Human Body Posture via Hierarchical Evolutionary Optimization

C. Robertson and E. Trucco
Joint Research Institute on Image and Signal Processing,
School of Engineering and Physical Sciences,
Heriot Watt University, Edinburgh, UK

Abstract

This paper presents an evolutionary approach to estimating upper-body posture from multi-view markerless sequences. We fit a 24-dof skeleton model to sparse 3-D stereo data from an array of cameras. We use a particle swarm optimization algorithm which is intrinsically parallel, can incorporate constraints and does not require motion models. We subdivide the high-dimensional search space based on limb dynamics from application sequences and perform hierarchical fitting from the least to the most uncertain body parts. We show experimentally the advantages of this scheme against non-hierarchical optimization in terms of sharper error decrease. We report results with 3-D scanner data of a model human and noisy, calibrated stereo disparity maps of a real videoconferencing scene.

1 Introduction and motivation

This paper presents an evolutionary approach to estimating upper-body posture from multi-view markerless sequences. Our reference application is *immersive videoconferencing*, which aims to create an impression of presence, or co-location, among a group of participants situated at different geographical locations but meeting in a common, augmented reality space [17, 18, 19]. Vital for presence are 3-D visual cues, e.g., rendering figures consistently with the instantaneous viewer's viewpoint. This requires estimates of the 3-D structure of the scene, which is typically done by multiview disparity analysis (see [1] and references above). Disparity maps (DMs) allow IBR-oriented systems to perform novel view synthesis consistent with the target viewpoint. A major problem is to achieve in real time DMs of the moving human body with sufficiently high quality, as typical immersive environments use large screens. Notice that videoconferencing setups rarely allow high numbers of cameras surrounding the scene [2]; occlusions created by gestures make DMs sparse.

Assuming suitable computing power, *integrating bottom-up disparities with body models* can yield superior quality DMs. Previously, only local DM enhancement has been attempted by infilling [3, 4]. In order to further this idea, we intend to fit human body models to DMs of multi-view videoconferencing sequences. This paper concentrates on the first step, skeleton fitting for posture estimation, and introduces a novel evolutionary approach to the high-dimensional optimization problems typical of body model fitting with markerless sequences.

We fit a 24-dof skeleton model to sparse 3-D stereo measurements, obtained from one or two pairs of cameras. We use *particle swarm optimization* (PSO), a highly-parallel, population-based evolutionary algorithm. We subdivide the high-dimensional search space on the basis of the limb dynamics observed in typical application sequences, then perform hierarchical fitting from the most predictable to the most uncertain body parts. We show experimentally the advantages of this scheme against non-hierarchical optimization in terms of sharper error decrease. To facilitate reproduction of our results, we include pseudocode of the basic and parallel PSO skeleton fitting.

In the remainder of this paper, Section 2 places our work in the context of the recent literature; Section 3 sketches our body model; Section 4 introduces our evolutionary algorithm, and Section 5 its parallel version. Section 6 presents some experimental results, and Section 7 offers some conclusions.

2 Related work

For reasons of space we focus only on important issues relevant for our work.

Body models. Capturing the shape and motion of the human body, a highly articulated object with complex surfaces, calls first of all for an adequate model. The popular *layered models* [5] consist of two parts: an articulated stick figure, or *skeleton*, defining posture, and a *skin model*, defining the body surface. The skeleton is a kinematic chain of limbs (sticks) linked by joints with rotational degrees of freedom subject to anatomical constraints. The skin surface has been variously modeled within computer graphics, e.g., by triangular meshes [21], subdivision schemes [6], and implicit surfaces [20]. Skin elements (e.g., mesh vertices) are connected to skeleton limbs to make the skin surface consistent with posture. *Example-based representations*, as opposed to geometry-based ones, have also been tried in graphics to synthesize novel human shapes or motion by interpolating observed examples from a database of video or scanned material [8, 7, 9].

