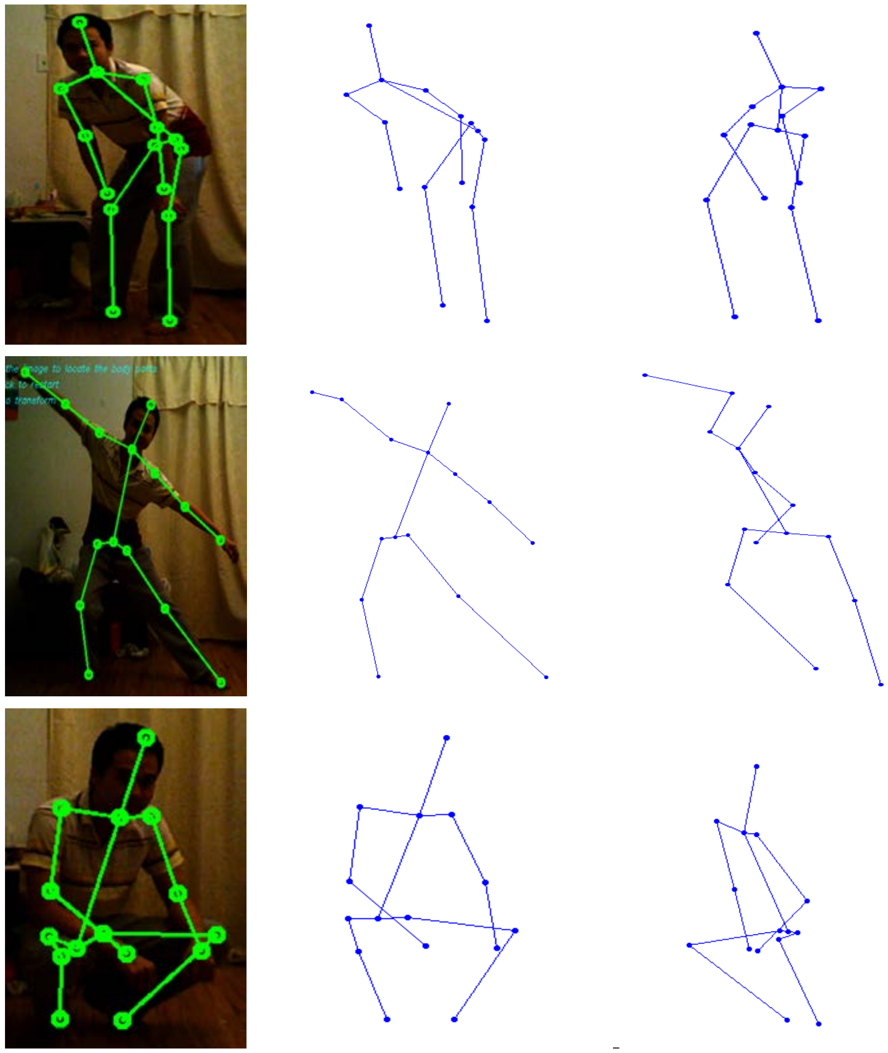


Figure A.2: The results obtained by applying the reconstruction algorithm to calibrated images. Left column contains the original images with different poses. The middle and the last columns show the estimation results with different view directions.

APPENDIX B

B.1 Context

In Section 3.3.2 and 3.3.3, we introduced our methodology for fully reconstructing the arm (including upper and lower arms) in 2D. We also mentioned the effect of the foreshortening issue on the arm estimation: when swinging the arm, the closer the hand is to the other half side, the clearer the foreshortening is. It results in changes of limbs length (i.e. shorter), so we had to apply the Monte Carlo sampling technique to efficiently solve the problem. This issue, however, brings another obstacle: the angle at the elbow calculated in 2D space is not the same as that in 3D space if significant foreshortening happens. This error did not affect much our elbow angle results compared to the Kinect's (see Table 4.II) due to negligible foreshortening for all our subjects. This is understandable since the walk does not require strong movement of limbs like running, or jogging. However, for a broader range of applications, we explain below how we can deal with foreshortening when it occurs.

