1. Introduction

During the past three decades, important efforts in Computer Vision research have been focused on visual human motion analysis [30,47,4,15]. Broadly speaking, the goal was first set to generate quantitative descriptions about where human motion was being performed. Nowadays, the analysis of video sequences incorporates cognitive processes which allow understanding the observed motion. That is, the goal is now set to generate qualitative descriptions about the meaning of motion, i.e. about what, how and even why motion is being detected.

To achieve the aforementioned goals, the human sequence evaluation (HSE) scheme [17] defines the information transformation processes involved in going from the low-level sensor data, to the high-level interpretation of human motion. Hence, a key part of the HSE scheme is the transformation process from monocular image sequences to a 3D representation of the human body, enabling the subsequent generation of qualitative descriptions from the motion being observed. Towards this end, this work aims to recover the 3D human body motion parameters from a monocular image sequence.

The main challenges involved in full-body 3D tracking from a monocular image sequence arise from four main issues. First, the complexity of the human body leads to very high-dimensional models to be estimated. Thus, we must deal with 2D–3D projection ambiguities between the real world and the projected images. Third, most of the times the 2D position of the body joints are not observable in the images due to self-occlusions and occlusions with other objects. Additionally, the size and appearance of the human body may change drastically over time due to illumination changes, rotations in-depth of limbs, and loosely fitting clothing. As a result, locating the position of body joints in the image is difficult, and often results in poor, missing or even wrong estimates.

To overcome these issues, many approaches make use of a priori knowledge in the form of a geometrical model of the human body whose parameters are to be estimated given its projections on the 2D images [30]. However, exhaustively searching for the best match within the high-dimensional state space is usually not feasible given the complexity of the model. Therefore, some approaches use machine learning techniques to learn view-dependant mappings between features extracted from the 2D images and the full state space in order to solve the 3D–2D ambiguities [44,21,8]. Alternatively, a common approach is to explore only a part of the state space and use local optimization techniques to find a good solution, such in [9], or to sample a density over the state space using a particle filter (PF) [22]. However, PFs suffer from many well known problems related to its discrete nature [25], and to the fact that a good model of the system dynamics is required in order to represent the posterior pdf properly. Additionally, although it has been shown that there exists an upper bound for the number of particles to achieve a certain estimation error [31], there exists an exponential relationship, up to
a certain bound, between the number of required particles and the dimensionality of the state space for being properly populated [28]. This becomes critical for 3D human motion tracking where the models employed are usually high-dimensional, and dynamics are highly non-linear.

In general terms, there are two main methodologies to improve PFs: modifying the algorithm itself to prevent particle wastage, or designing better dynamic and observation models for the tracked objects. The first ones control the particle quality by directly modifying them. For instance, the kernel-based PF [20] faces the sample impoverishment issue by approximating the likelihood and the posterior densities by a mixture of Gaussians (MoG) at each time step and using them as proposal distributions for sampling new particles. An hybrid search strategy was introduced by Sminchisescu et al., which combines the global particle representation of the posterior with deterministic search for local optimization [42]. Then, in [14] authors make the particle set more efficient, by adapting its size during the estimation process. The key idea is to bound the error by increasing the number of particles when the uncertainty on the state space is high. Finally, some approaches aim to lower the searching complexity by assuming that the state space can be decomposed. For example, in [16] they perform a hierarchical search of the body configuration, or in [28] they use partitioned sampling to build independent observation densities over each dimension of the state space. Alternatively, Deutscher’s et al. annealed PF algorithm [11] replaces the likelihood pdf by a fitness function which measures the quality of a particle relative to an observation. The whole posterior pdf is no longer propagated, thus requiring less costly likelihood evaluations. Additionally, a searching technique is presented which introduces the influence of narrow peaks in the fitness function, gradually. As a result, particles are guided efficiently to a global maximum of the fitness function. See [30] for additional examples.

Alternatively, PFs’ efficiency can be improved by designing better observation models (likelihood) and dynamic models (prior) for the tracked objects. Observation models evaluate the fitness of predicted postures to the measurements available [37,46,3,6,23]. They must deal with severe illumination and viewpoint changes, and most of the times only a few set of body joints may be observable from images. In general, it is very difficult to design robust likelihood functions. As a result, the update of the predictions may not be reliable for a certain period of time due to weak and noisy measurements. Therefore, strong motion priors are also needed to avoid tracking failures. Human motion priors can be exploited to guide the exploration of the state space and propagate particles efficiently to areas of interest. For instance, Sidenbladh et al. sample new body postures from a database of pre-recorded motions [38]. Although they achieved good tracking results, the model could only predict postures which were present in the motion database. Alternatively, Ning et al. [32] tracked a 12 DOF body model of a walking sequence using a PF and a dynamic model of walking including constraints formulated as independent Gaussian distributions per each joint. Chai et al. [10] presented a learning scheme for large motion sets (about 1 h) to re-construct 3D motion in real-time from a few 2D control signals. The system learnt a series of local linear models for the on-line mapping with the control signals and the reconstructed motion consisted in the retrieved postures from the database which best matched the 2D tracked markers. Recently, Urtasun et al. [45] introduced the use of a Gaussian process dynamical model (GPDM) to learn 3D posture and motion priors for 3D human tracking from a small training set. They successfully tested their priors using the 2D position in images of some 3D joints. However, only lateral walking sequences were tested, and instead of a PF, they used an offline inference approach to obtain the MAP estimate over a time window including past and future events. Similarly to our work, Wu et al. [49] learnt a model of feasible hand postures from a real motion database represented in a PCA space, which was used as the importance function in a PF framework for articulated hand tracking. They defined a set of basis hand configurations based on its topology and observed that most natural hand motion can be constrained by the set of linear manifolds spanned by any two basis. However, it is not clear how to define such set of basis for human body postures, and linear manifolds seem to be too restrictive to accommodate natural body motion. Another work using a PCA space to constrain articulated hand motion was carried out by Stenger et al. [43]. They use a tree-based grid filter, partitioning the PCA state space by clustering motion capture data. Then, dynamics are modelled as a first order process with learnt transition probabilities between the states. In addition, a likelihood function based on edge and color information is defined, and initialization of the tracking process is handled by combining hierarchical detection and Bayesian filtering. Recently, this work has been successfully extended to body tracking in a controlled environment [33] by introducing body shape information for the likelihood computation. However, the method strongly relies in the good behavior of the likelihood function, since limbs’ dynamics are modelled using a simple first order motion model.