Computer vision has concentrated mostly on tracking and posture identification, not on photorealistic image generation. Consequently, the common model is an articulated skeleton attached to simple geometric primitives, say truncated cones [10]. No detailed skin model is needed, although meshes are used by some authors [11, 12]. In this work, we adopt a limb-based 24-dof skeleton modelling upper body posture. In this case we have used very simple cylindrical volume primitives, although the addition of more complex surfaces is clearly a straightforward extension, as explained in Section 4.3.

Optimization. Fitting articulated skeletons to human figures leads to high-dimensional optimization problems with complex energy landscapes. *Bayesian approaches*, from classic Kalman filtering [14, 13] to various flavours of particle filtering [15, 16] are the majority, but it is not often easy to incorporate anatomical constraints, parallelize the algorithms, devise suitable observation models as pdf, and devise realistic body motion models.

PSO is intrinsically parallel, can include anatomical constraints (angular intervals) in a straightforward way, and does not require statistical observation models, indeed it requires little *a priori* knowledge of the problem.

Search space partitioning has been used to concentrate search on subspaces, sometimes searched in a special order. Subspaces can simply be predefined grids [21], or "soft partitions" found automatically [15, 11]. Unlike unrestricted full-body movements, video-conferencing scenes contain typically persons sitting with near-stationary trunk, slightly mobile head, and mobile arms. This suggests a well-defined search sequence through the

limb subspaces, and there is no need for automatic partitioning.

Search in Bayesian approaches has been directed to special directions, e.g., by re-scaling covariances, requiring local analysis of the search space [11]. PSO, instead, explores the state space *directly*, and swarm search bypasses the need to steer model parameters towards directions identified by local approximations of the energy landscape. A further feature of solutions reported is *dimensionality reduction*, typically by PCA [15, 10, 8], in order to simplify the very complex energy landscape. PSO does not really need dimensionality reduction thanks to its highly parallel space exploration.

3 Body model

We use a 24-dof skeleton model describing the top half of the body, shown in Figure 1 and outlined in Tables I and II. The constraints on this model are those of a real human, for example the elbow joint has a single degree of freedom restricted in the range $[0, 145]$. The full range of motion envelope is outwith the scope of this paper, however these constraints have been measured directly using goniometric equipment [28] and derived from online databases. Mean limb radii are measured directly from the participants or come from mean values found in [28]. The overall skeleton position is controlled by the first six parameters, describing the root position and rotation. The other limbs and joints are transformed hierarchically by compounding homogeneous co-ordinate system transformations, e.g., for a generic joint:

$$R_T(\vartheta_x, \vartheta_y, \vartheta_z, \mathbf{t_{ab}}) = \begin{pmatrix} R(\vartheta_x, \vartheta_y, \vartheta_z) & \mathbf{t_{ab}} \\ \mathbf{0} & 1 \end{pmatrix}. \quad (1)$$

The compound transformation for a generic joint position is then

$$R_T(\Sigma_w, \Sigma_i) = R_T(\Sigma_w, \Sigma_{root}) \quad R_T(\Sigma_{root}, \Sigma_{clavicle}) \quad \ldots R_T(\Sigma_{i-1}, \Sigma_i) \quad (2)$$

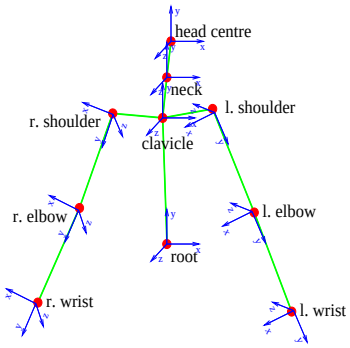


Figure 1: Parameterization of joint positions and coordinate systems

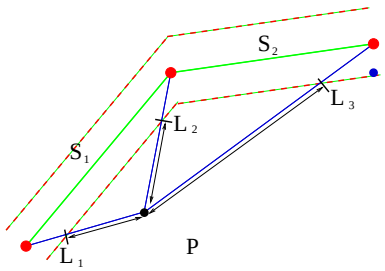


Figure 2: Evaluation function

For these experiments we have used a very simple set of cylindrical volumes along the limb sections, whose radius is the maximum thickness of the limb section as measured from the subject.