B.2 Methodology

The foreshortening of upper arm was already numerically measured as ρ_{ua} in Section 3.3.3. In order to facilitate the use of ρ_{ua} in next step, we build a mapping function $F_{\rho_{ua}}$ that maps one real value of ρ_{ua} into degrees as follows

$$F_{\rho_{ua}} = \mathbb{R}_{(0,1]} \mapsto \wp_{[90^\circ,0^\circ]} \tag{B.1}$$

This mapping is based on one characteristic of ρ_{ua} that is: if $\rho_{ua} \approx 0$ means the maximum foreshortening in which the upper arm is perpendicular to the sagittal plane of the body (90°). If $\rho_{ua} = 1$ means no foreshortening, the upper am joint length observed in the image space is its maximum length, i.e. the upper arm is parallel to the sagittal plane (0°).

In order to resolve this issue and improve the accuracy of our elbow angle measurement, we propose a technique to reconstruct the arm fully in 3D. The final elbow angle is then calculated directly from 3D space. This method is inspired from [48] in which we made use of a full perspective technique and reconstructed the whole 3D skeleton model starting from prior known 3D positions of feet. With a very similar manner, we first try to locate the 3D shoulder position based on the 3D walk trajectory obtained in Section 3.3.5, then apply the full perspective technique (see Section A.5) to locate the elbow and hand. This process starts right after the 3D foot estimation, therefore all the 2D skeleton data for all frames are available.

B.2.1 3D shoulder

Our first objective is to locate the shoulder in the 3D camera space given its 2D position in image and some information about foot trajectory in the 3D world space L_{trajec} (see Section 3.3.5). Since there is no direct method for this problem, the particle filter is a suitable optimization technique to iteratively locate the correct 3D shoulder since we are able to estimate a good initial location for it. To reduce the search range as well as to have the best result, we need to have a good initialization of where the potential shoulder should be and then generate particles around that point. As shown in Fig. B.1, this can be done by using a kinematic constraint that, in order to keep the balance, the projection of shoulder (blue circle) onto the ground (green circle) should (1) fall within the 2D foot trajectory range (image projection of 3D foot trajectory in Section 3.3.5) (red arrows), and (2) be close to one of the foot position inside that trajectory in 3D space.

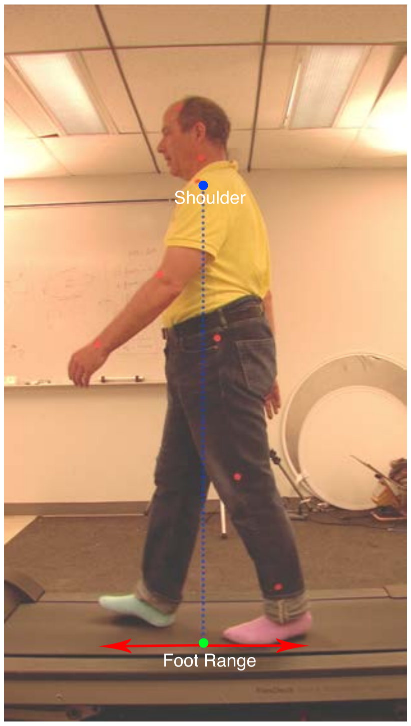


Figure B.1: The projection of shoulder (blue circle) onto the ground (green circle) should lie inside the foot range which is indicated by the 2-head arrow.

Taking this information into account, we can locate the virtual foot p_{vfoot} in image space by projecting the 2D shoulder into the 2D version of L_{trajec} (L^{2d}_{trajec}) regardless of the current position of the foot. After finding p_{vfoot}, its corresponding 3D position P_{vfoot} in world space is calculated based on the relative position of p_{vfoot} within L_{trajec}. Since the overall height of the person does not change much, a good 3D shoulder candidate $P^{init}_{shoulder}$ is deduced directly from P_{vfoot} and $H_{shoulder}$ in world space (anthropometry data [50]) using weak perspective technique. The next step is to apply the particle filter starting from this candidate by firstly creating the set of particles Υ following a 3D Gaussian distribution $(P^x_{vfoot}, P^y_{vfoot}, P^z_{vfoot} + H_{shoulder}, \sigma^x_{sh}, \sigma^y_{sh}, \sigma^z_{sh})$. Thank to the good shoulder candidate estimated above, we can keep the search range small in x and y axis (i.e., $\sigma^x_{sh}, \sigma^y_{sh}$ are small) and larger in z axis ($\sigma^z_{sh} = b\sigma^y_{sh}$ where $b = 2.5$ in our experimentation).

