Human Body Pose Estimation With Particle Swarm Optimisation

S. Ivekovic spelaivekovic@computing.dundee.ac.uk
School of Computing, University of Dundee, Dundee DD1 4HN, UK

E. Trucco manueltrucco@computing.dundee.ac.uk
School of Computing, University of Dundee, Dundee DD1 4HN, UK

Y. R. Petillot y.r.petillot@hw.ac.uk
Department of Electrical, Electronic and Computer Engineering, EPS, Heriot-Watt University, Edinburgh EH14 4AS, UK

Abstract

In this paper we address the problem of human body pose estimation from still images. A multi-view set of images of a person sitting at a table is acquired and the pose estimated. Reliable and efficient pose estimation from still images represents an important part of more complex algorithms, such as tracking human body pose in a video sequence, where it can be used to automatically initialise the tracker on the first frame. The quality of the initialisation influences the performance of the tracker in the subsequent frames. We formulate the body pose estimation as an analysis-by-synthesis optimisation algorithm, where a generic 3-D human body model is used to illustrate the pose and the silhouettes extracted from the images are used as constraints. A simple test with gradient descent optimisation run from randomly selected initial positions in the search space shows that a more powerful optimisation method is required. We investigate the suitability of the Particle Swarm Optimisation (PSO) for solving this problem and compare its performance with an equivalent algorithm using Simulated Annealing (SA). Our tests show that the PSO outperforms the SA in terms of accuracy and consistency of the results, as well as speed of convergence.

Keywords

Articulated Human Body Pose Estimation, Still Multi-View Images, PSO.

1 Introduction

Human body pose estimation from images is an important topic in many research areas including surveillance, motion capture, human gait analysis, sign language recognition, medical analysis, and many others. While the emphasis in most of these domains is on estimating the body pose over a period of time to identify the behaviour or action, it is equally important to be able to bootstrap this process reliably and without the need for manual intervention.

In this paper we focus on the latter and address the problem of estimating the body pose from a set of still images representing the first frame of a video sequence requiring pose analysis. The results of our pose estimation algorithm can be used to initialise a full video sequence analysis, for example a body tracker. Our database contains images representative of immersive videoconferencing scenes (Isgrò et al. (2004); Mulligan et al. (2003); Baker et al. (2002)) and therefore features a person sitting

at a table, performing various gestures. Our body pose estimation algorithm focuses on the upper body as the lower body of the person is occluded by the table. The presented algorithm can be extended in a straightforward manner to handle the full body pose.

1.1 About the Application

Immersive videoconferencing aims at recreating the sense of *presence*, i.e, the sensation of sharing the same physical space with the other participants. Eye contact is an integral part of an ordinary meeting, but, as it is impossible to place a camera in the middle of the screen (where the "eyes" of remote participants appear), the video acquired from remote cameras must be warped to achieve eye contact consistently (Criminisi et al. (2003)). Warping is achieved by high-quality view synthesis, which in turn requires high quality stereo disparity data, normally difficult to achieve given the wide baselines imposed by immersive videoconferencing setups and the complexity of the body. By estimating the upper-body pose of the conference participant and subsequently fitting a model to the available disparity data, the quality of the view-synthesis can be significantly improved and the sense of presence strengthened.

1.2 Paper Structure

The structure of this paper is as follows. We begin with an overview of the related work in Section 2. Section 3 describes the particle swarm algorithm and the importance of the inertia value in directing the exploratory behaviour of the swarm as well as influencing its convergence. In Section 4 we formulate the problem of pose estimation and describe the building blocks of the algorithm that we use. Section 5 describes the steps necessary to formulate the pose estimation as a PSO optimisation problem. Section 6 presents the results of pose estimation on a test set of multi-view images and compares the performance of the PSO with simulated annealing and gradient descent optimisation and Section 7 presents the conclusions.

2 Related Work

Human body pose estimation from video data is a well-known problem. In video production and medical contexts motion and pose is often acquired with commercial sytems based on a variety on markers attached to the body. Computer vision and graphics research has instead concentrated on markerless pose and motion estimation. Prototype solutions have been reported with and without explicit body models, using image data or 3-D scanner data, and single or multiple-viewpoint video sequences. Several surveys exist which cover the recent and not-so-recent contributions in this field and are a good starting point (Gavrilla (1999); Moeslund and Granum (2001); Moeslund et al. (2006)).

Body pose estimation from images using a human body model has been addressed by various researchers, some working on pose estimation from still images and others on pose tracking in video sequences. Deutscher et al. (2000) use a skeleton combined with conical sections and an annealed particle filter. Plaenkers and Fua (2001) use an implicit surface body model which they fit to 3-D stereo data constrained by silhouette contours using the Levenberg-Marquardt optimisation. Carranza et al. (2003) use a triangular mesh body model and silhouette constraints coupled with Powell's method.

Mikic et al. (2003) use a stick figure fleshed out with ellipsoids and cylinders and an extended Kalman filter. Sminchisescu and Triggs (2003) use superquadric ellipsoids and a particle filter with covariance scaled sampling. Poppe et al. (2005) mention a similar application area to ours, virtual environments, and work on monocular video sequences using a simple body model composed of cylinders. Balan and Black (2006) combine an adaptive appearance model with the annealed particle filter and use a truncated-cone body model to represent the body pose. Balan et al. (2007) use a detailed learned model of shape and pose deformation, annealed particle filter and importance sampling search to simultaneously estimate pose and shape from multi-view images.

In the evolutionary domain, fewer attempts at articulated pose estimation and tracking from images have been reported. Shoji et al. (2000) use a genetic algorithm to estimate the pose of a 2-D articulated object from a silhouette extracted from a single view. Ho et al. (2002) use an intelligent genetic algorithm combined with a stick figure model to determine and interpret the pose of occluded articulated objects in monocular images. Ye and Liu (2005) augment the sampling step in the CONDENSATION algorithm with the genetic algorithm mutation and crossover operators to improve its performance. Hsu et al. (2006) describe a system where a stick model is manually initialised in the first frame of a single-view jump sequence and then a genetic algorithm-based search algorithm is used to find the stick models in the remainder of the sequence.

Applications of PSO to the articulated pose estimation problem are even more scarce. Schutte et al. (2004) report a parallel PSO implementation and illustrate its performance on an example of an ankle joint. Robertson et al. (2005) estimate a stick figure from a sequence of 3-D data, extracted from multiple view sequences. Robertson and Trucco (2006) use PSO to fit a stick model to sparse 3-D stereo data reconstructed from an array of cameras. In the precursor of this work, Ivekovic and Trucco (2006) estimate the pose of a subdivision upper body model from multi-view images.

The use of evolutionary optimisation methods to solve computer vision problems like articulated pose estimation still remains to be fully explored, although advances are being made. A common criticism of the evolutionary methods is their slow convergence rate, lack of consistency due to their random nature and a large number of design variables (Bureerat and Limtragool (2006)). This reputation acts as a deterrent when designing algorithms for computer vision problems where the research focus is on accuracy and on-line (real-time) performance.