The contributions of this paper are aimed to use a strong motion prior to improve the robustness and efficiency of a PF framework for monocular 3D full-body tracking of a given set of actions. Towards that end, we introduce an action-specific dynamic model of human motion to avoid particle wastage within the prediction step of the PF. Hence, particles are propagated taking into account their motion history, and previously learnt motion directions from real training data. Next, the state space is constrained by filtering out those body configurations which are not likely according to our motion model. As a result, as long as the truly performed motion lies within the bounds of our motion model, robustness is added to the whole tracker against non-reliable measurements from the image sequence, i.e. in case of occlusions and/or background clutter. In fact, experimental results show that the tracker allows the reconstruction of the 3D motion parameters of a full body stick figure model, using only the 2D positions of a very reduced set of observable joints, namely the head, one hand, and one foot.

The remainder of this paper is organized as follows. Section 2 details the process for learning the motion model from an input sequence of real motion-captured performances. Then, in Section 3 we explain the tracking framework used, and the integration of the learnt motion model within it. Tests over the overall tracking approach are carried out in Section 4 using several walking cycles from three well-known databases. Finally, some concluding remarks and future research lines are outlined in Section 5.

2. Action-specific motion model

This work uses a priori knowledge on how humans move while performing an action which is learnt from a training set of motion sequences acquired with a commercial optical MoCap system [2]. Although our motion model can be trained for any kind of action by selecting an appropriate training set, we choose the walking action to illustrate the overall tracking approach hereafter. Therefore, our motion model is trained with 3D motion capture data for the walking action from the Carnegie Mellon University’s (CMU) Graphics Lab Motion capture database.\footnote{Available at http://mocap.cs.cmu.edu/}. We considered the recorded motion sequences of 12 subjects showing different performances of the walking action. Subsequently, each performance was split into its composing walking cycles, defined as all the body postures in between two consecutive maximums of the angle between both legs
when the left leg remains in the back. As a result, we finally ended up with a set of 16891 body postures corresponding to 126 walking cycles performed by 12 different actors showing different speeds and body configurations.

2.1. Human posture representation

The body model employed in our work is composed of 12 rigid body parts (hip, torso, shoulder, neck, two thighs, two legs, two arms and two forearms) connected by a total of 10 inner joints, see Fig. 1(a). Limbs’ orientation is modelled using 2 DOF as shown in Fig. 1(b), without modelling self-rotation of limbs around its axis. The body segments are structured in a hierarchical manner, constituting a kinematic tree. The root, located at the hip, determines the global rotation of the whole body. Notice that body’s global position is not considered in the model. As a result, we represent a human body posture $\psi$ using 36 parameters, i.e.

$$
\psi = (\theta_1^l, \theta_2^l, \theta_3^l, ..., \theta_{12}^l, \theta_{12}^r, \theta_{12}^r),
$$

where $\theta_1^l, \theta_2^l, \theta_3^l$ are the directional cosines for the limb $l$ as shown in Fig. 1(b). Directional cosines constitute a good representation method for modelling a limb’s orientation since it does not lead to discontinuities, in contrast to other methods such as Euler angles or spherical coordinates. Additionally, unlike quaternions, they have a direct geometric interpretation [50]. However, given that we are using three parameters to determine only 2 DOF for each limb, such representation generates a considerable redundancy of the vector space components. Additionally, the human body motion is intrinsically constrained, leading to highly correlated data in the original space. Therefore, we perform principal component analysis (PCA) over all the postures from an action in order to find a more compact representation. Then, we project all the training postures to the PCA space found, i.e.

$$
\tilde{\psi} = [u_1, ..., u_b]^T(\psi - \bar{\psi}),
$$

where $\psi$ refers to the original posture, $\tilde{\psi}$ denotes the lower-dimensional version of the human posture represented in the PCA space, $[u_1, ..., u_b]$ correspond to the first $b$ selected eigenvectors, and $\bar{\psi}$ is the mean of all the training postures. We name the resulting space as the $a$Space [18] for a particular action. Additionally, each dimension describes a natural mode of variation of human motion while performing an action, resulting in a more suitable and compact representation than the original 36-dimensional vector $\psi$ from Eq. (1). Similar approaches that also use PCA to represent shape and pose variations are [5] to model human shapes from markers or [7] from image data.

In the case of the walking action, we named the resulting space as the $a$Walk space. In Fig. 2 we illustrate the main modes of variation found from PCA. The first (a), second (b), and third (c) components of the mean posture are varied from -3 to 3 times the standard deviation found. As we may observe, the main motion present is related to arms and legs, as the first dimension accounts for the coupled motion between them. Notice that one arm moves accordingly to the leg on the opposite body side. Complementarily, the second and third components encode more subtle motions of legs and arms.

2.2. Synchronizing motion performances

Let us define a particular performance $\Psi_i$ of an action as a time-ordered sequence of $F_i$ postures such as $\Psi_i = \{\psi^1_i, ..., \psi^F_i\}$, where the index $i$ denotes the number of performance. Then, the training set for a particular action is composed of all the $P$ performances that belong to that action, i.e. $A = \{\Psi_1, ..., \Psi_P\}$.

The training sequences were acquired under very different conditions, showing different durations, velocities and accelerations during the performance of the action. Therefore, we need to synchronize the whole training set so that we can establish a mapping between postures from different cycles.

Towards this end, we used a dense matching algorithm based on dynamic programming [35], which computes the disparity matrix between each pair of training sequences and selects the best matching path between them as the common time pattern which minimizes their disparity. Finally, we extract from the input dataset the best global time scale pattern for synchronization according to an intra-class minimum global distance criterion. As a result, all the walking cycles are synchronized to the given time pattern. We may refer the reader to [35] for a detailed explanation of the whole training set synchronization process.

In Fig. 3(a) we show the first $b = 4$ dimensions of the $P = 126$ input sequences for the walking action. They are represented in the $a$Walk space, explaining a total of 90.4% of the original variance present. As shown, they have different durations and the speed they were performed changes along each sequence. The results of applying the synchronization algorithm are shown in Fig. 3(b). Hereafter, we will consider the synchronized version of the input performances when referring to the training set for our motion model. Therefore, $A = \{\Psi_1, ..., \Psi_P\}$ will refer to the synchronized version of the training set in the remainder of this paper, in order to simplify the notation.
2.3. Learning the motion model

Once all the motion sequences share the same time pattern, we learn an action specific motion model which will be used to improve the performance of the PF tracker as detailed below.

First, we extract from the training set, \( A = \{ \Psi_1, \ldots, \Psi_P \} \), a mean representation of the action by computing the so-called mean performance \( \bar{\Psi} = \{ \bar{\psi}_1, \ldots, \bar{\psi}_F \} \), where each mean posture \( \bar{\psi}_t \) is defined as

\[
\bar{\psi}_t = \frac{1}{P} \sum_{i=1}^{P} \psi^i_t, \quad t = 1, \ldots, F.
\]

where \( \psi^i_t \) corresponds to the \( t \)-th posture from the \( i \)-th training performance, and \( F \) denotes the total number of postures of each synchronized performance.