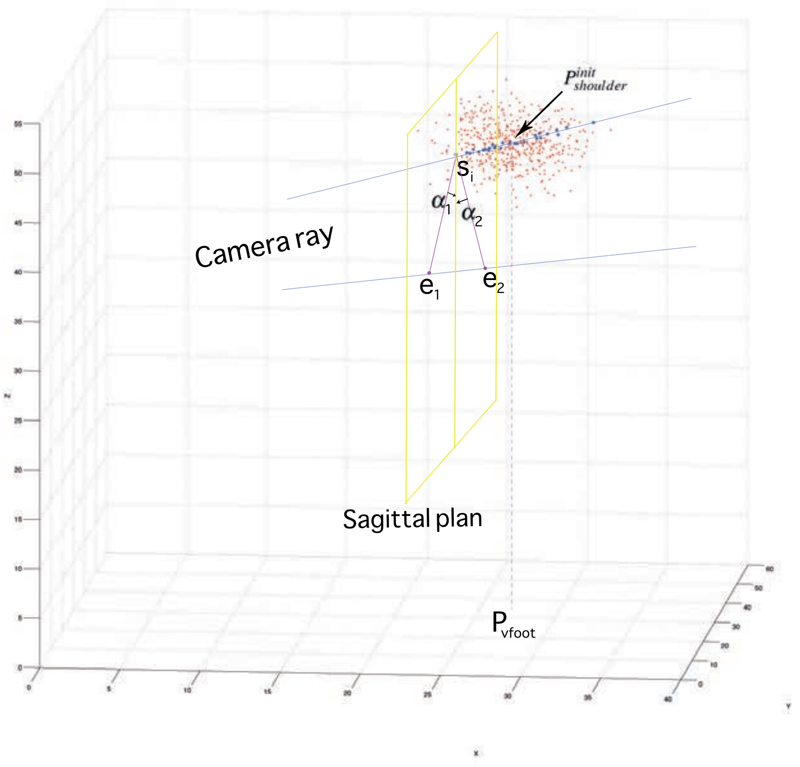


Figure B.2: 3D shoulder assessment overview: particles (red points) are generated around $P^{init}_{shoulder}$ which is deduced from P_{vfoot}. The first layer keeps only blue points which represent a ray from camera. The second layer evaluates each remaining candidate s_i and its two corresponding elbow candidates (e_1 and e_2) by finding the one having the angle to sagittal plane (α_1 and α_2) closest to the ρ_{ua} constraint (Equation B.1).

To define a cost function to evaluate particles, we rely on the upper arm length L_{UA} (anthropometry data [50]) constraint between shoulder and elbow as well as their 2D projection onto image plane. These prior constraints allow a two-step evaluation:

1. The first step keeps only particles having the 2D projection exactly at the 2D shoulder, other particles are set to maximum error to constantly remove them from the selection step. This results in a 3D line of particles which represents a camera ray through a 2D point into the image plane.
2. The second step evaluates the accuracy of elbow position generated by each parti-

cle. This evaluation can be done with the help from prior knowledge about kinematic constraints and the known foreshortening index ρ_{ua}.

The final cost function for evaluating a shoulder particle is therefore a combination of two steps and is defined in Equation B.2.

$$E_S(pt_s^i) = \begin{cases} \infty & \text{if } \Gamma(pt_s^i) \neq P_{shoulder} \\ \min\limits_{j=pt_e^1,pt_e^2} [G(pt_s^i, pt_e^j, \rho_{ua}) + K(pt_s^i, pt_e^j)], & \text{otherwise} \end{cases} \tag{B.2}$$