Particle swarm optimisation (PSO) is an increasingly popular global search method which is gaining a somewhat different reputation (Poli et al. (2007), Poli (2007)). Schutte et al. (2004) describe advantages such as robustness, efficiency, fewer parameters than either GA or SA algorithms and its suitability to continuous variable problems. Robertson and Trucco (2006) state that it is intrinsically parallel and can easily incorporate search constraints. Many researchers report that generic settings of PSO parameters seem to work well on most problems. Recently, new theoretical advances have also been made to support these findings. A study by Poli (2008) shows that the canonical form of PSO exhibits a stable first-order moment of the sampling distribution and also reports a higher order stability for the fully informed particle

swarm (Mendes et al. (2004)). Blackwell and Bratton (2008) study the tail of the PSO position distribution and show that the standard PSO settings are a good compromise between the exploration power of the particle swarm and its convergence.

Our work differs from the related work in two aspects - the choice of the model and optimisation. Unlike other reported work, we use an articulated subdivision surface body model which was motivated by the scalability and efficiency requirements of our application, and a global search method, PSO, to recover the articulated pose parameters from multi-view still images, accurately and efficiently.

3 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) is an evolutionary computation technique introduced by Kennedy and Eberhart (1995). The idea originated from the simulation of a simplified social model where the agents were thought of as collision-proof birds and the original intent was to graphically simulate the unpredictable choreography of a bird flock.

The original PSO algorithm was later modified to improve its search capabilities and convergence and several successful applications of PSO were reported in the literature. A good starting point for a fairly recent overview of the relevant research in this area can be found in Eberhart and Shi (2004). In this paper we use the PSO with an added inertia value parameter, introduced by Shi and Eberhart (1998) and described in the next section.

3.1 PSO Algorithm with Inertia Weight Parameter

Assume an n-dimensional search space $\mathbb{S} \subseteq \mathbb{R}^n$, a swarm consisting of N particles and a fitness function $f : \mathbb{S} \to \mathbb{R}$ defined on the search space. The i-th particle is represented as an n-dimensional vector $X_i = (x_{i1}, x_{i2}, ..., x_{in})^T \in \mathbb{S}$. The velocity of this particle is also an n-dimensional vector $V_i = (v_{i1}, v_{i2}, ..., v_{in})^T \in \mathbb{S}$. The best position encountered by the i-th particle so far (*personal* best) is denoted as $P_i = (p_{i1}, p_{i2}, ..., p_{in})^T \in \mathbb{S}$ and the value of the fitness function at that position $pbest_i = f(P_i)$. The index of the particle with the overall best position so far (*global* best) is denoted as g and $gbest = f(P_g)$. Let us also denote the optimum of the fitness function f by $sol = f(P_s)$, where the index s denotes the solution position in the search space. The PSO algorithm can then be stated as follows.

1. **Initialisation**:
   - Initialise a population of particles $\{X_i\}, i = 1...N$, with random positions and velocities in the search space $\mathbb{S}$. For each particle evaluate the desired fitness function and set $pbest_i = f(X_i)$. Identify the best particle in the swarm and store its index as g and its position as P_g.
2. **Repeat** until the improvement change becomes small enough or the number of iterations reaches a predefined limit:
   - Move the swarm by updating the position of every particle according to the following two equations:

$$\begin{aligned} V_i^{t+1} &= wV_i^t + \varphi_1(P_i^t - X_i^t) + \varphi_2(P_g^t - X_i^t) \\ X_i^{t+1} &= X_i^t + V_i^{t+1} \end{aligned} \quad (1)$$

where φ_1 and φ_2 are random numbers defined by an upper limit which is a parameter of the system and w is the inertia weight parameter.

- For $i = 1...N$ update $pbest_i$ and $gbest$.

The parameters φ_1 and φ_2 influence the *social* and *cognition* components of the swarm behaviour (Shi and Eberhart (1998)). They are composed of a random number and a constant and can be written as $\varphi_1 = c_1 rand_1()$ and $\varphi_2 = c_2 rand_2()$, where c_1 and c_2 are two constants and $rand_1()$ and $rand_2()$ two random numbers in the interval $[0, 1]$. In our experiments the values of the constants c_1 and c_2 were both set to integer 2, which on average made the weights for social and cognition components of the swarm equal to 1 (Shi and Eberhart (1998); Kennedy and Eberhart (1995)).

3.2 Inertia weight parameter

The value of the inertia weight w can remain constant throughout the search or change with time. It plays an important role in directing the exploratory behaviour of the particles. Higher inertia values push the particles to explore more of the search space and emphasise their individual velocity while lower inertia values force particles to focus on a smaller search area and move towards the best solution found so far.

Shi and Eberhart (1998) addressed the influence of different inertia values on the exploratory abilities of the swarm. They tested inertia values in the interval $[0, 1.4]$ and found that for a constant inertia value a medium value of w, i.e., $0.8 < w < 1.2$, had the best chance of finding the global optimum while also requiring a moderate number of iterations. Large values of w, i.e., $w > 1.2$ made PSO behave more like a global search method always trying to exploit new search areas.

In this paper we use a time-varying inertia weight. We model the change over time with an exponential function which allows us to use a constant sampling step while gradually guiding the swarm from a global to more local exploration:

$$w(x) = \frac{A}{e^x}, \quad x \in [0, \ln(10A)], \tag{2}$$

where A denotes the starting value of w when $x = 0$. The optimisation terminated when $w(x)$ fell below 0.1. The sampling variable x was incremented by $\Delta x = ln(10A)/N$, where N is the desired number of inertia weight changes.

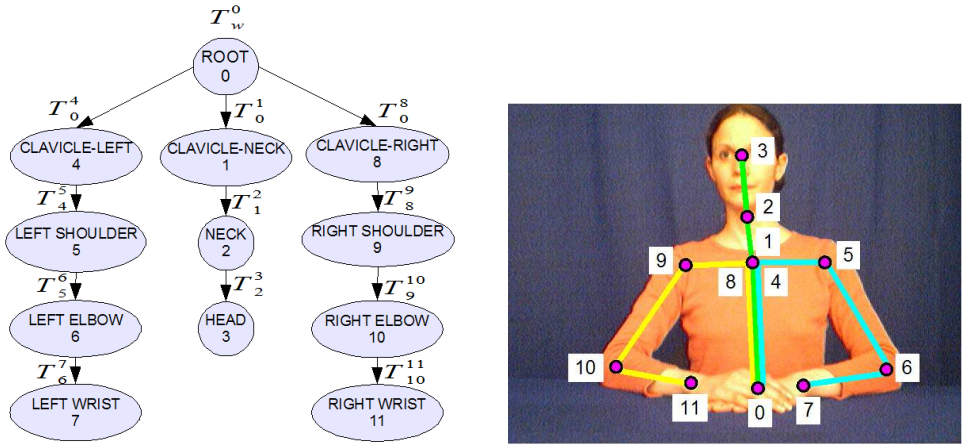
The swarm is allowed to explore the search space with a particular inertia value for as long as every move of the swarm improves the current global optimum estimate. As soon as an iteration fails to improve the estimate, the value of the sampling variable increases and the inertia weight value decreases accordingly. This forces the swarm to identify possible optimum regions at the very beginning, then focus on the best few, and eventually settle down in the most promising region and find the global optimum.

