Real-Time Multi-View Human Motion Tracking Using 3D Model and Latency Tolerant Parallel Particle Swarm Optimization

Bogdan Kwolek[1,2], Tomasz Krzeszowski[2,1], and Konrad Wojciechowski[2]

[1] Rzeszów University of Technology
W. Pola 2, 35-959 Rzeszów, Poland
bkwolek@prz.edu.pl

[2] Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008 Warszawa, Poland
bytom@pjwstk.edu.pl

Abstract. This paper demonstrates how latency tolerant parallel particle swarm optimization can be used to achieve real-time full-body motion tracking. The tracking is realized using multi-view images and articulated 3D model with a truncated cones-based representation of the body. Each CPU core computes fitness score for a single camera. On each node the algorithm uses the current temporary best fitness value without waiting for the global best one from cooperating sub-swarms. The algorithm runs at 10 Hz on eight PC nodes connected by 1 GigE.

1 Introduction

Markerless 3D human motion tracking is an important problem in computer vision due to many potential applications, including, but not limited to, visual surveillance, recognizing human activities, clinical analysis and sport (biomechanics) [10]. Commercial systems for human motion capture are typically based on optical or magnetic markers and usually require laboratory environment and the attachment of markers on the body segment being analyzed. Thus, a technique for articulated human motion tracking that does not need markers attached to body would greatly extend the applicability of the motion capture.

Tracking articulated motion is difficult task because of generally unpredictable nature of human movements, high variability of human appearance, self-occlusions and depth ambiguities. The high-dimensional non-linear search space and the exponentially increasing computational overload are the main challenges in full articulated body tracking on the basis of markerless techniques. Three dimensional model based methods are generally more accurate in comparison to methods relying on learned mapping between pose exemplars and a set of image features. An articulated human body can be perceived as a kinematic chain consisting of at least eleven parts corresponding to body parts. Typically such a 3D human model consists of very simple geometric primitives like cylinders or truncated cones. On the basis of such geometrical primitives a lot of hypothetical body poses are generated and after projecting to the image space are compared with real images through a likelihood function.

Particle filtering is one of the most important and common tracking algorithms in non-intrusive human motion capture. In a particle filter each sample represents some hypothesized body pose. For a 3D model consisting of eleven geometric primitives we need around 26 parameters to describe the full body articulation. That means that tracking full articulated body is very computationally demanding task. For instance, in [2] processing for 5 seconds long video took about one hour using a particle filter with 200 samples and 10 annealing layers. In more recent work [6], the processing time of *Lee walk* sequence from Brown University is larger than one hour. Several attempts were proposed to mitigate the inherent limitations of particle filtering such as degeneracy, loss of diversity and course of dimensionality. Recently, Particle Swarm Optimization (PSO) was proposed as an alternative of particle filtering for full-body articulated motion tracking [5][8][13]. Some work has also been done in order to achieve real-time articulated body tracking [12][8].

In this work we propose a communication latency tolerant parallel algorithm for PSO based articulated motion tracking. The algorithm consists of multiple swarms that are executed in parallel on multiple computers connected via a peer-to-peer network. The computers exchange information about the location of the best particle and its corresponding fitness function of a sub-swarm. Next to each optimization iteration, information about the global particle location and the corresponding fitness score is sent asynchronously without blocking the sending thread. The message contains also data about the frame number and the iteration number. The computers receive the data in a separate thread. On the basis of arriving data the receiving threads are responsible for determining the best particles for each frame and each iteration. The best values are stored in a mutually exclusive memory. After each iteration, the processing thread checks if its global particle is better than the particle sent via other computers. If yes, it updates its own best particle and continues the optimization.

The contribution of our work is a parallel particle swarm optimization algorithm for real-time object tracking. The novelty of our work lies in the asynchronous exchange mechanism for the best particle location and its fitness score during the multiple calls of particle swarm optimization, which take place during object tracking. This results in a communication latency tolerant parallel algorithm for object tracking. The algorithm is fast and affective because it strongly relies on the stochastic nature of Particle Swarm Optimization algorithm. In particular, a sub-swarm, which as a first one finished tracking of the object in a given frame, it carries out the rediversification of the particles using its current global best particle, without waiting for the best locations of the remaining sub-swarms. In such circumstances the algorithm takes the best locations of the cooperating sub-swarms from the previous iterations, which were determined for the considered frame. The algorithm has been evaluated in multi-view based markerless full-body tracking. The tracking can be done at real-time frame rates using ordinary network of peers consisting of multi-core PCs.