Joint	DOF
Root	X, Y, Z
Clavicle	X, Y, Z
Neck	X Y Z
Head	Tip
Clavicle Left Shoulder	X, Z
Left Shoulder	X, Y, Z
Left Elbow	X
Left Wrist	Tip
Clavicle Right Shoulder	X, Z
Right Shoulder	X, Y, Z
Right Elbow	X
Right Wrist	Tip

Table 1: Limbs in the skeleton model

Limb
Root - Clavicle
Clavicle - Neck Top
Neck Top - Head
Clavicle - Left Shoulder
Left Shoulder - Left Elbow
Left Elbow - Left Wrist
Clavicle - Right Shoulder
Right Shoulder - Right Elbow
Right Elbow - Right Wrist

Table 2: Joints in the skeleton model

4 Evolutionary skeleton fitting

4.1 Particle swarm optimization

PSO is a population-based evolutionary optimization technique, introduced by Kennedy and Eberhart [22]. The population is termed a *swarm* and each population member a *particle*. Each particle[1] is a candidate solution or search point in the search space. Unlike other evolutionary methods, each particle has an associated position and velocity. Particles move through the solution space with changing position and velocity, governed both by their historical behaviour as well as the behaviour of their local group. Particles have a tendency to move towards better areas of the search landscape since they are attracted to the population member that has the best overall evaluation score at any given time. One of the attractive elements of PSO for tracking is that useful information is naturally retained in the system from one frame to the next.

4.2 Basic PSO fitting

We have investigated several variations on the original algorithm to achieve a solution particularly well suited to posture identification and tracking in a high-dimensional space. Below is the basic form of the optimization :

1: Population set N of dimension D
2: **for all** $i \in N$ **do**
3: Randomize particle positions, $\underline{p}_i$ and velocities $\vec{v}_i$
4: Initialize best positions as the current start positions, $\underline{b}_i = \underline{p}_i$
5: **end for**
6: **while** Stopping criteria not met **do**
7: **for all** $i \in N$ **do**
8: compute the desired evaluation function, $E_i = f(\underline{p}_i)$
9: **if** $E_i < pbest_i$ **then**
10: $pbest_i = E_i$

[1] Notice that the meaning is different from *particle filtering* since in PSO a particle is a *candidate solution* or a point in state space; in particle fi ltering, a particle is a *weight* attached to a sample of an unknown pdf.

11: $\underline{b}_i = \underline{p}_i$
12: **end if**
13: **end for**
14: Find best particle in the neighbourhood, $\underline{p}_g$
15: **for all** $i \in N$ **do**
16: Update p_i and $\vec{v}_i$ using formulae (2) and (3)
17: **end for**
18: **end while**

$$\vec{v}_{id} = \vec{v}_{id} + c_1.rand()(\underline{b}_{id} - \underline{p}_{id}) + c_2.rand().(\underline{p}_{gd} - \underline{p}_{id}) \tag{3}$$

$$\underline{p}_{id} = \underline{p}_{id} + \vec{v}_{id} \tag{4}$$

Tracking is generally difficult for evolutionary optimization algorithms, PSO included, given its constantly changing evaluation function [23]. This problem has been addressed in several ways: *applying PSO as normal*, in the belief that since convergence is generally fast the system will reconverge as expected [24]; *re-evaluating and resetting the previous best* whenever the evaluation landscape changes, which works well when the inter-frame changes are small [25]; and *re-randomizing* particles when change is detected [26].

When tracking, we assume that instantaneous changes are small in *some* (not all) parameters. This allows us to formulate an optimization scheme where the next state is, at least partially, close to the last. A subset of particles already placed in the right (global-minimum) region of the state space means a high likelihood of converging to the right solution quickly.