4 Pose Estimation

In this section we formulate our pose estimation problem and describe its building blocks in more detail. We describe the body model which illustrates the estimated pose and define the cost function used to evaluate the quality of a particular pose estimate.

4.1 Problem Formulation

The input to our pose estimation algorithm is a set of still images taken from multiple viewpoints. The images are background-foreground segmented and the foreground



(a) (b)

Figure 1: (a) Kinematic tree structure of the upper body model, consisting of three chains with a common root node, and the corresponding transformations. (b)Kinematic chains shown overlaid on the upper body.

converted into silhouettes which serve as constraints for the pose estimation. The individual pose estimates are expressed in the form of a 3-D body model (see Figure 10). The assumption is that when the pose expressed by the model closely matches the pose of the person in the original set of images, the silhouettes generated by the model also closely match the silhouettes extracted from the original images. The process of estimating the body pose is then formulated as finding the pose which maximises the overlap of the silhouettes.

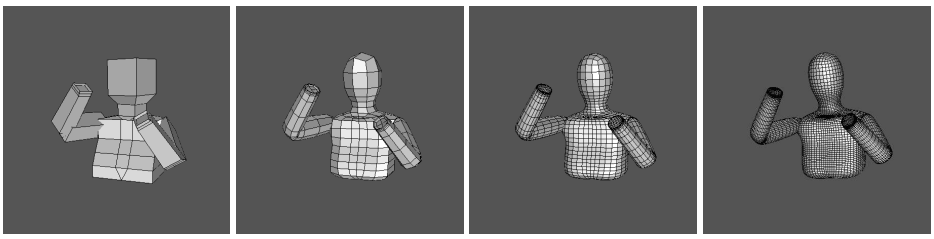
4.2 Body Model

We use a 3-D layered subdivision surface body model consisting of two layers, the *skeleton* and the *skin*. The skeleton layer is defined as a set of homogeneous 4×4 transformation matrices T_i^j which encode the information about the position and orientation of every joint with respect to its parent joint in the kinematic tree hierarchy:

$$Skeleton = \{T_w^0, T_0^1, T_1^2, ..., T_{N-1}^N\}, \tag{3}$$

where $N + 1$ is the number of joints in the skeleton and T_i^j is a homogeneous transformation matrix encoding the orientation of the coordinate system of joint j with respect to the coordinate system of joint i. Subscript w denotes the world coordinate system. The top of the hierarchy is the root joint which branches out into a number of kinematic chains modelling the skeletal structure of the human body as shown in Figure 1.

The skin layer represents the second layer in the model and is connected to the skeleton through the joints' local coordinate systems. Each of the joints controls a certain area of the skin. Whenever a joint or limb moves, the corresponding part of the skin moves and deforms with it. The skin can therefore be described as a set of transformation matrices forming the skeleton layer, combined with the sets of points influenced



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

Figure 2: The skin is modelled as a subdivision surface. (a) The base mesh. (b)-(d) The model after 1, 2 and 3 levels of subdivision.

by each of the transformations:

$$Skin = \{\{T_w^0, P_{T_w^0}\}, \{T_0^1, P_{T_0^1}\}, \ldots, \{T_{N-1}^N, P_{T_{N-1}^N}\}\} \tag{4}$$

In order to generate a smooth skin surface of the model, all the skin points $P_{T_i^j}$ have to be transformed into a common coordinate system such as the world coordinate system by multiplying together the appropriate transformation matrices, as specified by the kinematic tree hierarchy $\mathcal{H}$:

$$P_w = T_w^0 \cdot T_0^1 \cdot \ldots \cdot T_i^j \cdot P_{T_i^j}, \quad \forall\, i,\ j \in \mathcal{H} \tag{5}$$

The points (vertices) P_w^i are connected with edges into faces F to form a base mesh (see Figure 2(a)):

$$\mathcal{M}_0 = \{V, F\}, \text{where } V = \{P_w^i\}, F = \{P_w^{i_1}, P_w^{i_2}, P_w^{i_3}, P_w^{i_4}\} \tag{6}$$

Finally, to obtain the smooth limit surface $\mathcal{M}_\infty$ of the base mesh, i.e., the skin in Figure 2(b)-(d), the base mesh is repeatedly subdivided:

$$\mathcal{M}_\infty = \lim_{k\to\infty} \mathcal{S}^k \mathcal{M}_0, \quad k > 0, \tag{7}$$

where $\mathcal{S}^k$ denotes the k-th application of the Catmull-Clark subdivision operator (Warren and Schaeffer (2004)).

4.3 Cost Function

The cost function compares the silhouettes extracted from the original images with the silhouettes generated by the model in its current pose. The original images can be acquired from N different viewpoints. Each image is foreground-background segmented and binarised to obtain a silhouette. Let the images containing the *original* silhouettes be denoted as I_i^o, $i = 1...N$. Similarly, let I_i^m, $i = 1...N$ denote images of the *model* silhouettes. The cost function can then be written as follows:

$$E = \sum_{i=1}^{N} \frac{\omega_i}{Z_i} \sum_{1}^{row} \sum_{1}^{col} (I_i^o \,\&\, I_i^m), \tag{8}$$

where row and col denote the image dimensions, i.e., number of rows and columns, respectively, and $\&$ denotes the logical *AND* operation. Coefficients Z_i are the normalisation constants obtained by counting the number of silhouette pixels in every original image. Weights ω_i control the contribution of every view to the total error count. We tested different weight configurations and describe the results in Section 6.

JOINT (index)	DOF
Root location (root)	3
Root orientation (0)	3
Clavicle-neck orientation (1)	2
Clavicle-left orientation (4)	2
Left Shoulder orientation (5)	3
Left Elbow orientation (6)	1
Clavicle-right orientation (8)	2
Right Shoulder orientation (9)	3
Right Elbow orientation (10)	1
TOTAL	20

Table 1: Degrees of freedom (DOF) of the upper body model. For reference, the joints have been enumerated according to the kinematic tree in Figure 1.

5 Pose Estimation with PSO

In PSO, each particle represents a potential solution in the search space. Our search space is the space of all plausible skeleton configurations. The individual particle's position vector in the search space is specified as follows:

$$X_i = (root_x, root_y, root_z, \alpha_x^0, \beta_y^0, \gamma_z^0, \alpha_x^1, \beta_y^1, \gamma_z^1, ..., \gamma_x^N), \tag{9}$$

where $root_x, root_y, root_z$ denote the position of the root joint (first joint in the kinematic chain) with respect to the reference (world) coordinate system, and $\alpha_x^i, \beta_y^i, \gamma_z^i$ refer to rotational degrees of freedom of joint i around the x, y, and z-axis, respectively.

5.1 Hierarchical Approach

As shown in Table 1, 20 parameters were used to define the upper-body pose. Optimising the entire configuration at once amounts to searching for an optimum of a multi-modal function in a 20-dimensional search space which is a challenge for any optimisation method and initial experiments with the PSO confirmed that this was indeed the case. The complexity of the cost function would at the very least require a large swarm size which was not feasible in our approach as the cost function became prohibitively expensive with the increase in the number of particles and consequently the time needed for PSO to produce a pose estimate became unacceptably long.