2 Relevant Work

PSO was applied in a number of areas as a technique to solve large, non-linear optimization problems [11]. The applications of PSO in computer vision and graphics are still rather limited. The main applications of PSO in computer vision are connected with non-articulated object tracking. For example, [14] shows that in tasks consisting in tracking human face a variant of PSO, called sequential PSO behaves better than a particle filter in terms of tracking accuracy.

Existing algorithms for articulated motion tracking can be roughly divided into two categories, namely, discriminative and generative [9]. Discriminative approaches attempt to learn a direct mapping between image descriptors, such as edges or shapes to the 2D human pose. A major limitation is that their performance is considerably lower in circumstances in which is difficult to obtain reliable features, for instance in the cluttered scenes. Generative approaches generate a number of plausible pose hypotheses, which are then evaluated against the current image for evidence. The pose hypotheses are generated on the basis of a 3D model of the human body. Such a model is projected onto an image plane and an error function is calculated to indicate the quality of the match. The mentioned approaches are based on a rather coarse 3D models of the human body. In methods introduced in [3][4], realistic human body models were developed to accomplish tracking through analysis-by-synthesis. In such an approach the texture mapping is used to obtain a precise textured model of the person.

Very recently, PSO has been successfully applied to full-body articulated motion tracking [5][8][13]. In [5], the articulated pose is estimated through a hierarchical search. The articulated human body model is represented as a 3-D kinematic tree consisting of 13 nodes. The experiments were performed on *Lee walk* sequence, which was downsampled at frame rate of 30 Hz. On images of size 640×480 the average error distance between estimated pose and ground-truth pose is larger than 50 mm, whereas the processing time of the sequence with 75 images is larger than one hour. The above mentioned sequence has also been used in [13]. The average error on 15 virtual markers is about 40 mm. Our work differs from theirs in a number of ways, of which the most crucial is the focus on full body motion tracking in real-time. To the best of our knowledge, ours is the first near real-time system that is able to accomplish full-body articulated motion tracking. The quality of tracking on various number of computers was compared by analyses carried out both through qualitative visual evaluations as well as quantitatively through the use of the motion capture data as ground truth. The preliminary results demonstrate that the tracking accuracy is in the same range as the accuracy in work mentioned above.

Some parallel PSO algorithms were proposed to speed-up the optimization of complex engineering optimization problems but, to the best of our belief, so far, no parallel PSO algorithm for object tracking has been proposed. In particular, our algorithm executes not only the PSO iterations in parallel in a given frame, but being latency tolerant and asynchronous it starts processing the next frame without waiting for all best locations of the cooperating sub-swarms.

3 3D Body Model and Cost Function

The skeleton of the human body is modeled as a kinematic tree. The articulated 3D model consists of eleven segments with the limbs represented by truncated cones, which model the pelvis, torso/head, upper and lower arm and legs. The configuration of the model is defined by 26 DOF. It is parameterized by position and orientation of the pelvis in the global coordinate system and the relative angles between the connected limbs. In order to obtain the 3D human pose each truncated cone is projected into 2D image plane via perspective projection. In such a way we obtain an image with the rendered model in a given configuration. Such image features are then matched to the person extracted by image analysis.

The fitness function consists of two components: $f(x) = w_1 f_1(x) + w_2 f_2(x)$, where w_i stands for weighting coefficients that were determined experimentally. The function $f_1(x)$ reflects the degree of overlap between the body parts and the projected segments of the model into 2D image. It is expressed as the sum of two components. The first component is the overlap between the binary image and the considered rasterized image of the model. The second component is the overlap between the rasterized image and the binary one. The larger the degree of overlap is, the larger is the fitness value. The function $f_2(x)$ is calculated on the basis of distance transform based Chamfer matching.