4.3 Evaluation Function

We compute the absolute distance of each point in the dataset, $\underline{p}_i, i \in [1,N]$, to the positions pairs describing each limb of the skeleton, $S_k, k \in [1,9]$. Our evaluation function is then

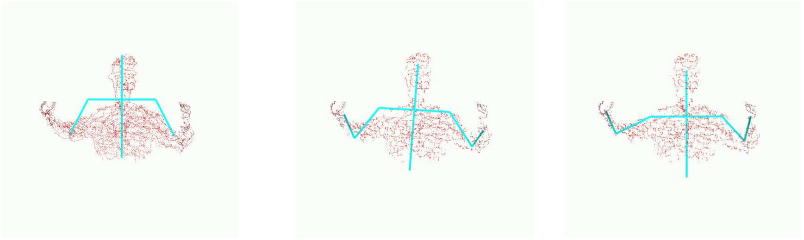
$$E = \sum_{i=1}^{i=N} \min_k [D(\underline{p}_i, S_k)], \tag{5}$$

where $D()$ is a distance function related to the cylindrical surface of each limb, $\underline{p}_i$ is the particle position and S_k is a limb (the line between the limb endpoints). The most efficient function we have used for $D()$ is the sum of distances between pairs of limb positions: e.g., in our example in Figure 2, for the point P, $D(P,S_1) = PL_1 + PL_2$ and $D(P,S_2) = PL_2 + PL_3$. This evaluation is quick to perform for low numbers of limbs. Notice that the model fitting performs, in practice, a run-time segmentation of the data set, according to limbs. Our volume primitives are simple cylinders, as explained in Section 3.

4.4 Hierarchical Fitting

The idea behind hierarchical fitting is that parts moving the least are the most predictable, and their parameters should be determind first as the most reliable ones. This clearly depends on the movements observed, and we refer the reader to the discussion in Section 2. We iteratively increase the number of parameters used in the fit so that at each stage we optimise a superset of the set in the previous stage. Parameters not being optimized are given values from a canonical skeleton pose, a fairly general upright posture. Our

optimization hierarchy is: root position; root orientation; entire trunk, neck and head; shoulder joints; elbow joints. This hierachical protocol is empirically derived from observing the maximum error reduction when performing a non-hierachical fit. Some of the phases are seen working in Figure 3. The efficiency and accuracy of hierarchical fitting is quite pronounced when compared to optimizing all of the parameters simultaneously, especially when tracking, since it is likely that there will be changes in a subset of the parameters only at any one time.



Root position + orientation Shoulders + trunk All

Figure 3: Some phases of the hierarchical fi tting

5 Parallel PSO skeleton fitting

Our use of PSO is motivated by its intrinsically parallel nature. We have designed an efficient parallel version of PSO in order to test the optimization across multiple CPUs.

There are two methods of parallelizing the algorithm over P processors: to maintain P separate search populations, or to maintain a single population split into P parts. The former method implies no inter-swarm communication between the search populations, thus finding P local minima. This method is almost linearly scalable as there is no information interchange. We have adopted the latter method as a single large population is better suited for searching for optima around a known last-best estimate (tracking), with a limited number of particles exploring farther regions of the parameter space.

Using the MPI (message passing interface) parallelization API [27] we have developed the following parallel PSO algorithm. It uses a parameter optimization mask, allowing us to switch selected parameters on or off as required. In each phase, we optimize a superset of parameters from the last phase as outlined in the previous Section. The phase change criteria (line 10) refers to examining the first derivative of the error over several iterations and initiating phase change when it is low, i.e. no further improvements are taking place.

1: Population set N of dimension D on P processors
2: **for all** $j \in P$ **do**
3: **for all** $i \in N$ **do**
4: Randomize particle positions, $\underline{p}_i$ and velocities $\vec{v}_i$
5: Initialize best positions as the current start positions, $\underline{b}_i = \underline{p}_i$
6: Distribute the 3D points to each processor
7: **end for**
8: **end for**
9: **for all** Phases of parameter masks **do**

```
    while Phase change criteria not met on P_0 do
      for all j ∈ P do
        for all i ∈ N do
          compute the desired evaluation function, E_i = f(p_i)
          if E_i < pbest_i then
            pbest_i = E_i
            b_i = p_i
          end if
        end for
      end for
      Find best particle in the neighbourhood, p_g
      Send best particle to processor P_0
      if P_0 then
        Send best particle over the whole swarm to all P_i, i ≠ 0
      end if
      for all j ∈ P do
        for all i ∈ N do
          Update p_i and v_i using formulae (2) and (3) subject to parameters active in this phase
        end for
      end for
    end while
  end for
```