where pt_s^i represents a particle's 3D shoulder position; $\Gamma(pt_s^i)$ is the projection of pt_s^i into image space; pt_e^1 and pt_e^2 are the two possible locations of elbow generated by the current particle pt_s^i using full perspective technique (see Section A.5); $K(pt_s^i, pt_e^j)$ controls the kinematic correctness of the upper arm segment which is defined as follows: for the half left body, the elbow should be at the left side of the shoulder, and should be at the right side of the shoulder for the half right body. It returns maximum value for the correct upper arm segment, and gives no contribution for the incorrect one. $G(p_1, p_2, \rho_{ua})$ measures the similarity between the line $L_{p_1p_2}$ created by connecting two points p_1, p_2 and the foreshortening ratio ρ_{ua}. The angle (in degrees) α between $L_{p_1p_2}$ and the sagittal plane of the human body is then extracted. Using Equation B.1 for the current ρ_{ua}, we obtain the theoretically predicted angle $\alpha_{\rho_{ua}}$ for the upper arm. The term G is eventually the difference between α and $\alpha_{\rho_{ua}}$. For better compatibility with $F_{\rho_{ua}}$ defined in Equation B.1, the value of G is scaled down to $[0,1]$. The detail of this process is described in Algorithm6.

The convergence of the particle filter is fast thanks to our first step which eliminates most of the bad particles and focuses more on important particles. The particles set is kept to be used for predicting shoulder location for next frames. By doing this way, we do not have to find P_{vfoot} for each frame (only for the first frame), but directly use the estimated 3D shoulder as the seed for next frames. Since the subject is relatively stationary with respect to the camera point of view, the search range does not need to be large leading to an efficient and less computationally intensive tracking.

Algorithm 6: 3D shoulder estimation by particle filter.

```
    Input: $P^{init}_{shoulder}$, number of particles $N_p$
    Output: 3D positions of $P^{3d}_{shoulder}$
 1  Initialization: initialize a weight for each particle $i$ as $w_i^0 = \frac{1}{N_p}$
 2  Sampling:
 3  for $i = 1, \ldots, N_p$ do
 4      sample particle $\Upsilon_i^t \sim p(\Upsilon_t | \Upsilon_{t-1}^i)$
 5      calculate $w_i^t = 1 - F_i(\Upsilon_i^t)$ using Equation B.2
 6  Normalize weights
 7  Selection: replace old particles by new particles according to their weights
 8  $P^{3d}_{shoulder} \leftarrow$ take the average of $M$ best particles
 9  $t \leftarrow t + 1$
10  Return to Sampling until converge
11  return $P^{3d}_{shoulder}$
```

B.2.2 3D elbow and hand

After successfully finding the 3D position of the shoulder $P^{3d}_{shoulder}$, we apply the technique in [48]: starting from shoulder, we generate two different candidates for elbow, each candidate continues to propagate to two different candidates for hand. In summary, we have four possible candidates for the whole 3D arm whose structure is defined as $< P^{3d}_{shoulder}, P^{3d}_{elbow}, P^{3d}_{hand} >$. The filtering process involves kinematic constraints, the walking orientation and left or right half body, etc., to remove unrealistic arm poses. The best solution, however, can not be chosen from the filtering alone since there are still chances that multiple poses might pass through the tests.

We hence propose a simple method that aims to extract the most globally consistent series of arm poses if possible. We keep all the possible arm candidates that pass the filtering process (maximum four poses for each frame) in a constant number of frames N_l in memory. N_l should be at least as large as one walking cycle duration for our method to work correctly. Starting from the observation that the arm foreshortening usually happens in a short amount of time within a gait cycle, our strategy relies on without-foreshortening frames, we call them full-arm-length frames, in which all four candidates are identical and can be considered one unique pose. In essence, with 60 fps of the camera, we expect a smooth animation of the arm within N_l frames. Based on that, the aim is to find a set of arm poses having the least total inner difference from one frame to another across the N_l frames. To that end, we build a table DP (with size $4 \times N_l$) and apply dynamic programming to find the shortest path from the first frame to the last frame, see Fig.B.3 for more detail.