4 Latency Tolerant Parallel PSO for Object Tracking

Particle swarm optimization is a population based optimization technique, which is stochastic in nature and makes use of the memory of each particles as well as the knowledge gained by the swarm as a whole. In the ordinary PSO algorithm the update of particle velocity and position is given by the following equations:

$$v_j^{(i)} \leftarrow w v_j^{(i)} + c_1 r_{1,j}^{(i)} (p_j^{(i)} - x_j^{(i)}) + c_2 r_{2,j}^{(i)} (p_{\mathrm{g},j} - x_j^{(i)}) \tag{1}$$

$$x_j^{(i)} \leftarrow x_j^{(i)} + v_j^{(i)} \tag{2}$$

where w is the positive inertia weight, $v_j^{(i)}$ is the velocity of particle i in dimension j, $r_{1,j}^{(i)}$ and $r_{2,j}^{(i)}$ are uniquely generated random numbers with the uniform distribution in the interval [0.0, 1.0], c_1, c_2 are positive constants, $p^{(i)}$ is the best position found so far by particle i, p_{g} denotes a best position, which can be:

- a global best that is immediately updated when a new best position is found by any particle in the swarm
- neighborhood best where only a specific number of particles is affected if a new best position is found by any particle in the sub-population

A topology with the global best converges faster as all the particles are attracted simultaneously to the best part of the search space. Neighborhood best allows parallel exploration of the search space by multi-swarm. Such configuration decreases the susceptibility of falling into local minima, however, it typically slows down the convergence speed.

The equation (1) has three main components. The first component, referred to as inertia, models the particle's tendency to continue the moving in the same direction. Thus, it controls the exploration of the search space. The second component, called cognitive, attracts towards the best position $p^{(i)}$ previously found by the particle. The last component is referred to as social and attracts towards the best position p_{g}. The fitness value that corresponds to $p^{(i)}$ is called local best $p_{\mathrm{best}}^{(i)}$, whereas the fitness value corresponding to p_{g} is referred to as g_{best}.

The PSO is initialized with a group of random particles (hypothetical solutions) and then it searches hyperspace (i.e. R^n) of a problem for optima. Particles move through the solution space, and undergo evaluation according to some fitness function f. Much of the success of PSO algorithms comes from the fact that individual particles have tendency to diverge from the best known position in any given iteration, enabling them to ignore local optima, while the swarm as a whole gravitates towards the global extremum. If the optimization problem is dynamic, the aim is no more to seek the extrema, but to follow their progression through the space as closely as possible. Since the object tracking process is a dynamic optimization problem, the tracking can be achieved through incorporating the temporal continuity information into the traditional PSO algorithm. This means, that the tracking can be accomplished by a sequence of static PSO optimizations to determine the best person's pose, which are followed by re-diversification of the particles to cover the possible state in the next time step. In the simplest case, the re-diversification of the particle i can be done as follows:

$$x_t^{(i)} \leftarrow \mathcal{N}(\hat{x}_{t-1}, \Sigma) \tag{3}$$

where $\hat{x}_{t-1}$ is the state estimate in time $t-1$. In the global best configuration the estimate $\hat{x}_{t-1}$ is equal to p_{g} determined in the last iteration. In the configuration with neighborhood best it is selected as the best position of any sub-swarm.

PSO is parallel in nature. To shorten the optimization time several studies on parallelizing the algorithm were done so far. In general, two parallelization strategies are considered, namely synchronous and asynchronous. In the synchronous algorithm at the end of each iteration all nodes communicate with each other to determine the global best fitness. In asynchronous parallelization the particles use the current temporary best fitness without waiting for the global best one. However, up to now all of the published literature reported parallel PSO algorithms for *static optimization*, where the particles are evaluated and evolved in parallel in several iterations until the global extremum is found out.