6 Experimental results

We carried out experiments on a dual CPU Intel Xeon (3GHz) running the GNU/Linux OS. We report briefly results with a model human as well as a real videoconferencing scene.

Examples of results. Example data sets and fitted skeletons from a model figure are shown in Figures 4 and 5. The input data is a cloud of 3-D points from a laser scanner developed in our group using neural calibration [29]. Each fitting has between 500 and 1,000 points and we have tested swarms from many hundreds to a few tens of particles. The examples shown here typically require between 25 and 200 particles and there is a trade-off between speed of execution and quality of final fitting. To take a specific example, the data and results in Figure 4(a) has a convergence profile as shown in Table 3. This Table also has the following features: the number of points used is around 600; by completion we mean that the rate of error change is under 1.0. As an indication, however, if we immediately change the data after convergence to another posture we have found that convergence is significantly faster, often requiring only a few iterations.

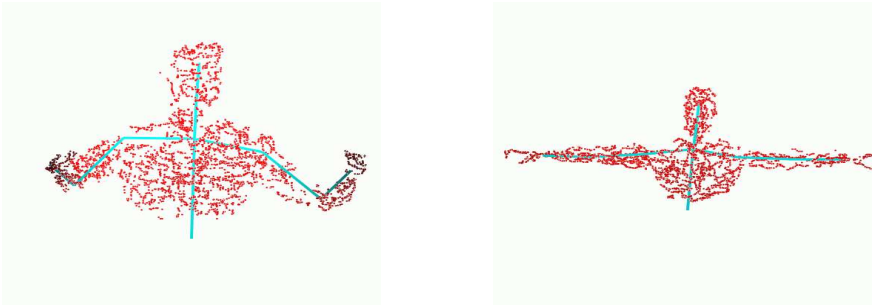
Particles	Convergence iterations	Convergence time (sec)	Time for full run (sec)
200	20	11.8	67.1
100	20	7.0	43.2
50	30	6.7	29.4
25	65	8.7	20.4

Table 3: Typical convergence results

Figure 5 (bottom row) shows the skeleton fitted to a texture-mapped, severely noisy,

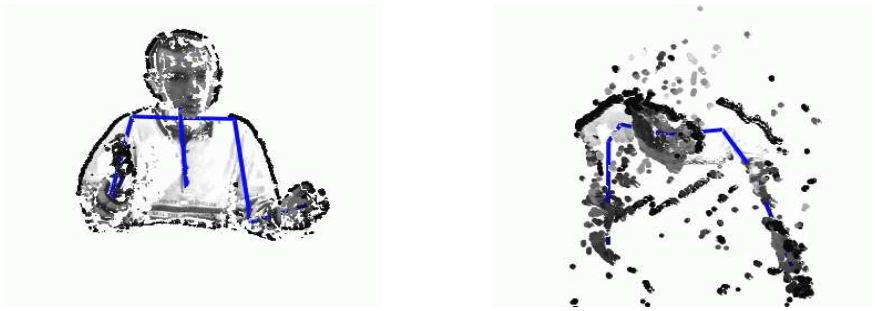
raw disparity map of a real videoconferencing frame, again using a few thousand particles and in this case around 3,000 points. The disparity map was computed by a pyramidal correlation-based algorithm. Notice the holes (no-data regions) and severe erroneous results withstood by the fit.

We have found that the minimum number of particles to ensure stable fitting with randomized initialization is 50 although there is a trade-off between speed and accuracy. As with all evolutionary techniques we recommend integrating local search into the algorithm (thus creating a so-called *memetic* algorithm) to both speed up initialization and final fit, although discussion of this is beyond the scope of this paper.