Then, the key idea is to learn different parameters of the motion model associated to each time step of the mean performance. Towards this end, we quantify how much the training performances \( \Psi_i \) vary from the mean performance \( \bar{\Psi} \) of Eq. (3). Therefore, for each time step \( t \), we compute the standard deviation \( \sigma_t \) of all the postures \( \psi^i_j \) that share the same time stamp \( t \), i.e.

\[
\sigma_t = \sqrt{\frac{1}{P} \sum_{i=1}^{P} (\psi^i_t - \bar{\psi}_t)^2}.
\]

Fig. 4 shows the learnt mean performance \( \bar{\Psi} \) (red solid line) and \( \pm 3 \) times the computed standard deviation \( \sigma_t \) (dashed black line) for the walking action. The first \( b = 6 \) principal components from the \( a\text{Walk} \) space are represented in the figure, which explain the 93.3% of the total variation of training data.

Then, we characterize the temporal evolution of the action by computing the mean direction of the motion \( \mathbf{v}_t \) for each subsequence of \( d \) postures from the mean performance \( \bar{\Psi} \) as

\[
\mathbf{v}_t = \frac{\mathbf{v}_t}{\| \mathbf{v}_t \|}, \quad \mathbf{v}_t = \frac{1}{d} \sum_{j=t-d+1}^{t} (\bar{\psi}_j - \bar{\psi}_{j-1}) / \| (\bar{\psi}_j - \bar{\psi}_{j-1}) \|,
\]

where \( \mathbf{v}_t \) is a unitary vector representing the observed direction of motion averaged from the last \( d \) postures at a particular time step \( t \). In our experiments, we used \( d = 10 \) as the length of the subsequences considered out of a mean walking cycle length of \( F = 198 \) postures.
Finally, we learn the expected error from the dynamic model at a given position of the mean performance. Assuming that new particles are propagated within the PF following a first order motion model with Gaussian noise, we characterize the expected error committed by the dynamic model by applying our dynamic model to every training posture, and computing the error w.r.t. the truly performed ones. Then, we will use the covariance of this error at each time step to define the diffusion term of the PF. As a result, different parameters for the diffusion model can be applied depending on the time step within the walking cycle.

The step by step process is as follows. First, for each posture \( \psi_i \) of each training performance \( \Psi_i \), we predict a posture \( \hat{\psi}_i^{t+1} \) for time \( t + 1 \) using the velocity observed in the mean performance at the corresponding time instant, i.e. \( \psi_i^{t+1} = \psi_i^t + (\hat{\psi}_i^{t+1} - \bar{\psi}_t) \), being \( \hat{\psi}_i^{t+1} - \bar{\psi}_t \) the observed velocity from the mean performance at time \( t \). Then, we compute the error between the predicted posture \( \psi_i^{t+1} \) and the real posture from the training set at time \( t + 1 \) as \( e_i^t = |\psi_i^{t+1} - \psi_i^t| \). This is done for all the postures from all the training performances, ending up with \( P \) error measures per each time step \( t \), i.e. \( e_t = (e_1^t, \ldots, e_P^t) \). Then, assuming that the error committed by the constant velocity model follows a normal distribution, its covariance at time \( t \) is calculated as

\[
\Sigma_t = E\left[ (e_t - \frac{1}{P} \sum_{i=1}^{P} e_i^t) (e_t - \frac{1}{P} \sum_{i=1}^{P} e_i^t)^T \right].
\]  

where \( e_t = (e_1^t, \ldots, e_P^t) \) is the error committed by the assumed first order motion model. The computed covariance matrices \( \Sigma_t \) characterize the Gaussian diffusion model in the stochastic state estimation process of our tracking approach. This results in an adaptive-noise term within the PF which improves the efficiency of this stochastic exploration process as opposed to fixed-diffusion models. Fig. 5 shows the first three dimensions of 100 samples from each Gaussian distribution for the diffusion model, centered at their corresponding posture from the mean performance.

Finally, the action-specific motion model \( \Gamma \) is defined as \( \Gamma = \{ \Psi, \sigma_t, \nu_t, \Sigma_t \} \), where \( \Psi \) is the mean performance, and \( \sigma_t, \nu_t, \Sigma_t \) correspond to the computed standard deviation, mean direction of motion and covariance matrix of the error at time step \( t \), respectively.

3. Probabilistic tracking framework

We face the problem of full-body 3D tracking as a Bayesian inference task on the motion parameters of our 3D body model over time. The Bayesian filter decomposes the problem into two differentiated steps, i.e. the prediction and update steps. The prediction step projects forward the model parameters to the next time step by means of a temporal prior. Then the update step makes use of a likelihood probability function in order to evaluate the fitness of the predictions to the evidences available at each moment.

Formally, within the Bayesian filtering framework, we formulate the computation of the posterior distribution \( p(\phi_t | I_t) \) of our model parameters as

\[
p(\phi_t | I_t) \propto p(I_t | \phi_t) \int p(\phi_t | \phi_{t-1}) p(\phi_{t-1} | I_{t-1}) d\phi_{t-1},
\]

where, at time \( t \), \( \phi_t \) represents a particular body posture, \( I_t \) are the evidences from image data available, \( p(I_t | \phi_t) \) is the likelihood of observing the evidences \( I_t \) given \( \phi_t \), and finally \( p(\phi_t | \phi_{t-1}) \) is the temporal prior.

Unfortunately, an analytical solution of Eq. (7) cannot be calculated unless strong assumptions about Gaussianity of the involved distributions are made. Instead, we use a PF [12,22] to approximate the true posterior pdf by means of a discrete set of \( N \) weighted samples, i.e. \( \{ \phi^1_n, \ldots, \phi^N_n \} \), where \( \phi^k_n \) is the particle number \( n \)-th and \( \pi^k_n \) its normalized weight at time \( t \). Then, we define the state of the object at a given time as the expected value of the posterior pdf, i.e. \( \phi_t = E(\phi_t) = \sum_{n=1}^{N} \pi^k_n \phi^k_n \).

3.1. Improving tracking performance

Although the PF framework has been widely used for human body tracking [11,28,38], it suffers from a significant number of problems as stated in Section 1. This work tackles these problems by using a low-dimensional body posture representation, and the introduction of two main contributions: an efficient dynamic model for predicting human postures, and the constraint of the state space to the most plausible solutions. Thus, particle wastage is avoided and robustness is added to the overall tracker compared to a standard PF with a generic motion prior.

As a result, we propagate each selected particle \( \phi^k_n \) corresponding to a b-dimensional point from the 6Space representation (Eq. (2)), as follows:

1. Identify which part of the learnt mean performance \( \bar{\Psi} \) from Eq. (3) is more similar to \( \phi^k_n \). Thus, we probabilistically match the particle \( \phi^k_n \) and a subsequence of the last estimated motion history against all the subsequences from \( \Psi \) having the same length.
2. Propagate the particle over time by means of a first order motion model and a Gaussian diffusion term whose parameters are retrieved from the action model at the time instants that matched the mean performance.
(3) Constrain the possible solutions to feasible postures according to the learnt action model at the matched time instant. If the predicted posture is considered to be invalid, the prediction is wasted, and a new particle is stochastically selected from the particle set representing the posterior pdf. Then, start over with step 1 until a prediction is accepted.