The latency tolerant parallel PSO uses asynchronous exchange mechanism for the best particle location and its fitness score during the multiple calls of particle swarm optimization, which take place during object tracking. In particular, subsequent to each iteration no barrier synchronization is executed as the algorithm strongly relies on the stochastic nature of PSO. Particularly, if a sub-swarm, which as a first one finished object tracking in a given frame, it carries out the rediversification of the particles using its current global best particle, without waiting for the global best optimum determined by the participating sub-swarms. It is worth mentioning that in such circumstances the estimate of the object state is determined using the global best locations of cooperating sub-swarms, which were available during determining in each iteration the global best location of the considered population. After each optimization iteration, information about the global particle location and the corresponding fitness score is sent asynchronously without blocking the sending thread. The frame number and the iteration number are included in the message for a control mechanism aiming at processing the same frame by all computers without large inter-frame delays. The threads receive the data in a separate thread. On the basis of arriving data the receiving threads are responsible for determining the best particles for each frame and iteration. The best values are stored in a mutually exclusive memory. After each iteration, the processing thread checks if its global particle is better than the particle sent via other computers. If yes, it updates its own best particle and continues the optimization.

5 Experimental Results

The algorithm was tested in two multi-camera systems consisting of synchronized and calibrated cameras. The first system consists of two calibrated and synchronized cameras. It acquires images of size 640×480 at frame rate of 15 Hz. Figure 1 depicts sample images that were acquired by the cameras. At the figure we can also see the projected and overlaid model on both input images.

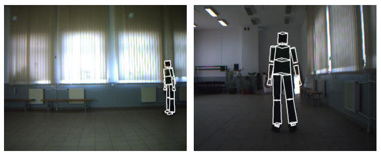

Fig. 1. Human motion tracking using two cameras. The images illustrate the initial model configuration overlaid on the image in first frame.

In the second system the images were captured by four calibrated and synchronized cameras acquiring images of size 1920×1080 with rate 24 fps. Each pair of the cameras is approximately perpendicular to the other two, see the placement of video cameras in Fig. 2. A commercial motion capture (moCap) system from Vicon Nexus provides ground truth motion of the body at rate of 100 Hz. The system uses reflective markers and sixteen cameras to recover the 3D position of such markers. The synchronization between the moCap and multi-camera system is based on hardware from Vicon Giganet Lab. The digital cameras are capable to differentiate overlapping markers from each camera's view.

The precision of human motion tracking was evaluated experimentally in scenarios with a walking person. The analysis of gait is currently an active research

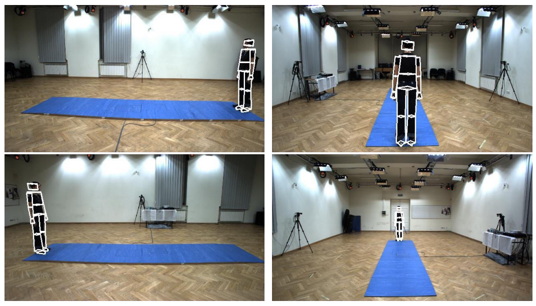

Fig. 2. Layout of the laboratory with four cameras. The images illustrate the initial model configuration, overlaid on the image in first frame and seen in view 1 and 2 (upper row), and in view 3 and 4 (bottom row).

area due to various applications in medicine, surveillance, etc. Although we focused on tracking of torso and legs, we also estimated the pose of both arms as well as of the head. The body pose is described by position and orientation of the pelvis in the global coordinate system as well as relative angles between the connected limbs. The results obtained on various number of computers were compared by analyses carried out both through qualitative visual evaluations as well as quantitatively by the use of the motion capture data as ground truth.

Figure 3 shows results obtained in the two camera system. The left images in each image pair depict the projected and overlaid model on the image from the first camera, whereas the right images are from the second one. The initialization of the system was done manually through fitting the 3D model onto the images, see Fig. 1. The tracking was done using 300 particles and 10 iterations.

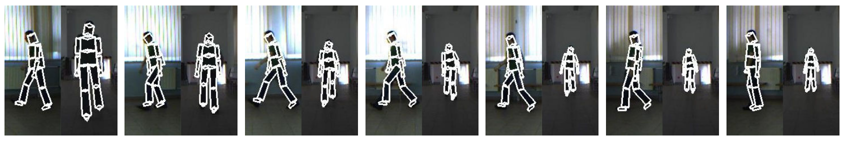

Fig. 3. Articulated 3D human body tracking in two camera setup. Shown are results in frames #10, 20, 30, 40, 50, 60, 70. The left sub-images are seen from view 1, whereas the right ones are seen from view 2.