(a) Fitted skeleton in generic pose **(b)** Fitted skeleton in crucifix pose

Figure 4: Two examples of fitted skeletons



(a) Real teleconferencing scene with skeleton overlaid **(b)** Real teleconferencing scene with skeleton overlaid (in plan)

Figure 5: Example fittings

Hierarchical fitting. The benefits of our hierarchical scheme are exemplified by the performance graph in Figure 6. Here we used around 1,000 data points and 200 particles. The phases of hierarchical switching are denoted by dotted vertical lines and the effect of switching on the various parameter sets is evident: in each region error decrease is accelerated. Also plotted for comparison is an error graph for ordinary (non-hierarchical) fitting which reaches a plateau (a local minimum in fact) after around 40 iterations. From our experience, this behaviour is characteristic: with a large degree of mobility, the arms have a tendency to accumulate too many points in the segmentation and become 'trapped'. In our experience, this effect is removed completely by hierarchical fitting.

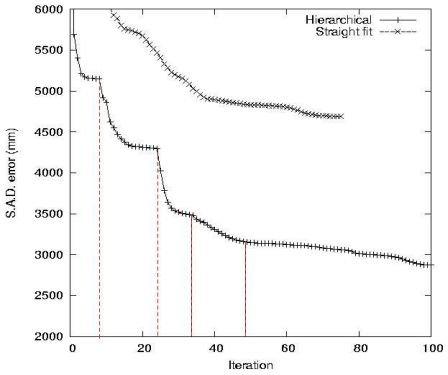


Figure 6: Typical hierarchical fi tting errors versus iterations. Note the fi ve hierachical phases.

7 Conclusions

We have presented a novel algorithm fitting a high-dimensional skeleton to markerless upper-body sequences. The algorithm is intrinsically parallel, can include anatomical constraints in a straightforward way, and does not require statistical observation models. Our parallel version of PSO implements a hierarchical scheme based on the dynamic properties of videoconferencing speakers. The technique has excellent potential for tracking a continuously moving set of 3D data in real time on a modest number of processors, although real time can also be achieved by reducing the number of particles and/or that of 3-D data points. Our initial prototype achieved good results with clouds of points of model figures obtained from a laser scanner, as well as with noisy, calibrated DMs from real videconferencing sequences. Further investigations include setting optimal, general system parameters (i.e., with different speakers, different motion styles, min particle numbers); combining several, co-operating trackers analysing simultaneously data collected from multiple sources; fast bootstrapping; and run-time joint length optimization.

Acknowledgements

This work was carried out under EPSRC grant number GR/T11890/01.

References

[1] A Criminisi, J Shotton, A Blake and P Torr, Gaze Manipulation for One-to-One Videoconferencing, Proc. Int. Conf. on Comp. Vision ICCV 03, 191–198.

[2] T Kanade, P Rander and R Narananyan, Virtualized reality: constructing virtual world from real scenes, IEEE Trans. Multimedia, 4(1):34–46, 1997

[3] S. Ivekovic, E. Trucco, Dense Wide-Baseline Disparities from Conventional Stereo for Immersive Videoconferencing, Proceedings of the 17th International Conference on Pattern Recognition, 2004, Volume 4, pp. 921-924

[4] N. Chang and A. Zakhor, View generation for three-dimensional scenes from video sequences, IEEE Trans Image Process. 6:584–598, 1997.

[5] Collins, G. and Hilton,A, Modelling for Character Animation, Software Focus, 2(2): 44–51, 2001.

[6] B. Allen, B. Curless, Z. Popovic Articulated Body Deformation from Range Scan Data, ACM Transactions on Graphics, 21(3):61619, 2002

[7] B Allen, B Curless, Z Popovic, The space of human body shapes: reconstruction and parameterization from range scans, ACM Trans Graphics (TOG), 22(3), 2003.

[8] L Molina Tanco and A Hilton, Realistic Synthesis of Human Body Movements from a Database of Motion Capture Examples, Proc. IEEE Workshop on Human Motion, 2000.