We decided to instead exploit the nature of the problem and formulate the search in a hierarchical manner, taking advantage of the hierarchical nature of the kinematic chain. The hierarchy means that the positions of the joints lower in the chain are constrained by the configurations of the joints higher in the chain and we can optimise for the joint rotations in a hierarchical manner, thereby reducing the complexity of the search. Depending on the level of articulation of individual joints and dependency between them the optimisation can be done one joint at a time or by grouping several joints together.

TORSO (**Step 1**) Root location 3DOF: $root_x, root_y, root_z$ (**Step 2**) Root orientation 3DOF: $\alpha_x^0, \beta_y^0, \gamma_z^0$	**RIGHT UPPER ARM** (**Step 5**) Clavicle-right orientation + Right shoulder orientation 4DOF: $\alpha_x^8, \gamma_z^8, \alpha_x^9, \gamma_z^9$
NECK & HEAD (**Step 3**) Clavicle-neck orientation 2DOF: α_x^1, γ_z^1	**LEFT LOWER ARM** (**Step 6**) Left shoulder orientation + Left elbow orientation 2DOF: β_y^5, α_x^6
LEFT UPPER ARM (**Step 4**) Clavicle-left orientation + Left shoulder orientation 4DOF: $\alpha_x^4, \gamma_z^4, \alpha_x^5, \gamma_z^5$	**RIGHT LOWER ARM** (**Step 7**) Right shoulder orientation + Right elbow orientation 2DOF: β_y^9, α_x^{10}

Table 2: 7 steps in the hierarchical optimisation

We performed the hierarchical optimisation in 7 steps (see Table 2). First, we optimised the location of the skeleton in space, i.e., the location of the root joint, followed by the root joint orientation. These were both 3 DOF optimisations. Once the skeleton was positioned in space, we optimised the neck and head sub-chain, for which we only used 2 DOF in the clavicle neck joint to model the tilt of the head. The movement of the clavicle left and clavicle right joint on their own does not produce enough variation in the silhouette shape to be optimised individually. Therefore, in the next step, we combined the left clavicle joint with two rotational dimensions of the shoulder joint and optimised the parameters of the left upper arm, a 4 DOF optimisation. Likewise, we then optimised the right upper arm, again 4 DOF. At the end we were left with the left and right lower arm, each modelled with 2 DOF. The two 4 DOF upper arm optimisations required a slightly denser sampling of the inertia weight function to correctly locate the optimum region.

When optimising joint parameters hierarchically, the joints lower in the hierarchy mislead the silhouette overlap count as they contribute to it despite of not having been optimised yet. We avoid this by deforming the subdivision body model so that at a particular stage of the optimisation only those body parts which are currently optimised or have already been optimised are visible. The hands were excluded from the model entirely to avoid having to interpret their high level of articulation which often considerably misled the optimisation.

The shoulder optimisation is split into two separate stages because the rotation of the shoulder around its own axis (β_y^5 and β_y^9 in Table 2) affects the movement of the lower arm and is only properly visible when the lower arm is present. Due to the hierarchical nature of our optimisation algorithm, the lower arms become visible only in steps (**6**) and (**7**) and correspondingly the shoulder parameters β_y^5 and β_y^9 are optimised then. A further illustration can be found in Ivekovic and Trucco (2006).

Figure 3: The left image shows the result under the influence of the error propagation in the hierarchical approach. The right image shows the correction.

Using the hierarchical approach as described has a disadvantage in cases where the pose estimation is not very accurate early in the hierarchy. In those cases the erroneous estimate propagates throughout the hierarchy and influences the quality of the subsequent estimates (see Figure 3). When the silhouette information is sufficiently articulated and unambiguous, this can be corrected by running another optimisation on the whole kinematic chain at once, by initialising the particles around the hierarchical estimate and letting them settle into a new optimum. We found that, most of the time, the slight misalignment of the torso which triggered the error propagation, did not really need correcting itself and we instead focused the effort on the arms only. To make this step more efficient, we optimised each arm separately as a 6 DOF optimisation.

6 Experiments

This section describes the experimental results. For the purpose of the comparison a set of 6 images (6 different viewpoints) of 12 different body poses was acquired. The front view for each of the 12 poses is shown in Figure 4.

6.1 Experimental Setup

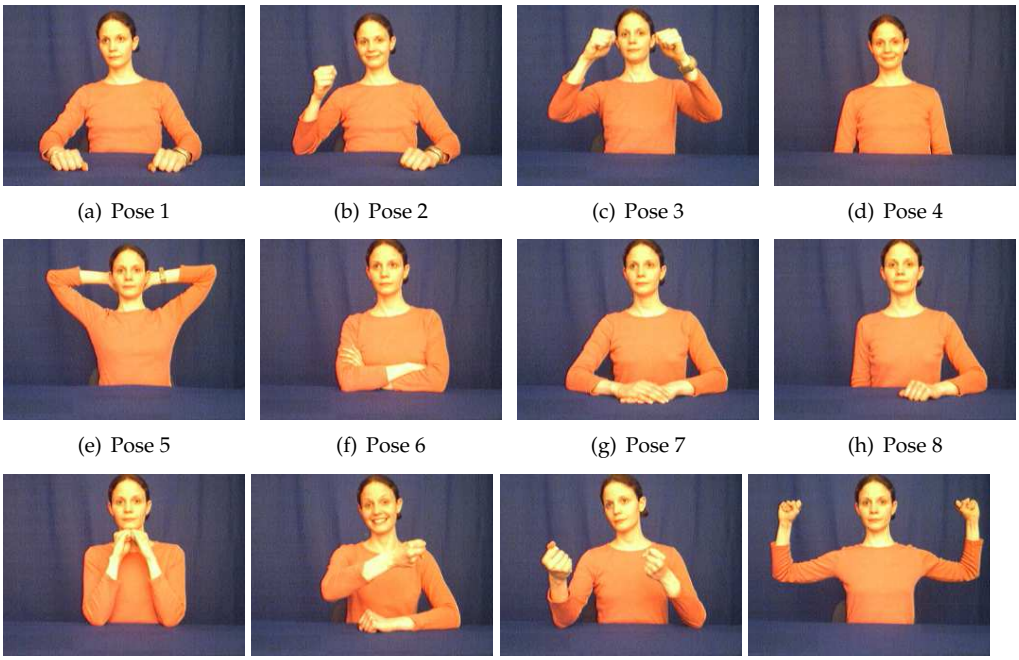
The images were acquired with a set of 6 IEEE 1394 webcams. The configuration of the cameras is schematically shown in Figure 5 with example views from each camera. The images were acquired in the 640×480 RGB mode. A uniform blue colour of the background was used to facilitate the foreground-background segmentation and generation of silhouettes. This is not an unreasonable demand for the type of application that we were modelling with the setup.

6.2 Cost Function

The cost function defined in Equation 8 can be interpreted in different ways, depending on the choice of the weights ω_i. The simplest choice is to set all ω_i to be equal. However, when certain views contain occlusions, it would be intuitively useful to give more weight to views which are not occluded or less occluded and can guide the optimisation better. In the test set, the top view contains the lowest number of occlusions and could be given a greater emphasis to disambiguate other views. Another choice is to change the importance of the views dynamically in each iteration, by checking the quality of the overlap in every view and assigning more importance to the view where the optimisation is doing worst, thereby forcing it to explain it better.