In the following subsections a detailed explanation of each step is given. In addition, the full posterior’s propagation process can be found in Algorithm 1.

3.1.1. Probabilistic match

Our probabilistic matching approach aims to identify which part of the mean performance is more similar to the current particle. On the one hand, we define the subsequence of estimated motion to be matched at time \( t \), by concatenating the currently selected particle with the last \( (d-1) \) estimated postures of the motion history, i.e. \( \Phi^\theta_1 = (\Phi_{t-d+1}, \ldots, \Phi_{t-1}, \Phi_t) \).

On the other hand, we define a motion subsequence of length \( d \) from the mean performance at time instant \( i \) as \( \Psi_i = (\bar{\psi}_{i}, \ldots, \bar{\psi}_d) \).

Then, we define a similarity measure between two motion subsequences of length \( d \) within the \( \alpha \)Space, namely \( \Psi = (\bar{\psi}_1, \ldots, \bar{\psi}_d) \) and \( \Phi = (\phi_1, \ldots, \phi_d) \), as

\[
S(\Phi, \Psi) = \exp(-D_M(\Psi, \Phi)) \left[ (\mathbf{v}_\Psi \cdot \mathbf{v}_\Phi) + 1 \right]^2, \tag{8}
\]

where \( \cdot \)stands for the dot product between the average motion direction vectors \( \mathbf{v}_\Psi \) and \( \mathbf{v}_\Phi \) from Eq. (5). \( D_M \) is the sum of the Mahalanobis distances within the \( \alpha \)Space between each subsequences’ postures \( \bar{\psi}_i \) and \( \phi_j, j = 1, \ldots, d \).

This similarity measure is composed of two terms. The exponential term accounts for the spatial proximity between postures within the \( \alpha \)Space, while the dot product term expresses similarity w.r.t. directions of motion across time, regardless the body postures shared between different action models is kept by the approach. The final similarity computation is multimodal, only the most similar subsequences \( \Psi_i \) from the mean performance as \( S_i = S(\Psi_i, \Phi^\theta_1) \), \( i = 1, \ldots, F \), and then randomly selecting a matching sequence \( \Psi^\circ_i \) with probability

\[
p(\Psi^\circ_1|\Psi^\theta_1) = \frac{s_i^\theta}{\sum_{i=1}^F s_i^\theta}, \quad i = 1, \ldots, F, \tag{9}
\]

where \( F \) is the total number of postures from the mean performance.

Alternatively, one could opt for a deterministic matching approach by selecting the subsequence \( \Psi^\circ_i \) with the highest similarity \( S_i^\theta \). However, in case \( p(\Psi^\circ_1|\Phi^\theta_1) \) is multimodal, only the mode with highest similarity would be selected, thus loosing the remaining ones. Furthermore, the extension to multiple action models is straightforward given the probabilistic definition of the matching process. Hence, the multimodality introduced in \( p(\Psi^\circ_1|\Phi^\theta_1) \) by similar postures shared between different action models is kept by the approach. In such case, the cost of matching for a time step of the PF tracker is \( N \cdot \sum_{i=1}^F s_i^\theta \), where \( N \) is the number of particles used, and \( F_i \) is the number of postures of the mean performance for the action \( i \) out of a total of \( K \) actions.

3.1.2. Dynamic model definition

Within the prediction step of the PF, we project forward the particle set representing the posterior at time \( (t - 1) \) by drawing new samples \( \Phi^\circ_i \) from the dynamic model \( p(\phi_{t+1} | \phi_t) \) of Eq. (7). Following the approach described by Sidenbladh in [38], we extend the state space to store the history \( \Phi_{t-1} = (\phi_1, \ldots, \phi_{t-1}) \) of the last \( d \) estimated postures, and sample from the conditional distribution \( p(\phi_t | \Phi_{t-1}) \) instead of considering only the last posture \( \phi_t \). Finally, given \( \Phi^\circ_i \), new samples \( \Phi^\circ_i \) are computed as

\[
\Phi^\circ_i = \Phi^\circ_{i-1} + (\bar{\psi}_{i+1} - \bar{\psi}_i) + \eta \Sigma_{i+1}, \tag{10}
\]

where \( i \) is the index of the motion subsequence from the mean performance which probabilistically matched \( \Phi^\circ_i \) according to Eq. (9). Hence, \( (\bar{\psi}_{i+1} - \bar{\psi}_i) \) corresponds to the velocity present in the mean performance \( \bar{\Psi} \) (Eq. (3)) at the matched subsequence, and \( \eta \Sigma_{i+1} \) is a zero-mean Gaussian noise function with covariance \( \Sigma_{i+1} \) from Eq. (6). Therefore, by sampling from the prior \( p(\phi_t | \Phi_{t-1}) \) a particle is propagated by a first order motion model that uses the learnt velocity and error’s covariance from the matched subsequence \( \Psi_i \) of the mean performance. Hence, a priori knowledge on human motion is used to guide the exploration of the state space.

As a result, on the one hand, we achieve a more efficient use of the particle set compared to more generic dynamic models as long as both the training and testing sequences belong to the same class of motion. On the other hand, a poor estimate could be obtained from the mean performance in case the motion to be tracked is too different from the learnt model, or its framerate differs too much from the framerate of the training sequences. While the former would require training the system with examples of this kind of motion, the latter is accommodated by the present approach as long as the framerate difference is not too large. This is due to the probabilistic matching approach, and the nature of the particle filtering framework. Hence, although in the presence of substantial framerate differences between the testing sequence and the mean performance, the final matching probability is still maximum between the most similar subsequences, since its similarity scores are normalized in Eq. (9). Then, the Gaussian diffusion term from Eq. (10) and the stochastic nature of the particle filtering framework contribute in accommodating the prediction error, up to a certain limit.