Figure 4 demonstrates some results that were obtained in the four camera system. The quality of tracking is illustrated using images from first and second camera. The initialization of the tracking was done manually. Optionally, the tracking can be initialized on the basis of data from the moCap system. The same number of particles and iterations was utilized as in the previous experiment. In all experiments on image sequences from the four camera system we used images of size 480×270.

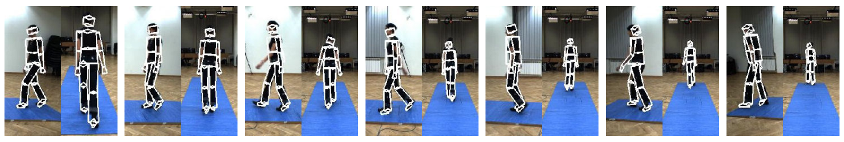

Fig. 4. Articulated 3D human body tracking in four camera setup. Shown are results in frames #20, 40, 60, 80, 100, 120, 140. The left sub-images are seen from view 1, whereas the right ones are seen from view 2.

Figure 5 depicts the errors that were obtained during motion tracking using one and two desktop computers. The experiments were done on image sequences from the four camera system. The errors of tracking the head, torso and knee were calculated using moCap data as ground truth. In optimizations we used 300 particles and 10 iterations. In the configuration consisting of two computers the optimizations were achieved using 150 particles on each computer. As we can observe in the plots shown at Fig. 5, the difference between error estimates obtained by the ordinary algorithm and the parallel algorithm running on two computers is not significant.

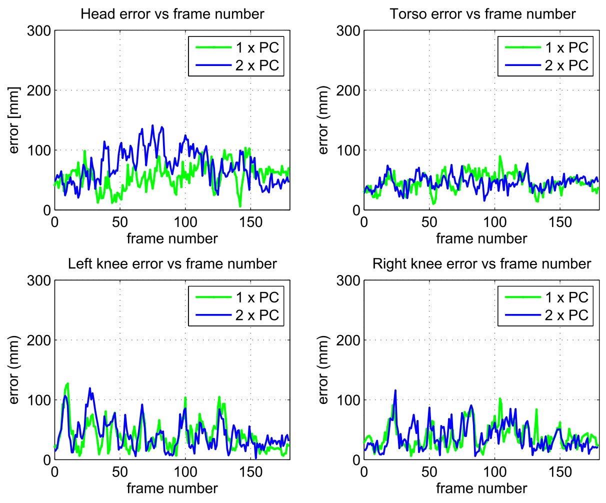

Fig. 5. Tracking errors [mm] versus frame number at 1 and 2 PCs.

In Fig. 6 are shown the error estimates that were obtained on single and eight computers. In a PC cluster with 8 nodes the optimizations were performed using 38 particles on each computer. As we can observe, the average error is far below 90 mm. It is worth noting here that something better results can be obtained using our GLPSO (Global-Local PSO) algorithm [7].

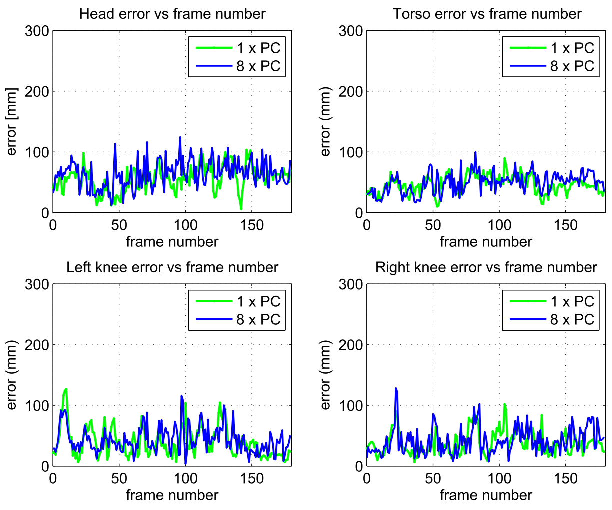

Fig. 6. Tracking errors [mm] versus frame number at 1 and 8 PCs.