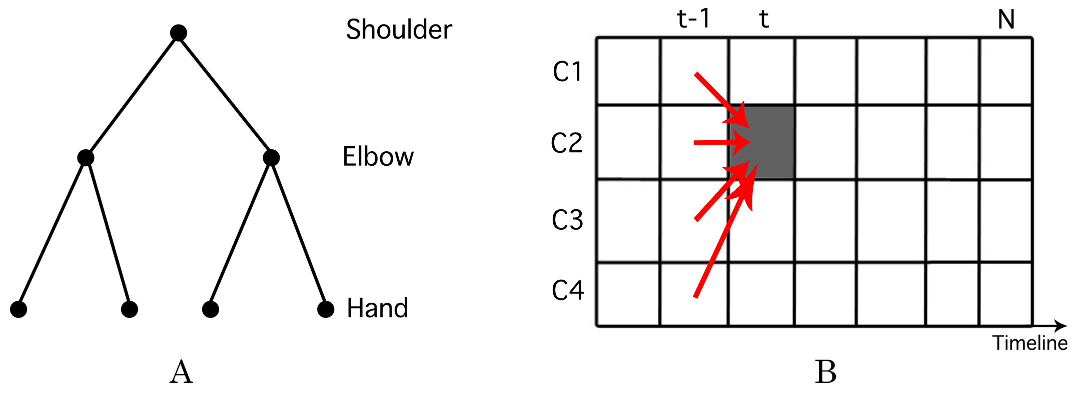


Figure B.3: (A) Elbow and hand candidates generated by weak perspective technique in [48]. Each path from shoulder to hand represents an arm pose, there are four paths in total (c_1, c_2, c_3, c_4). (B) Dynamic programming: each cell of the table is deduced by taking information from all cells in the previous column. The shortest path is found by locating the minimum cell in the last column, then tracing backward to the first column.

Each cell value (t, ct) of the table represents an arm pose and is attached with a cost function value E_{eh} defined in Equation B.3. To initialize the table, cost values of all cells in the first column of the table are set to 0.

$$E_{eh}(t, ct) = \min_{ct'=c_1,\ldots,c_4} [K(ct) + \lambda ||DP(t, ct) - DP(t-1, ct')||_2] \tag{B.3}$$

where the the first term $K(ct)$ validates the kinematic constraint of the current arm pose ct (our filtering process), the second term controls the accumulated difference propagated from previous frame $(t-1)$ to current frame t, λ is the parameter controlling the importance of each term over the other one. Each term is normalized to the range $[0,1]$ for assuring its equal contribution. We used the L2-norm (least squares) to measure the second term since L2-norm has the ability to emphasize the error differences, thus making the shortest path finding more efficient. It thus is defined as

$$||DP(t,ct) - DP(t-1,ct')||_2 = [((P^{3d}_{elbow})^{t}_{ct} - (P^{3d}_{elbow})^{t-1}_{ct'})^2 + ((P^{3d}_{hand})^{t}_{ct} - (P^{3d}_{hand})^{t-1}_{ct'})^2] \tag{B.4}$$

Briefly, Equation B.4 calculates the difference between two poses including one candidate for elbow and another one for hand. Its role is to measure the smooth of the arm movement across frames. About the performance, the complexity of our method is $\Theta(4 \times N_l)$, i.e. we have maximum four candidates in N_l frames results in quick computation. In our experimentation, we set $N_l = 4 \times 60$ (4 seconds which is equal to one cycle duration, 60 is the fps of our camera frame rate after deinterlacing). The detail algorithm is presented in Algorithm7.

Algorithm 7: Dynamic programming to find the arm poses within a N_l frames.

```
  Input: $P^{3d}_{shoulder}$, number of frames $N_l$
  Output: $< P^{3d}_{shoulder}, P^{3d}_{elbow}, P^{3d}_{hand} >$ for each frame
1 Create the table $DP$ with size $4 \times N_l)$
2 for $ct = c_1, \ldots, c_4$ do
3     $DP(0,ct) \leftarrow 0$
4 for $t = 1, \ldots, N_l$ do
5     for $ct = c_1, \ldots, c_4$ do
6         $DP(t,ct) \leftarrow$ calculate cost function using Equation B.3
7 $((P_{shoulder})^{3d}, (P_{elbow})^{3d}, (P_{hand})^{3d})^* \leftarrow$ find arm pose with minimum cost $E^*_{eh}$
  among cells at $N_l$-th column
8 Trace backward to get the pose for each frame
9 return $\forall i \in [1,N_l], < (P_{shoulder})^{3d}_i, (P_{elbow})^{3d}_i, (P_{hand})^{3d}_i >$
```

Starting from next block N_l frames, values of the first column are set as the last column of the last block. Finally, the angle at elbow is calculated from the 3D arm pose obtained above.