3.1.3. Constrained solution space

After new postures are sampled, we apply a filtering step which discards predicted particles which do not correspond to feasible human postures according to our action model. Hence, given the matched subsequence \( \Psi_i = (\bar{\psi}_{i-d+1}, \ldots, \bar{\psi}_i) \), we determine that a particular posture \( \phi_i \) lies within the variation bounds accepted for the action if

\[
|\phi_{ij} - \bar{\psi}_i| < k \cdot \sigma_{i+1,j}, \quad \forall j = 1, \ldots, b_{ij}, \tag{11}
\]

where \( \phi_{ij} = (\phi_{ij}^1, \ldots, \phi_{ij}^b) \) is the \( b \)-dimensional predicted particle representing a particular body posture in the \( \alpha \)Space. Complementarily, \( \bar{\psi}_i = (\bar{\psi}_{i+1,1}, \ldots, \bar{\psi}_{i+1,b}) \) is the next posture to \( \Psi_i \), i.e. the immediately following posture from the mean performance which probabilistically matched the current particle according to Eq. (9). Then, \( \sigma_{i+1,j} = (\sigma_{i+1,1}, \ldots, \sigma_{i+1,b}) \) stands for the learnt standard deviation for the \( i \)-th posture of the matched subsequence. Subsequently, \( k \) is a scale factor for the variance allowed. Hence, too small values for \( k \) lead to accepting only postures almost equal to the ones stored in the mean performance \( \bar{\Psi} \). Typically we set \( k \) to 3, thus including the 99.73% of variation of the training set.

Finally, \( b_{ij} \) determines the number of dimensions from the \( \alpha \)Space considered for filtering. Notice that \( b_{ij} \leq b \), where \( b \) is the total number of dimensions in the \( \alpha \)Space (Eq. (2)). While the accuracy of the representation is a matter of as more dimensions the better, by
not using all the b dimensions for filtering, we allow to track subtle motions which were not present in the training set, while filtering out non-likely postures according to the most important modes of variation found for the action. Hence, \( b_{\text{bias}} \) controls the trade off between generality and specificity of the filtering method. In our experiments, we achieved better results for the walking action setting \( b_{\text{bias}} = 3 \) and \( b = 5 \).

Therefore, the approach is as follows: given a selected particle \( \phi^p_{n-1} \) to be propagated, we use the dynamic model defined in Eq. (10) to obtain a new sample. Then, if the new sample is not accepted according to Eq. (11), this particle is dropped and a new one is stochastically selected. Then, the propagation process is restarted until a prediction is accepted.

By removing the particles which are not accepted we modify the particle set representing the posterior distribution. Thus, after this process, they are not representing that distribution anymore, but a pruned version of the posterior pdf, since indeed, the posture filtering step can be seen as a pruning of the state space where particles live. An alternative to dropping rejected particles, which does not modify the convergence results of the PF, is to sample additional particles by importance sampling until enough accepted particles are achieved. However, such a method requires computing the likelihood of each extra sampled particle, which is usually the most computationally expensive part of a PF framework. Consequently, being aware that we are sampling from a pruned version of the posterior pdf, in this work we implement the particle removal method and show that also good tracking results are obtained by dropping beforehand those predictions which are not likely to appear during the performance of a particular action.

### 3.2. Updating the predictions

Once the particle set \( \{ \phi^p_n \} \) has been propagated from \( t - 1 \) to \( t \), the predictions are updated assigning a weight to each particle corresponding to the fitness of the predicted posture to the evidences available at time \( t \). This is done by evaluating the likelihood function \( p(1|\phi^r_i) \) from Eq. (7) for each particle from the set \( \{ \phi^r_i \} \). Then, the computed weights are normalized, obtaining the posterior representation \( \{ \phi^r_i, \bar{p}^r_i \} \) at time \( t \).

Towards this end, we first reconstruct the human body posture encoded by each particle. Hence, given a predicted particle \( \phi^r_i \) in the aSpace, we project back to the original 36-dimensional representation and divide each limb’s direction cosines vector \( (\theta_1, \theta_2, \theta_3) \) by its norm, so that the restriction \( (\theta_1)^2 + (\theta_2)^2 + (\theta_3)^2 = 1 \) is satisfied. Then, we compute the 3D absolute positions of each joint \( j \) from the human body model, thus obtaining the vector \( X = (x_1, \ldots, x_f) \) which represents the human body defined as a set of \( f \) virtual markers.

Subsequently, using the 3D joints positions available from ground truth data for the test sequences, we define the likelihood of the evidences \( k_i \) given the predicted particle’s \( \phi^r_i \) corresponding posture as

\[
p(1|\phi^r_i) \propto e^{-\gamma \sum_{j=1}^{f} \text{dist}(x_j, x_{cj}^T)},
\]

where \( \text{dist} \) stands for the Euclidean distance between the 3D joint position \( x_j \) from the predicted posture and its corresponding 3D joint position \( x_{cj}^T \) from ground truth data. Additionally, \( \gamma \) is a scale factor which determines the “peakness” of the likelihood function, with a direct impact on the particle survival rate. Hence, the higher \( \gamma \) is, the higher is the difference between the probability of the most likely and the most non-likely postures. In our experiments, \( \gamma = 80 \) showed to be a good trade off for keeping a balanced particle set to represent the posterior pdf for the tested sequences.

However, given the nature of the evidences available, i.e. a monocular image sequence, we can only perform 2D measurements about the body joints configuration. Therefore, we defined a more suitable method for evaluating the fitness of the predicted postures to the evidences. First, we projected the predicted and ground truth 3D joints positions following a perspective projective model in an specified viewpoint, thus obtaining a set of \( J \) joints’ 2D positions. Then, the likelihood function is defined analogously to Eq. (12) but \( x_j \) and \( x_{cj}^T \) stands for the 2D positions of the projected joints. Notice that we can vary which joints are involved in the likelihood computation, and from which viewpoint we obtain their 2D measurements.

Therefore, this work assumes that the tracker has been already initialized, i.e. that we know the first \( d \) 3D body postures from the performed motion. At present, work is being done to combine direct 3D body posture inference methods that use 2D body shape information extracted from monocular video images [36]. Such methods perform well with postures having low 2D/3D ambiguity which are suitable to be used for initializing the tracker and recovering from critical failures.

Finally, if the tracker has been already initialized, the pseudocode for propagating the posterior estimation over time is shown in Algorithm 1.