We experimentally compared the following 4 different weight configurations:

1. All views equally important, $\omega_i = 1.0$;



(a) Pose 1 (b) Pose 2 (c) Pose 3 (d) Pose 4

(e) Pose 5 (f) Pose 6 (g) Pose 7 (h) Pose 8

(i) Pose 9 (j) Pose 10 (k) Pose 11 (l) Pose 12

Figure 4: 12 poses used in the comparison experiment.

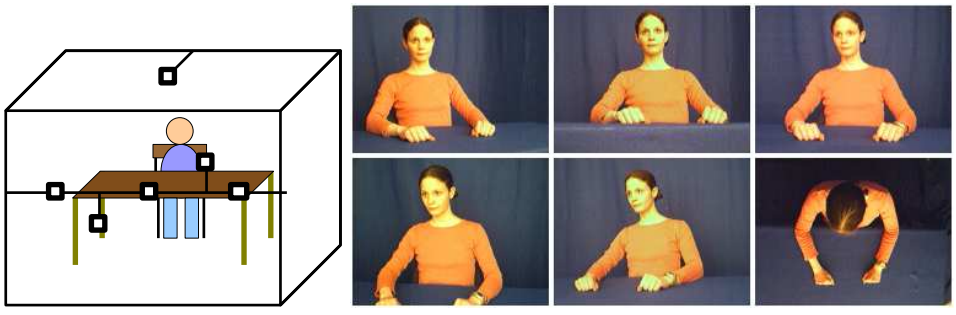


Figure 5: A schematic display of the camera configuration and an example of the corresponding camera views.

Configuration	Overlap Mean	Overlap Std. deviation
Configuration 1	0.887156	0.050874
Configuration 2	0.885688	0.052481
Configuration 3	0.887059	0.052004
Configuration 4	0.887792	0.051282

Table 3: Different weight configurations of the cost function and the corresponding pose estimation results calculated over 10 repetitions of the hierarchical pose estimation on 12 different pose examples. Each pose estimate was evaluated by comparing the size of the model-silhouette overlap with the size of the original silhouette, averaged over all 6 views. The entries in the table show what percentage of the original silhouettes the overlap achieved - the higher the percentage, the more of the original silhouettes were explained by the estimated pose. An average silhouette size computed over all 12 poses used in the experiment is approximately 77588 pixels. This shows that, on average, the overlap size using the configuration 4 was approximately 49.3 pixels larger than the overlap size using the configuration 1.

2. More importance was given to the top view with the weight $\omega_{top} = 0.25$, while the other views had weights set to $\omega_i = 0.15$

3. One higher weight, $\omega_{worst} = 0.25$, was dynamically assigned to the view with the worst fit, while all the other views had weights set to $\omega_i = 0.15$;

4. Out of six views we manually chose three which contained fewer or no occlusions and gave them a higher weight $\omega_j = 0.25$, while the remaining three had weights set to $\omega_i = 0.0833$, $i \neq j$.

Each weight configuration was tested on 12 different poses by running hierarchical pose estimation with PSO 10 times per pose. The quality of the fit over all 120 runs was then evaluated for each configuration and the resulting mean and standard deviation statistics are shown in Table 3. As we do not have the ground truth information (such as optical motion capture data) available for the poses in the test set, and setting the ground truth manually is rather subjective, we instead evaluate the quality of the fit by counting the number of pixels in the final overlap between the model's silhouettes and the silhouettes extracted from the original images. The bigger the overlap, the better the agreement between the true pose and that of the model.

Experimental results show that configuration 4, where more important views were chosen manually, performed best, followed by the configuration 1, where all views were given equal importance. Although the top camera does contain more information in poses with front-view occlusions, giving solely the top view a higher weight did not work as well as simply allowing all views to have the same influence. As we do not have a way of estimating the importance of the views automatically at this point, using configuration 4 would require labelling views manually in advance of every pose estimation. Our goal is a fully automatic pose estimation method and we have therefore decided to use the second best configuration instead and treat all views as equally important.

6.3 PSO Pose Estimation Results

We tested the performance of the PSO algorithm on a number of images acquired by the setup schematically shown in Figure 5. Due to the complexity of the evaluation function, we limited the swarm size to only 10 particles and set the starting inertia value to $w = 1.2$. The swarm was initialised around the default pose shown in Figure 10(a), which made an assumption about the approximate root position, as the person in the image was sitting on a chair, but made no assumptions about the body pose itself. The swarm was scheduled to move once per each inertia value, which changed if a better solution was found. In that case, the corresponding inertia value was kept the same until a swarm move without improvement on the global optimum forced it to decrease again.

Results in Figure 6 are the examples of poses where the estimate matches the real pose rather accurately as the silhouette constraints were informative enough and did not contain significant occlusions. Figure 7 shows the results of running the pose estimation on poses with missing or folded lower arms (invisible in the silhouette), a realistic scenario in a videoconferencing application. Although all views were given equal importance, the optimisation often managed to fold the arms in front or under the body to achieve a better overlap in the top view (see the lower arms in top camera view for pose 4, 6 and 9). This makes sense as by the time the optimisation reached the lower-arm step (steps 6 and 7 in Table 2), the top view was the only one with bits of silhouette left unexplained, a consequence of the rather approximate generic model dimensions.

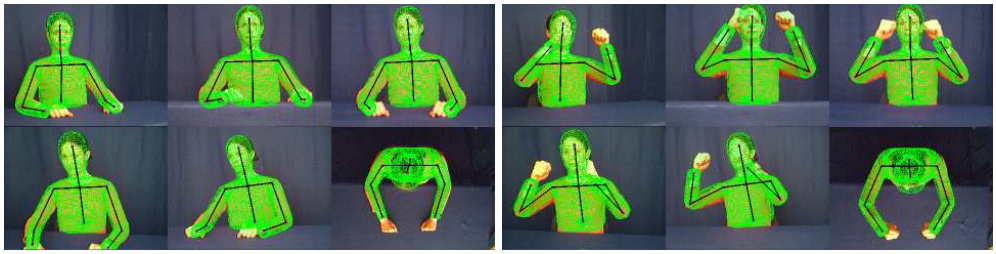
Figure 8 shows the pose examples where the optimisation failed to recover the correct pose. In pose 2, the not-so-ideal shape and size of the model made the correct pose less obvious and the upper arm estimate almost completely occluded the lower arm in two of the views. In pose 10, the right arm was occluded in most of the views, while the lower right arm estimate also managed to lock onto the left hand to increase the overlap count. In pose 11, the left lower arm was occluded by the body in three views and by the upper arm in two views. The resulting pose explains the top camera very well and provides a minimally visible left lower arm in the rest of the views. The corresponding pose estimate overlaps with the silhouettes are also shown in Figure 8 to illustrate the ambiguity in silhouette constraints.