[9] R. Urtasun, P. Glardon, R. Boulic, D. Thalmann and P. Fua, Style-based Motion Synthesis, Comp. Graph. Forum, 23(4):1–14, 2004.

[10] H Sidenbladh, M Black and L Sigal, Implicit Probabilistic Models of Human Motion for Synthesis and Tracking, Proc ECCV 02, 784–800, 2002.

[11] C Sminchisescu and B Triggs, Covariance Scaled Sampling for Monocular 3-D Body Tracking, Proc IEEE CVPR 01, 2001.

[12] S. Ivekovic, E. Trucco. Human Body Pose Estimation with PSO, Proceedings of IEEE Congress on Evolutionary Computation (CEC) 2006, WCCI 2006, Vancouver, Canada,

[13] C Bregler, J Malik, and K Pullen, Twist based Acquisition and Tracking of Animal and Human Kinematics, Int. Journal of Computer Vision, 56(3), 179–194, 2004.

[14] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman, Human body model acquisition and tracking using voxel data, Int Jour Comp Vision, 53(3):199–223, 2003.

[15] J Deutscher and I Reid, Articulated Body Motion Capture by Stochastic Search, Int Jour Comp Vision, 61(2):185-205, 2005.

[16] R Fablet and M Black, Automatic Detection and Tracking of Human Motion with a View-Based Representation, Proc ECCV 02, 476–491, 2002.

[17] E. Trucco, C. Plakas, N. Brandenburg, P. Kauff, M. Karl, O. Schreer, Real-Time Disparity Analysis for Immersive 3-D Teleconferencing, Proc Workshop on Video Registration, ICCV01, Vancouver, 2001.

[18] Baker, H. Harlyn; Tanguay, Donald; Sobel, Irwin; Gelb, Dan; Goss, Michael E.; Culbertson, W. Bruce; Malzbender, Thomas, The Coliseum Immersive Teleconferencing System, *Proc. International Workshop on Immersive Telepresence*, 6 December 2002, Juan Les Pins, France.

[19] H. Towles, National Tele-Immersion Initiative: Sharing Presence with Distant Colleagues, Presentation given at Internet2 Spring Meeting held in Washington DC, March 7-9, 2001.

[20] R. Plankers and P. Fua, Tracking and modeling people in video sequences, Computer Vision and Image Understanding, Volume 81, Number 3, March 2001, pp. 285-302(18).

[21] J. Carranza, C. Theobalt, M. A. Magnor, H. Seidel, Free-viewpoint video of human actors, ACM Trans. Graph., **22:3**, 2003.

[22] J. Kennedy and R. Eberhart, Particle Swarm Optimization, *Proc. IEEE International Conference on Neural Networks*, 1995, Perth, Australia.

[23] Y. Shi and R. Eberhart, Tracking and optimizing dynamic systems with particle swarms, *Proc. Congress on Evolutionary Computation 2001*, Seoul, Korea, 2001.

[24] K. E. Parsoupolos and M. N. Vrahatis, Particle swarm optimizer in noisy and continuously changing environments, *Artifical Intelligence and Soft Computing*, pp 289-294, IASTED Press, 2001.

[25] A. Carlisle and G. Dozier, Adapting PSO to dynamic environments, *Proc. of International Conference in Artificial Intelligence*, pp 429-434, Las Vegas, Nevada, 2000.

[26] X. Hu and R. Eberhart, Adaptive particle swarm optimization: detection and response to dynamic systems, *Proc. IEEE Congress on Evolutionary Computation*, Honolulu, Hawaii, 2002.

[27] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI 2nd edition*, MIT Press, 1999; ISBN 0-262-57132-3.

[28] C. Robertson, E. Trucco, An Ergonomically Constrained Upper Body Model, Heriot-Watt Converse Group Technical Report.

[29] C. Robertson, E. Trucco, Extrapolative Range Sensing Using Direct Neural Stereo Calibration, Heriot-Watt Converse Group Technical Report.