**Algorithm 1.** Pseudo-code of the posterior’s propagation algorithm over time.

* for \( t = 2, \ldots, T \)
  o for \( n = 1, \ldots, N \)
    (1) Select a particle to be propagated according to its weight:
      Draw \( n' \sim \{1, \ldots, N\} \) such that \( p(n' = i) = \bar{p}^r_{n-1} \), \( i = 1, 2, \ldots, N \)
    (2) Propagate the selected particle \( \phi^{p}_{n-1} \):
      a) Build \( \Phi^r_{n-1} = (\phi_1, \ldots, \phi_k, \phi^p_{n-1}) \)
      b) Probabilistically match \( \Phi^r_{n-1} \) with a motion subsequence \( \tilde{Y}_t \) of the same length from the mean performance:
        i) Compute \( s^p_i = S(\tilde{Y}_t, \phi^{p}_{n-1}), i = 1, 2, \ldots, T \) using Eq. (8)
        ii) Stochastically select \( \tilde{Y}_t \) with matching probability \( p(\tilde{Y}_t|\phi^{p}_{n-1}) \) given by Eq. (9)
      c) Predict a new particle \( \phi^r_i \) using the dynamic model, Eq. (10): \( \phi^r_i = \phi^r_{i-1} + (\tilde{Y}_t - \psi_{i-1}), \psi_{i} = (A^p_{i1}, \ldots, A^p_{ib}) \)
      d) Constrain the solution space by filtering out non-feasible predictions according to Eq. (11):
        i) Compute deviation between the predicted particle \( \phi^r_n \) and the last posture \( \bar{p} \), from the matched sequence \( \tilde{Y}_t \) of the mean performance: \( A^p_{ij} = |\phi^r_n - \bar{p}_{ij}| = (A^p_{i1}, A^p_{i2}, \ldots, A^p_{ib}) \)
        ii) Check if the particle is accepted by the action model:
          if \( (A^p_{ij} < k\cdot\sigma_{ij}) \), \( \psi = 1, \ldots, b_{f_{ij}} \) (posture lies within the boundaries):
            Accept the prediction \( \phi^r_n \) for this particle and proceed on propagating the next particle.
          else Drop the predicted particle \( \phi^r_n \) and proceed to step (1).
    endif
  endif
endfor

(3) Update step: Compute the likelihood of the prediction using Eq. (12):
\[
p^p_n \propto p(1|\phi^r_n)
\]
4. Experimental results

In order to test the performance of the overall tracking approach, we first present the testing set used and define a suitable error measure. Then, we discuss on choosing an appropriate number of dimensions for the aSpace representation. Finally, we evaluate our approach regarding the tracking efficiency improvement in terms of the number of needed particles, the computational cost, and the robustness against ambiguous and incomplete measurements to update the predictions, with testing sequences from three well-known databases.

4.1. Testing set

The proposed tracking approach is tested using motion performances of the walking action. However, the approach is easily extensible to any other actions by choosing a representative training set. Hence, several walking cycles are selected from different motion databases. The first sequence consists in four continuous walking cycles from the same database used for training, previously removed from the training set. The second test sequence consists of two and a half walking cycles from the HumanEva-I dataset [40]. This dataset comprises four subjects performing six different types of actions recorded in seven calibrated video sequences from different viewpoints. Additionally, the video sequences are synchronized with their corresponding motion captured 3D pose parameters. Notice that this database was acquired under different conditions and marker placements than the CMU database used for training. Finally, the last testing sequence corresponds to a walking video from the CAVIAR dataset [1], which comprises several manually annotated video sequences of real subjects in an entrance lobby and a shopping center, resulting in more realistic testing conditions.

4.2. Error measure

We represent the pose of the body using \( J = 15 \) virtual markers, corresponding to the joint centers and limb ends of the human body model. Thus, given a particular body posture in the aSpace representation, we rewrite its body configuration as \( X = (x_1, \ldots, x_J) \), where \( x_j \in \mathbb{R}^3 \) is the 3D position of the marker \( j \) as in Eq. (12). Then, the error between an estimated body posture \( X^e \) and the truly performed one \( X^{GT} \) from ground truth data is computed as the average squared distance between individual 3D joints, i.e.

\[
D(X^e, X^{GT}) = \frac{1}{J} \sum_{j=1}^{J} ||x^e_j - x^{GT}_j||^2.
\]

(13)

4.3. Determining the number of dimensions of the aSpace

Then, we determine the appropriate number of \( b \) dimensions considered for building the aSpace representation. On the one hand, we project back the training sequences to the original representation space as previously discussed. In Fig. 6(a) we show a boxplot of the mean reconstruction error in mm computed for all the training sequences varying the \( b \) parameter. It can be seen that the error is high for the first four dimensions, and it gets stabilized below 8 mm after \( b = 5 \) dimensions.

On the other hand, we empirically validate the most suitable value for \( b \) by running several complete tracking tests with fixed parameters but varying \( b \). Hence, we selected four different walking cycles from the CMU database, and ran the PF tracker fixing all the parameters involved but \( b \). We use \( N = 500 \) particles, \( d = 10 \), \( \gamma = 80 \), and \( \beta_{b_{en}} = 3 \). The likelihood of each predicted particle was computed according to the mean 2D distance of all the projected joints between ground truth and estimated postures, both from a lateral viewpoint. Hence, we can evaluate the whole framework without the influence of artifacts caused by image-based likelihood measures. The tracker was initialized with ground truth data.

Then, several runs were carried out varying the \( b \) parameter from 3 to 12. Finally, for each value of \( b \) tested, we computed the mean estimated 3D joints error from the four testing cycles. The results are shown in Fig. 6(b). We observed that while considering more dimensions for the aWalk representation actually lowers reconstruction error, the final estimation error gets higher as one keeps adding more dimensions after \( b = 5 \). This is due to the fact that the number of needed particles grows exponentially, up to a certain bound, w.r.t. the number of dimensions of the state space [28,31]. Therefore, choosing \( b = 5 \) results in a good trade off between the dimensionality reduction performed by PCA and accuracy of the estimation for the walking action with a manageable number of particles. Finally, with \( b = 5 \) dimensions we explain the 94.55% of the variance present in the original training data, with a mean reconstruction error of 7.68 mm.
4.4. Tracking results

In order to test the performance of the proposed approach, we carried out several tests comparing the results obtained between three different tracking methods. First, we used a standard PF tracker with a first order motion model. Second, our tracker using the prior model but without including the posture filtering step defined in Section 3.1.3. Finally, our full tracking approach using the prior model and the posture filtering step. Hereafter the three tracking methods are referred to as: generic PF tracker, our tracker with the posture filtering step, and our tracker without the posture filtering step, respectively. By posture filtering step, we refer to the rejection of predictions corresponding to non-likely postures according to Eq. (11). Notice that by setting the parameter $b_{filt} = 0$, we are accepting all the predictions, and thus, the posture filtering step is omitted.

The first tests are intended to show the efficiency improvement of our tracker in terms of the computational time and the number of particles needed to achieve a certain error, for motions belonging to the type of action learnt. Thus, we compare the time consumption and the mean estimation error obtained by a standard PF with a very general motion prior against our motion model guided tracker with and without the posture filtering step, while varying the number of particles used.

The PFs generic motion prior consists of a constant velocity model for the parameters, where each parameter is independent. The diffusion term consists of a fixed white additive Gaussian noise (WAGN) with covariance computed in the same fashion than Eq. (6) for the dynamic’s model expected error but considering all the postures from the training set. We used only 1 cycle from the CMU database for testing motivated by the fact that the generic PF tracker lost track very quickly, and after that point, the error obtained does not scale well to the number of particles used. Namely, we fixed $b = 5$, $d = 10$, $\gamma = 80$, and $b_{filt} = 3$ or 0 for runs with or without the posture filtering step, respectively. The same likelihood computation method as the previous test was used.