6.4 Different Test Subjects

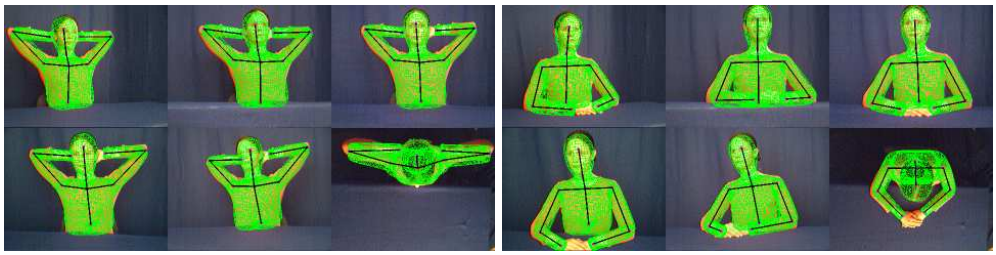
Estimating the body pose for different people requires resizing the model to fit the body proportions of each individual. Although this could be done automatically, we currently adjust the model for each person manually, using a default body pose similar to the one shown in Figure 10(a). The model must only be customised once per every individual. When adjusting the model dimensions, the assumption is made that the clothes are sufficiently tight not to significantly change the appearance of the body when the person moves. Figure 9 shows results on different test subjects.

6.5 Comparison Experiments

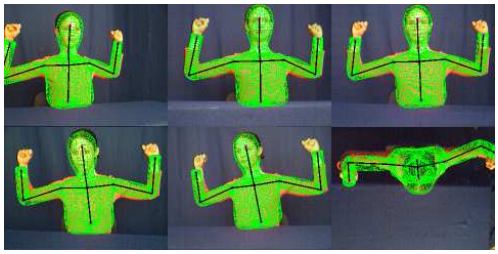
We performed a gradient descent (GD) optimisation with a constant step size parameter (Fletcher (2006)) to demonstrate that the pose estimation problem was complex enough to warrant a global optimisation approach such as PSO. The optimisation was initialised randomly around the canonical pose estimate shown in Figure 10(a).



(a) Pose 1 (b) Pose 3



(c) Pose 5 (d) Pose 7



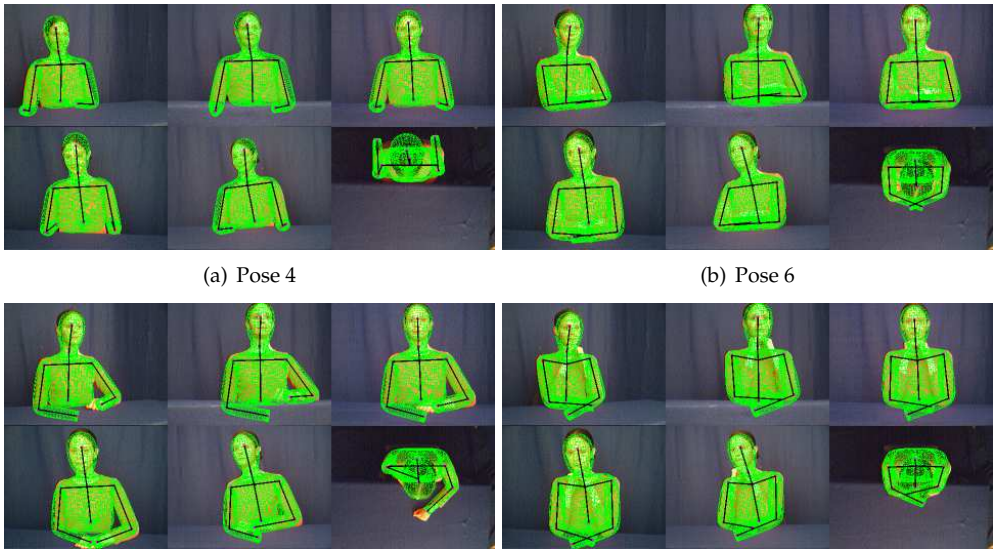
(e) Pose 12

Figure 6: Successful results of the pose estimation with PSO. For every pose example, the model is shown overlaid on top of the original image in the corresponding six views. The views in this figure are sufficiently discriminative to allow for a good interpretation of the pose.

Despite some of the runs producing a good pose estimate, the resulting pose estimates were sufficiently inconsistent to indicate that the cost function landscape was too complex for a simple downhill optimisation such as gradient descent (see Figure 10(b)).

We also compared the performance of PSO with the downhill-simplex simulated annealing (Press et al. (2002)). Simulated annealing (SA) with its annealing schedule and PSO with the decreasing inertia parameter intuitively behave in a similar way when exploring the search space. Both start with a high degree of randomness and global search and gradually slow down to explore a smaller area of the search space. Despite the apparent similarity in the general behaviour of the two methods, this is not true for the parameters which they use. In order to enable a fair comparison of the two methods, care must be taken when setting these parameters. We explain the chosen parameter settings next.

The PSO settings that we used were obtained through experimental trials. The



(a) Pose 4 (b) Pose 6

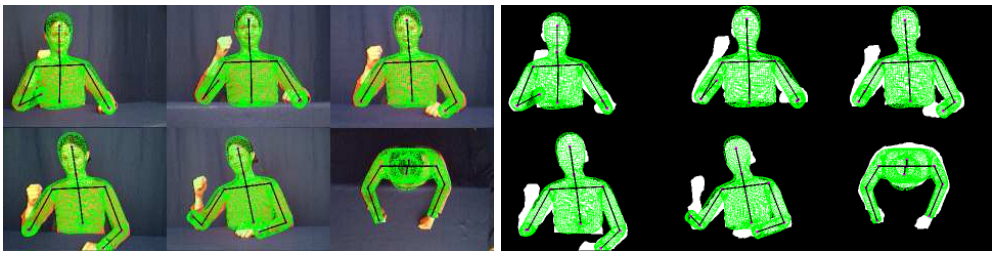
(c) Pose 8 (d) Pose 9

Figure 7: Results on deliberately occluded poses. PSO attempted to explain the lower arms by folding them unto or under the body. No default pose was enforced in these cases, the optimisation was left to explain the pose the best it could.

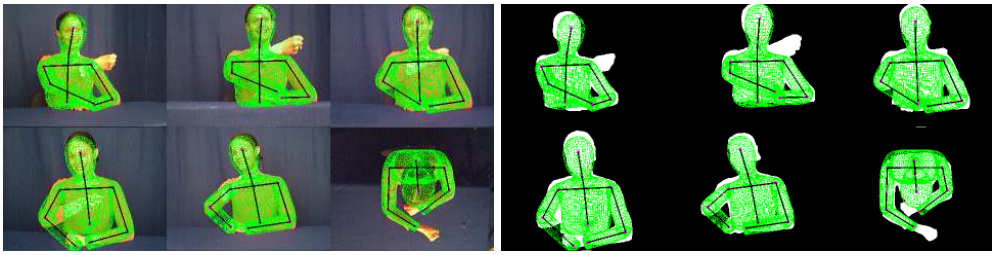
starting inertia value had to be high enough to enable global exploration of space. In our experiments it was set to $w = 1.2$. The exponential function in Equation 2, used to model the inertia change over time, was sampled with a sampling step equal to 0.05. The optimisation was terminated when the inertia value fell below 0.1. The swarm consisted of 10 particles and was allowed one move per inertia value, unless it found a better solution, in which case it was allowed to stay at the current inertia value for another move.