B.3 Experimentations

Since our ground truth is only in 2D space, we will compare our reconstruction results with the Kinect, which is well-known for their high accuracy on upper body half, for validation. Since the methodology is still complex and the results are not very good, further improvements need to be done. We therefore only show our experimentation on only one subject and from the left camera only. We present our test in Fig. B.4 and Fig. B.5 which is extracted under light and heavy foreshortening problem respectively. Our method, in essence, takes the 2D arm information as the input and this is both an advantage and a drawback. In cases of good 2D arm estimation, our method produces stable and good 3D arm reconstruction, but this is not the case if the 2D arm is bad estimated (Fig. B.5 (last row)).

The source factor that leads to average or low accuracy is skin detection noise due to not-well-lit arm parts. In heavy foreshortening cases, the hand is sometimes almost hidden and is quite darker than the rest of the arm. This affects the skin detection first and then both lower arm length and orientation afterward. Another factor that reduces our accuracy is from the lower arm points collecting step to estimate the lower arm orientation (see Section 3.3.2). This method fails if the upper and lower arm have similar orientation and the lower arm suffers from heavy foreshortening (Fig. B.5(last row)). In this case, the system collects more points than necessary, the elbow (green circle) hence is estimated much farther than it should be. One solution we choose to avoid letting this case affect other frames is that, since the appearance frequency of this case is low in real life, we consider it as noise and remove this case from the list when running dynamic table to construct the 3D arm. Future work will focus on correctly locating the elbow in failure cases above since it is essential to have a good 3D reconstruction results.

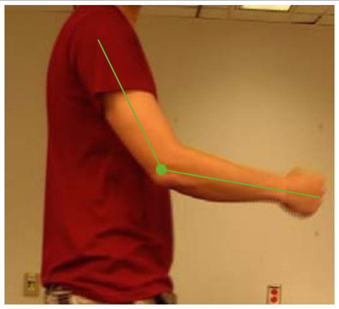
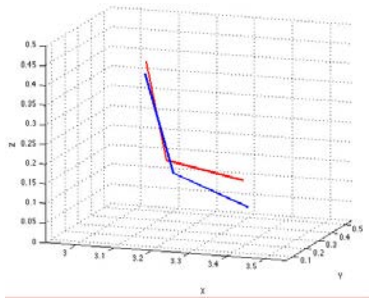
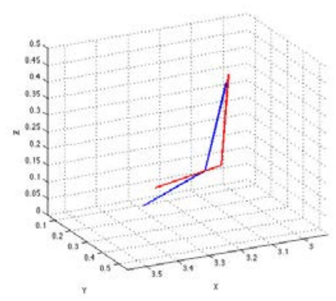
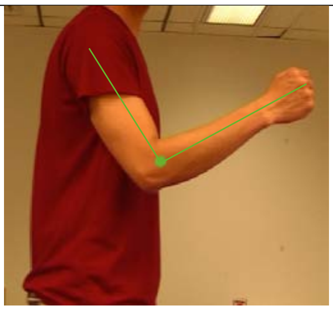
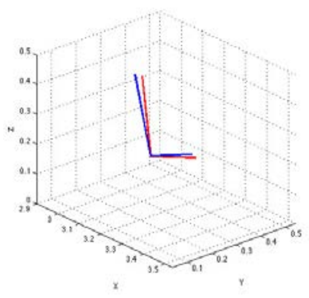
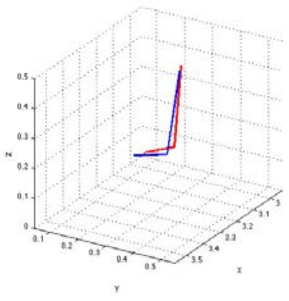
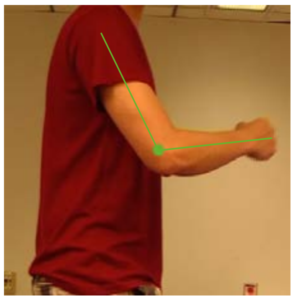
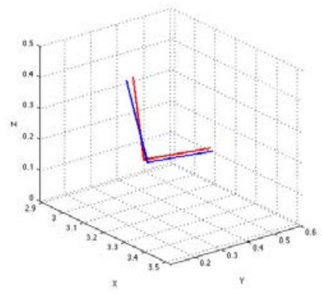
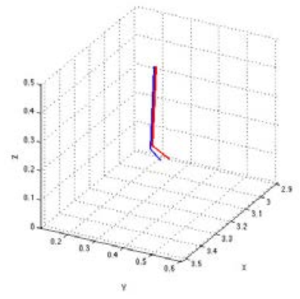
2D Arm	3D Arm (left view)	3D Arm (right view)
		