Fig. 7 relates the average computation time per frame, the mean estimation error, and the number of particles used in each filter run. All three trackers are implemented in MATLAB and running on a PC with an Intel(R) Pentium(R) 4 CPU at 3.2 GHz with 2 Gbytes of RAM, and the code has not been optimized for high performance. On the one hand, the high error obtained by the generic PF for less than 500 particles ($> 53$ mm) is explained by the fact that it totally lost track of the target after a few frames. Mistracks generally occurred after a large acceleration in the parameter space, since the generic motion prior assumes a constant velocity for each parameter. Hence, the ability of the generic PF to handle accelerations depends on the diffusion model and the number of particles considered. Thus, the larger the acceleration is, the larger the diffusion applied should be, demanding more particles to properly populate the state space.

On the other hand, from 500 particles or more, the generic PF tracker could complete a full cycle without losing track. However, the final error of the generic PF is much higher (21.32 mm for the $N=10000$ particles test) than any of the errors obtained by our tracker with posture filtering. Hence, even adding more particles to the generic PF the estimation error stabilizes and never achieves better results than the 100 particles test for the tracker with posture filtering (MSE of 19.53 mm). This is due to the role of the posture filtering step within the tracking process. Hence, we observed that indeed, it discards modes in the likelihood function which would give high weights to particles corresponding to badly estimated human postures at a given time instant. Thus, without filtering these non-likely postures, these particles are considered for computing the final estimated state and consequently, the overall tracking performance decays.

Regarding time consumption, the generic PF and the tracker without the posture filtering step have very similar processing times. This is explained by the fact that the computation of the likelihood is usually the most time consuming part of a particle filtering framework. In comparison, the overhead introduced by the dynamic’s model probabilistic matching is almost negligible in the experiments carried out.

Then, the tracker with posture filtering shows a slight increase w.r.t. the processing time at a given number of particles, but outperforms the other two regarding the final estimation error obtained. For instance, for the $N=10000$ particles test, the average processing time per frame was 62.97, 63.02 and 70.08 s, with an MSE of 28.32, 15.85 and 8.16 mm for the generic PF, the tracker without posture filtering, and the tracker with posture filtering, respectively. Comparatively, for the $N=100$ particles test, the results obtained were 0.63, 0.63 and 0.70 s with an MSE of 189.27, 36.93 and 19.53 mm.

Fig. 7 relates the average computation time per frame, the mean estimation error, and the number of particles used in each filter run. All three trackers are implemented in MATLAB and running on a PC with an Intel(R) Pentium(R) 4 CPU at 3.2 GHz with 2 Gbytes of RAM, and the code has not been optimized for high performance. On the one hand, the high error obtained by the generic PF for less than 500 particles ($> 53$ mm) is explained by the fact that it totally lost track of the target after a few frames. Mistracks generally occurred after a large acceleration in the parameter space, since the generic motion prior assumes a constant velocity for each parameter. Hence, the ability of the generic PF to handle accelerations depends on the diffusion model and the number of particles considered. Thus, the larger the acceleration is, the larger the diffusion applied should be, demanding more particles to properly populate the state space.

On the other hand, from 500 particles or more, the generic PF tracker could complete a full cycle without losing track. However, the final error of the generic PF is much higher (21.32 mm for the $N=10000$ particles test) than any of the errors obtained by our tracker with posture filtering. Hence, even adding more particles to the generic PF the estimation error stabilizes and never achieves better results than the 100 particles test for the tracker with posture filtering (MSE of 19.53 mm). This is due to the role of the posture filtering step within the tracking process. Hence, we observed that indeed, it discards modes in the likelihood function which would give high weights to particles corresponding to badly estimated human postures at a given time instant. Thus, without filtering these non-likely postures, these particles are considered for computing the final estimated state and consequently, the overall tracking performance decays.

Regarding time consumption, the generic PF and the tracker without the posture filtering step have very similar processing times. This is explained by the fact that the computation of the likelihood is usually the most time consuming part of a particle filtering framework. In comparison, the overhead introduced by the dynamic’s model probabilistic matching is almost negligible in the experiments carried out.

Then, the tracker with posture filtering shows a slight increase w.r.t. the processing time at a given number of particles, but outperforms the other two regarding the final estimation error obtained. For instance, for the $N=10000$ particles test, the average processing time per frame was 62.97, 63.02 and 70.08 s, with an MSE of 28.32, 15.85 and 8.16 mm for the generic PF, the tracker without posture filtering, and the tracker with posture filtering, respectively. Comparatively, for the $N=100$ particles test, the results obtained were 0.63, 0.63 and 0.70 s with an MSE of 189.27, 36.93 and 19.53 mm.
On average, the overhead introduced by the posture filtering step takes 10.1% of the total processing time on the experiments carried out. This overhead is directly proportional to the prediction acceptance ratio, defined as the ratio between the number of predictions sampled from the dynamic model and the number of postures accepted. Fig. 7(b) shows the mean prediction acceptance ratio per frame obtained. Hence, while the mean ratio is 6.26, there are two peaks of around 20 sampled predictions per accepted posture in the beginning and the end of the tested sequence. Interestingly, we observed that they occur in areas where the mean performance has high curvature, since particles show some inertia which makes its adaptation to abrupt changes in the state of the tracked object more difficult. This inertia can be explained by the assumption of a first order motion model and the probabilistic sequence matching technique of the prediction step. However, specially in this situation, the posture filtering step shows bigger improvements in the particle set efficiency, since most of the predictions that wrongly follow the particle’s inertia are rejected.

The second set of tests are intended to validate the ability of our approach to keep track of the target’s state when reduced and ambiguous measurements from the scene are used to update the predictions. Although there exist many works which extract the whole 2D position of body joints from images with relative success \[36,27,29\], only a reduced set of joints is typically observable at each time step. Hence, finding the head, hands and feet 2D position in images might be a feasible task. Therefore, we use 2D ground truth data to test the robustness of our tracker against incomplete evidences in two different manners. From the one hand, we use ground truth information from all the joints available, and from the other hand, we consider only three joints, namely the head, one foot and one hand. The viewpoint used is also varied between a lateral and a totally frontal one. Consequently, we designed different tests with an increasing level of difficulty regarding these parameters for the likelihood computation. In every test the proposed tracker is also compared to a PF tracker with our motion prior, but without including the posture filtering step.

We fixed some parameters and varied the number of joints and the viewpoint considered the likelihood computation. Specifically, we used \(N=500\) particles, \(\alpha=30\) from Eq. (8), \(\gamma=80\), \(b=5\), \(d=10\), and \(k=3\) and \(b_{\text{filt}}=3\) was used for the posture filtering step from Eq. (11).