For the Simulated Annealing, the starting and stopping conditions were set according to the guidelines in Salamon et al. (2002). The starting temperature was set to $T_0 = 10.0$ which meant that, at the start, every move was accepted and the global exploration was encouraged. The annealing schedule was chosen to resemble the inertia change function in PSO as closely as possible. The temperature decrease schedule was set to $T_{new} = 0.95 * T_{old}$. The algorithm had 10 iterations available at every temperature level, corresponding to the 10 particles in the PSO. Like PSO, if a better solution was found at a particular temperature level, the algorithm was allowed to remain at that temperature for another 10 iterations. The optimisation was terminated either when the temperature fell below 10^{-4} or when the improvement step fell below the tolerance threshold of 10^{-5}.

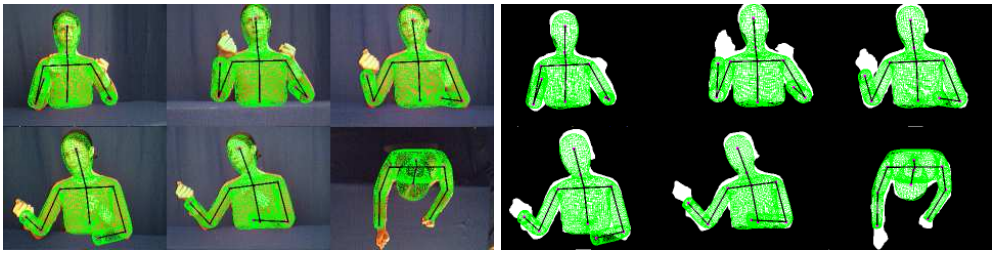
We attempted a comparison of both PSO and SA on a constrained search space. For PSO this meant that whenever a particle crossed the boundary of the search space in a particular dimension, the position of the particle in that dimension was set back to the boundary value and the corresponding velocity vector reversed. In SA, the cost function was designed to return a high error value (the pose estimation was



(a) Pose 2



(b) Pose 10



(c) Pose 11

Figure 8: Examples where PSO failed to recover the correct pose. These failures seem mainly the result of a lack of constraints as limbs are occluded in sufficiently many views to make the interpretation of the pose from the silhouettes very ambiguous. The overlap of the model and the silhouette is also shown to better illustrate where the pose optimisation has gone wrong.

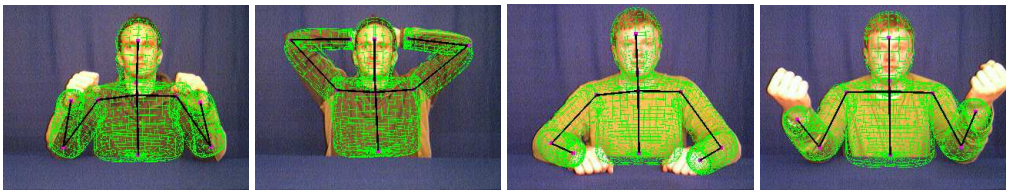
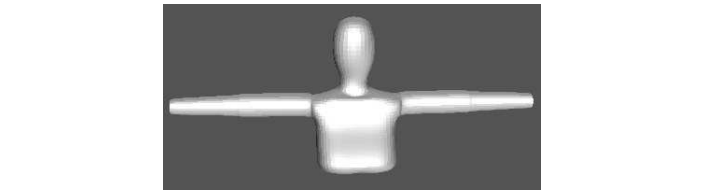
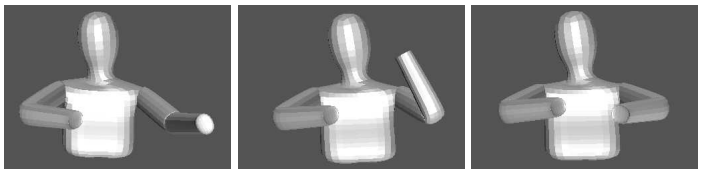


Figure 9: Pose can be estimated for different people, the only requirement is that the model dimensions are adjusted to those of the imaged person.

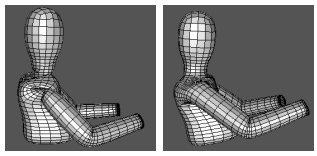


(a)



(b)

Figure 10: (a) Default model pose; all optimisation experiments were initialised randomly around the default pose estimate. (b) Three different pose estimates for pose number 3, generated by three sequential randomly initialised runs of gradient descent.



(a) (b)

Figure 11: Two different results of the unconstrained SA. The twist of the skin indicates that the rotation of the shoulder in the result (a) is overestimated by at least one period. In another run, (b), the same shoulder rotation was explained by the identical rotation in its original period. While unconstrained optimisation still produces a plausible pose estimate, even if outside the primary search space, it takes longer to converge, which is another significant drawback.

formulated as a minimisation) to discourage the optimisation from searching in that area. Experiments showed that with only 10 iterations per energy level this strategy did not allow the optimisation to make a sensible recovery once it crossed the border of the allocated search space and consequently that optimisation step failed. Constraining the search to the allocated search space was much more easily achieved using PSO as the direction of the particles was specified by their velocity vector and could be easily reversed when the boundary was reached. To allow for a fairer comparison, we present the results of the unconstrained SA instead, with the drawbacks illustrated in Figure 11.

Figure 12 shows the error-bar comparison of the three different optimisation methods. The error bars indicate the mean and standard deviation of the overlap, calculated for each individual pose. Although some SA and GD estimates did exceed PSO (see poses 11 and 9), the overall performance of PSO was better and more consistent. In terms of efficiency, PSO came top, followed by GD and SA. On the set of 12 poses, PSO took on average 136.11 ± 5.7 seconds to complete and required 414.92 ± 16.7 cost

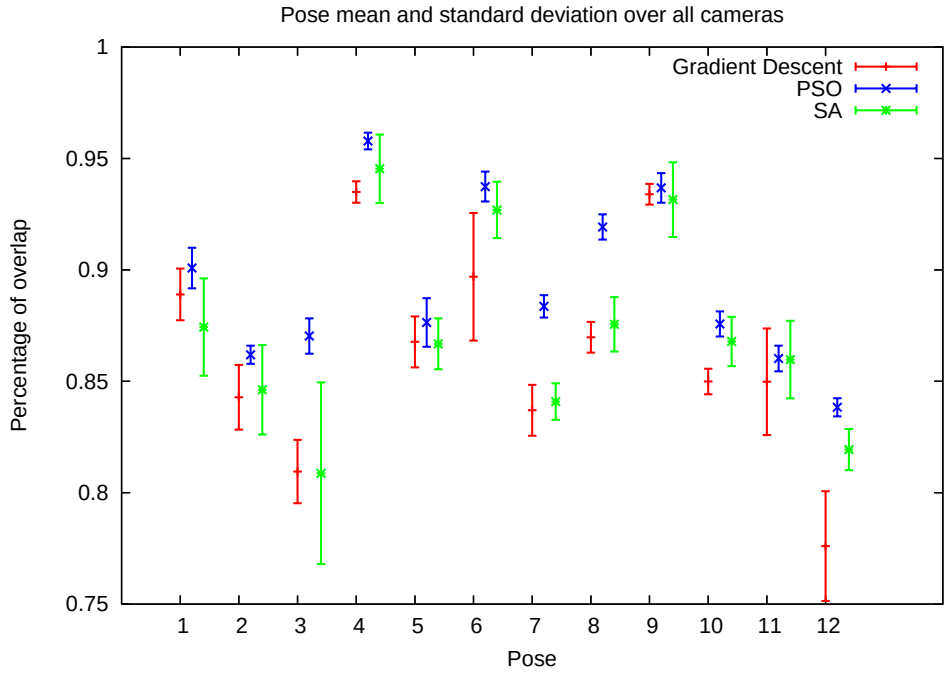


Figure 12: The results of the pose estimation comparison using GD, PSO, and SA. Results are shown in triplets for every pose, starting with GD, PSO and SA result for Pose 1. The GD error bar is always positioned above the corresponding pose number and the PSO and SA to the right of it. Across the entire test set, the pose estimates produced by PSO are more consistent than those of SA, while GD produces the worst performance. The variability in the SA results for pose 3 are a consequence of the lack of constraints as the search several times failed to correctly identify the root position and then produced significantly differing explanations of the remaining parts of the model. The SA performance on other poses was better, although the amount of variability in the results was larger than that of PSO.