		
		

Figure B.4: Arm 3D reconstruction under light foreshortening problem. First column is our 2D arm detection (green lines and elbow is marked as green circle), the second and third columns show our results (blue lines) and Kinect's (red lines) from left and right view side in 3D space respectively. In these cases, as we can see, the elbow angle from our results are closed to that from Kinect.

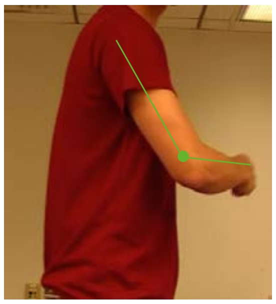
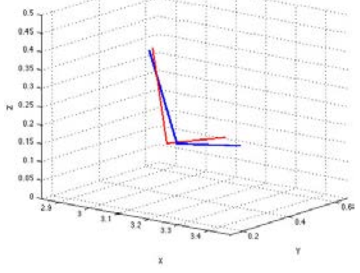
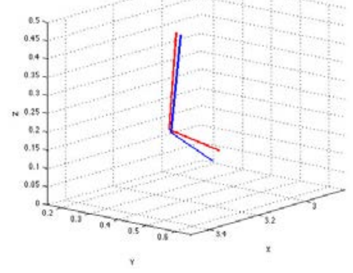
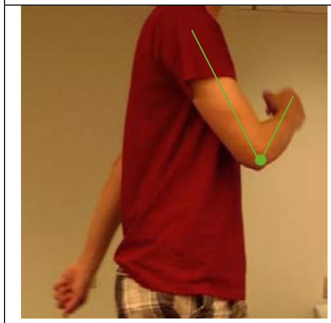
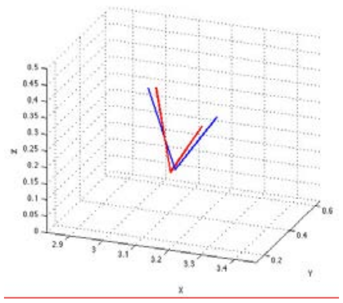
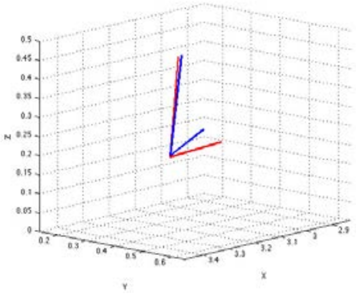
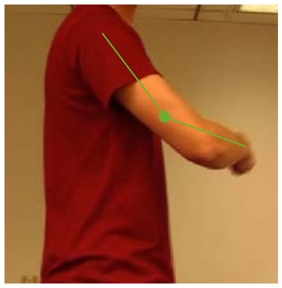
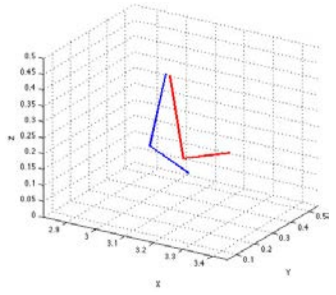
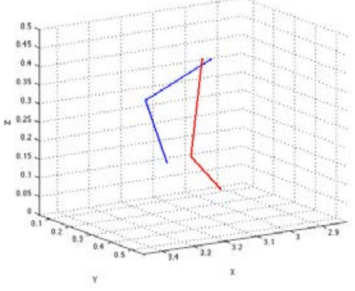
2D Arm	3D Arm (left view)	3D Arm (right view)
		
		
		

Figure B.5: Arm 3D reconstruction in heavy foreshortening problem. In these cases, the errors are clearer in the first and second row, in which the lower arm joint length is not very accurate. The last row, in which the elbow is badly estimated, produces the 3D arm reconstruction failure.