The experiments were conducted on desktop PCs with 4 GB RAM, Intel Core i5, 2.8 GHz. All measurements were conducted on a cluster that was composed of identical machines connected with a TCP/IP 1 GigE (Gigabit Ethernet) local area network. The parallelization of the code was done using OpenMP directives. The parallel computations were realized on multi-core (4-core) CPUs.

Currently, OpenMP is widely utilized standard for parallelizing programs in a shared memory environment [1]. It consists of a set of directives (pragmas) and library routines that can be inserted into Fortran or C/C++ codes to enable use of more than one thread. OpenMP provides a fork-and-join execution model in which a program begins execution as a thread. The thread executes sequentially until a parallelization directive for a structured block of code is found. If this takes place, such a thread creates a set of threads and becomes the master thread of the new group of threads. Each thread executes the same code redundantly until the end of the parallel section and the threads communicate by sharing variables. The exit point of a structured block is an implicit synchronization point for the master thread and the threads created for the block. After the synchronization the master thread continues with the computation and the other threads end. In our system each CPU core is responsible for calculation of the fitness function for single camera.

Table 1 shows computation times and speeds-up that were obtained on our PC cluster. The depicted times are needed to extract single model configuration using images from four camera views. In the experiments we focused on efficiency of parallel particle swarm optimization algorithm and therefore the computation times do not comprise the image processing. The image processing was done in advance and all images needed to compute the fitness score were stored on local hard drives. It is worth mentioning that time needed for image processing is about 20% of the total processing time. Moreover, the code of image processing can be easily parallelized. Currently, the communication between the PC nodes is realized using popular QT library. As we can see, the speed-up of our latency tolerant parallel PSO is considerable. Using a cluster consisting of 8 PCs and PSO with 300 particles and 10 iterations the human motion tracking can be done at about 10 fps. The tracking time of blocking version of the parallel PSO is considerably larger in comparison to our latency tolerant algorithm. When images from two camera system are used, we can perform full-body motion tracking together with image preprocessing in real-time with 10 fps.

Table 1. Tracking time [ms] for single human pose (computed on the basis of images from 4 camera views) and speed-up.

		Latency tolerant		Blocking	
#PCs	#particles	time [ms]	speed-up	time [ms]	speed-up
1	300	635.7	-	635.7	-
2	2×150	339.6	1.87	370.4	1.72
3	3×100	227.1	2.80	257.5	2.47
4	4×75	173.7	3.66	202.2	3.14
6	6×50	123.7	5.14	146.5	4.34
8	8×38	96.9	6.56	110.8	5.74

In Tab. 2 are depicted the average errors that were obtained during a computations on different numbers of computers. The pose error in each frame was determined on the basis of $M = 39$ markers $m_i(x) \in R^3, \quad i = 1, \ldots, M$ expressing locations in the world coordinates. The pose error was expressed as the average Euclidean distance:

$$E(x, \hat{x}) = \frac{1}{M} \sum_{i=1}^{M} ||m_i(x) - m_i(\hat{x})|| \tag{4}$$

where $m_i(x)$ denotes for marker's position that was calculated using the estimated pose, whereas $m_i(\hat{x})$ stands for the position that was determined using data from our motion capture system. From the above set of markers, four markers were placed on the head, seven markers on each arm, 6 on the legs, 5 on the torso and 4 markers were attached to the pelvis. Given the discussed placement of the markers on the human body the corresponding virtual marker's were assigned on the 3D model. The position of such virtual markers was determined for each estimate of the human pose and then employed in calculating the average Euclidean distance expressed by (4). The ground truth was extracted on the basis of data stored in `c3d` files. Finally, the average errors shown in Tab. 2 were calculated on the basis of the following equation:

$$Err(x,\hat{x}) = \frac{1}{LM}\sum_{k=1}^{L}\sum_{i=1}^{M} ||m_i(x) - m_i(\hat{x})|| \tag{5}$$

where L denotes the number of frames in the utilized test sequences. The discussed results were obtained on $L = 180$ images, see also Fig. 4, and averaged over 5 runs of the algorithm. As we can observe, for a configuration with multiple nodes the average error is smaller than the error obtained on a single node. This means that multiple swarms PSO can generate better results in comparison to PSO based on single swarm.

Table 2. Average errors [mm].