function iterations, GD took on average 323.31 ± 66.4 seconds to complete and required 9562.5 ± 1964.5 cost function evaluations, while SA took on average 503.44 ± 31.06 seconds to complete and required 15035.58 ± 864.1 cost function evaluations.

7 Conclusions

In this paper we have presented upper-body pose estimation with Particle Swarm Optimisation. The presented approach can be extended to a full body pose estimation simply by extending the kinematic chain and using it on different people only requires the correct initialisation of the generic model dimensions. The presented experiments illustrate the ability of the method to solve the problem consistently and more efficiently than an equivalent pose estimation algorithm using simulated annealing or gradient descent method. Current work is concentrating on PSO pose tracking on video sequences, which, due to the restrictions of our simple camera setup was not possible during the work reported here. We hope that the results shown in this paper are illus-

trative enough to foster the use of PSO in related problems and so help to bridge the gap between the evolutionary methods and their use in computer vision.

References

Baker, H., Tanguay, D., Sobel, I., Gelb, D., Gross, M., Culbertson, W., and Malzbender, T. (2002). The coliseum immersive teleconferencing system. In *International Workshop on Immersive Telepresence*.

Balan, A. O. and Black, M. J. (2006). An adaptive appearance model approach for model-based articulated object tracking. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, pages 758–765.

Balan, A. O., Sigal, L., Black, M. J., Davis, J. E., and Haussecker, H. W. (2007). Detailed human shape and pose from images. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, pages 1–8.

Blackwell, T. and Bratton, D. (2008). Examination of particle tails. *Journal of Artificial Evolution and Applications*, 2008:14 – 23.

Bureerat, S. and Limtragool, J. (2006). Performance enhancement of evolutionary search for structural topology optimisation. *Finite Elements in Analysis and Design*, 42(6):547–566.

Carranza, J., Theobalt, C., Magnor, M., and Seidel, H. (2003). Free-viewpoint video of human actors. *ACM Transactions on Graphics*, 22(3):569 – 577.

Criminisi, A., Shotton, J., Blake, A., and Torr, P. (2003). Gaze manipulation for one-to-one teleconferencing. In *Proceedings of ICCV 03*, pages 191 – 198.

Deutscher, J., Blake, A., and Reid, I. (2000). Articulated body motion capture by annealed particle filtering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2000*, volume 2, page 2126.

Eberhart, R. C. and Shi, Y. H. (2004). Special issue on particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3).

Fletcher, R. (2006). *Practical Methods of Optimization*. John Wiley and Sons.

Gavrilla, D. (1999). Visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82 – 98.

Ho, S.-Y., Huang, Z.-B., and Ho, S.-J. (2002). An evolutionary approach for pose determination and interpretation of occluded articulated objects. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 1092 – 1097.

Hsu, H.-H., Hsieh, S.-W., Chen, W.-C., Chen, C.-J., and Yang, C.-Y. (2006). Motion analysis for the standing long jump. In *26th IEEE International Conference on Distributed Computing Systems Workshops*, pages 47 – 52.

Isgrò, F., Trucco, E., and Schreer, O. (2004). Three-dimensional image processing in the future of immersive media. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(3):288–303.

Ivekovic, S. and Trucco, E. (2006). Human body pose estimation with pso. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1256–1263.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948.

Mendes, R., Kennedy, J., and Neves, J. (2004). The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210.

Mikic, I., Trivedi, M., Hunter, E., and Cosman, P. (2003). Human body model acquisition and tracking using voxel data. *International Journal of Computer Vision*, 53(3):199–223.

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

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

Mulligan, J., Kelshikar, N., Amd, X., and Daniilidis, K. (2003). Stereo-based environment scanning for immersive tele-presence. *IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Immersive Telepresence*, 14(3):304 – 320.

Plaenkers, R. and Fua, P. (2001). Articulated soft objects for video-based body modeling. In *8th International Conference on Computer Vision, ICCV 2001*, pages 394 – 401.

Poli, R. (2007). An analysis of publications on particle swarm optimisation applications. Technical Report CSM-649, University of Essex, Department of Computer Science.

Poli, R. (2008). Dynamics and stability of the sampling distribution of particle swarm optimisers via moment analysis. *Journal of Artificial Evolution and Applications*, 2008:4–13.

Poli, R., Kennedy, J., and Blackwell, T. (2007). Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57.

Poppe, R., Heylen, D., Nijholt, A., and Poel, M. (2005). Towards real-time body pose estimation for presenters in meeting environments. In *Proceedings of the 13-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2005*, pages 41 – 44.

Press, W. H., Teukolsky, S. A., Vetterling, W., and Flannery, B. (2002). *Numerical Recipes in C++*. Cambridge University Press.

Robertson, C. and Trucco, E. (2006). Human body posture via hierarchical evolutionary optimization. In *British Machine Vision Conference 2006*, pages 999 – 1008.

Robertson, C., Trucco, E., and Ivekovic, S. (2005). Dynamic body posture tracking using evolutionary optimisation. *Electronic Letters*, 41:1370 – 1371.

Salamon, P., Sibani, P., and Frost, R. (2002). *Facts, Conjectures and Improvements for Simulated Annealing*. SIAM Monographs on Mathematical Modeling and Computation.

Schutte, J. F., Reinbolt, J. A., Fregly, B. J., Haftka, R. T., and George, A. D. (2004). Parallel global optimization with the particle swarm algorithm. *International Journal for Numerical Methods in Engineering*, 61(13):2296 – 2315.

Shi, Y. H. and Eberhart, R. C. (1998). A modified particle swarm optimizer. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 69 – 73.

Shoji, K., Mito, A., and Toyama, F. (2000). Pose estimation of a 2d articulated object from its silhouette using a ga. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 3, pages 713 – 717.

Sminchisescu, C. and Triggs, B. (2003). Estimating articulated human motion with covariance scaled sampling. *International Journal of Robotics Research*, 22(6):371–391.

Warren, J. and Schaeffer, S. (2004). A factored approach to subdivision surfaces. *Computer Graphics and Applications*, 24.

Ye, Z. and Liu, Z.-Q. (2005). Genetic condensation for motion tracking. In *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, volume 9, pages 5542 – 5547.