#PCs	#particles	error [mm]	std. dev. [mm]
1	300	75.8	45.8
2	2×150	72.2	38.1
3	3×100	71.9	34.1
4	4×75	74.3	40.3
6	6×50	73.9	37.9
8	8×38	72.7	36.9

The complete human motion capture system was written in C/C++. One of the future research directions of the presented approach is to explore multiple GPUs to further shorten the processing time [8].

6 Conclusions

We have presented communication latency tolerant parallel algorithm for particle swarm optimization. We demonstrated experimentally that the parallel PSO is especially well suited for real-time full-body articulated object tracking. To show its advantages we have conducted several experiments on walking sequences and realized computations on different numbers of computers connected with a TCP/IP 1 GigE local area network. The quality of tracking was compared by analyses carried out both through qualitative visual evaluations as well as quantitatively through the use of the motion capture data as ground truth.

Acknowledgment

This paper has been supported by the project “System with a library of modules for advanced analysis and an interactive synthesis of human motion” co-financed by the European Regional Development Fund under the Innovative Economy Operational Programme - Priority Axis 1.

References

1. Chapman, B., Jost, G., van der Pas, R., Kuck, D.: Using OpenMP: Portable Shared Memory Parallel Programming. The MIT Press (2007)
2. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. In: IEEE Int. Conf. on Pattern Recognition. pp. 126–133 (2000)
3. Gavrila, D.M., Davis, L.S.: 3-D model-based tracking of humans in action: a multi-view approach. In: Proc. of the Int. Conf. on Computer Vision and Pattern Rec. pp. 73–80. CVPR ’96, IEEE Computer Society, Washington, DC, USA (1996)
4. Grard, P., Gagalowicz, A.: Human body tracking using a 3D generic model applied to golf swing analysis. In: Int. Conf. on Computer Vision / Computer Graphics Collaboration Techniques and Applications (2003)
5. Ivekovic, S., John, V., Trucco, E.: Markerless multi-view articulated pose estimation using adaptive hierarchical particle swarm optimisation. In: Proc. of Applications of Evolutionary Computation. (EvoApplications 2010). pp. 241–250. Lecture Notes in Computer Science, vol. 6024, Springer-Verlag (April 2010)
6. John, V., Trucco, E., Ivekovic, S.: Markerless human articulated tracking using hierarchical particle swarm optimisation. Image Vis. Comput. 28, 1530–1547 (2010)
7. Krzeszowski, T., Kwolek, B., Wojciechowski, K.: Model-based 3D human motion capture using global-local particle swarm optimizations. In: Int. Conf. on Computer Recognition Systems. pp. 297–306. AISC 95, Springer-Verlag (2011)
8. Krzeszowski, T., Kwolek, B., Wojciechowski, K.: GPU-accelerated tracking of the motion of 3D articulated figure. In: Proc. of the Int. Conf. on Computer Vision and Graphics: Part I. pp. 155–162. LNCS, Springer-Verlag (September 2010)
9. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Comput. Vis. Image Underst. 104, 90–126 (2006)
10. Muendermann, L., Corazza, S., Andriacchi, T.: The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications. Journal of Neuroengineering and Rehabilitation 3(1) (2006)
11. Poli, R.: Analysis of the publications on the applications of particle swarm optimisation. J. Artif. Evol. App. 2008, 4:1–4:10 (January 2008)
12. Schmidt, J., Fritsch, J., Kwolek, B.: Kernel particle filter for real-time 3D body tracking in monocular color images. In: IEEE Int. Conf. on Face and Gesture Rec., Southampton, UK. pp. 567–572. IEEE Computer Society Press (2006)
13. Zhang, X., Hu, W., Wang, X., Kong, Y., Xie, N., Wang, H., Ling, H., Maybank, S.: A swarm intelligence based searching strategy for articulated 3D human body tracking. In: IEEE Workshop on 3D Information Extraction for Video Analysis and Mining in conjuction with CVPR. pp. 45–50. IEEE (2010)
14. Zhang, X., Hu, W., Maybank, S., Li, X., Zhu, M.: Sequential particle swarm optimization for visual tracking. In: IEEE Int. Conf. on CVPR. pp. 1–8 (2008)