First, we projected the 3D postures to a 2D plane using a perspective projective model. We ran the tracker against testing sequences #1 and #2 for both viewpoints, and computed the likelihood of the predicted postures based on the 2D positions of all the body joints from the estimated postures vs. their ground truth. Additionally, every test was repeated 30 times in order to provide statistical significance to the results obtained. Fig. 8 shows the mean estimation errors per joint in mm for both testing sequences and the standard deviation observed. Blue solid and red dashed lines encode whether we used the filtering step (PFT filter) to constrain the state space or not (NPFT filter). First of all, we may observe that for the frontal viewpoints a greater overall error is obtained compared to the lateral ones. Second, we may observe that the combination of reducing the number of joints and using a frontal viewpoint has the worst performance. However, by including the posture filtering step, we can obtain a considerable improvement in the likelihood computation.
then a lateral viewpoint, both filters (PFT and NPFT) can track the sequences with a stable behavior. However, on the frontal viewpoint, the NPFT filter clearly fails after a few frames ($t \approx 300$ and $100$ in sequences #1 and #2, respectively), while the PFT filter shows a low and stable error over time.

The lateral viewpoint, both filters (PFT and NPFT) can track the sequences with a stable behavior. However, on the frontal viewpoint, the NPFT filter clearly fails after a few frames ($t \approx 300$ and $100$ in sequences #1 and #2, respectively), while the PFT filter shows a low and stable error over time.

Regarding the standard deviation, for both viewpoints and sequences, the NPFT filter shows much bigger variability between different runs than the PFT filter. Also, frontal viewpoint tests have bigger standard deviations than lateral ones. This is due to the fact that frontal 2D postures are more ambiguous and difficult to track.

Finally, we repeated the experiment but reducing the number of 2D joints considered. For the likelihood computation, we used the known 2D position of only three joints, namely the center of the head, the left hand and the left foot. In Fig. 9 we show the results obtained. As in the previous experiment, the frontal viewpoints show a greater overall error and standard deviation compared to the lateral ones for the same reasons mentioned above. From one hand, the performance of the non-constrained tracker is very poor even for the lateral viewpoint, since it rapidly loses track of the target as supported by Figs. 9(b)–(d). However, although it has higher standard deviation than in the previous experiment, the PFT filter does perform well and is able to keep track of the target with relatively low and constrained error throughout all the walking cycles from both sequences. Therefore, although resulting in a less generic tracker, the use of the posture filtering step is responsible for stabilizing the tracking error even using a very reduced set of joints to update the predictions, as supported by the tests performed.

Table 1 summarizes the experiments carried out regarding all the tests performed on the two sequences, namely the CMU and HumanEva-I chosen sequences, with and without filtering. The error measures have been obtained on a basis of 30 filter runs per sequence, due to the non-deterministic nature of PFs. NPFT and PFT refers to the results for the non-posture-filtered tracker ($b_{\text{filt}} = 0$), and the full tracking approach with posture filtering ($b_{\text{filt}} = 3$), respectively. Each row corresponds to a particular experiment. We present the mean, maximum and minimum of the mean estimation errors, in millimeters, obtained from each run for each testing sequence (S1 or S2), for each experiment. Additionally, the standard deviation observed in the mean error is also shown, as well as the standard deviation of the estimated postures across different runs to test the quality of the estimates. As previously discussed, the tracking error is considerably higher using frontal views as opposed to lateral views for the likelihood computation. Also, the tracker which does not include the posture filtering step presents an overall higher
mean error, specially in the 2D frontal viewpoint tests, since it lost track of the target in most of the experiments carried out. Additionally, for the tracker with posture filtering, the standard deviation of the mean error is noticeably lower among the different tests, since the estimation error remains constrained and stable throughout all the frames of the testing sequences. Regarding the standard deviation of the estimates, we computed the distance of Eq. (13) between each two estimated postures of each pair of runs. The value shown corresponds to the mean of the standard deviation of these distances along a testing sequence, in order to summarize the estimate variance observed in each test. The results point out that the estimated postures for the 2D lateral experiment considering all the joints are very stable for both the NPFT and PFT trackers. However, on the other experiments, the NPFT tracker shows large variance compared to the PFT one. This is explained by the fact that the PFT tracker constrains the state space and the estimated postures do not differ too much between runs as long as a severe mistrack does not occur.

Then, Figs. 10 and 11 show the stick figures of the estimated 3D postures for the testing sequence #2. They have been overlapped with their corresponding image frames from the HumanEva-I dataset, from two different cameras. The 6 DOF corresponding to 3D postures for the 2D lateral experiment considering all the joints are available. The global position of the subject was manually annotated, as well as the first postures of our body model in order to initialize the tracker.

Fig. 12 shows the results obtained on every 14th frame of the sequence. The stick figures representing the final estimated 3D body postures have been superimposed over their corresponding frames, and the available 2D ground truth position of the annotated limbs is depicted as a purple circle centered at their actual position. On the one hand, we observe that on frames where ground truth data about hands position are not available, the estimated arms 3D posture is given by the strong motion prior. Then, on frames with the right hand next to the subject’s face, the tracker is unable to track its true position due to the restrictive motion prior used. However, on the last frames the quality of the arms tracking improves, since the hand’s 2D ground truth position is more stable and the resulting posture is accepted as a feasible human posture by the action model. On the other hand, the approximate configuration of both legs is successfully estimated along all the walking cycles present, which shows the ability of the proposed approach to produce rough estimates of the performed motion even when a very reduced set of measurements are available.

5. Conclusions

This paper addresses the problem of recovering the approximate parameters of a full body 3D model from a monocular image sequence to be used as the basis for generating qualitative descriptions about how humans behave on the scene. Towards this end, a PF-based tracker uses the 2D positions of a variable set of body joints on the image plane to infer the state of a human body model. A strong motion prior is presented, which is used to deal with issues related to PFs’ discrete nature and to the lack of robustness of likelihood functions for monocular 3D full-body tracking applications.

Hence, the action-specific model of human motion presented is used as a priori knowledge within the PF. From the one hand, we introduced a dynamic model responsible of predicting new body postures given the previously estimated ones, which has proven to drastically improve the efficiency of the PF tracker compared to a constant velocity model. On the other hand, a posture filtering step has been added to discard predictions which correspond to non-feasible body postures. If the motion performed belongs to the trained action class, this adaptive constraint of the state space improves the overall tracking reliability, stabilizes the overall error and avoids mistracks due to ambiguous and incomplete measurements from the real world, as supported by the tests performed.

The whole approach has been trained with walking sequences from the CMU Motion Capture dataset, and tested against sequences from three well-known databases: the same one used for training (the CMU MoCap dataset), the HumanEva-I dataset and the CAVIAR project database. The results showed that the approach generalized

| Table 1 | Summary of the tracking error obtained from each experiment in mm. |
|---------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|
|         | Mean error NPFT | Max error NPFT | Min error NPFT | STD NPFT | Estimate STD NPFT | Mean error PFT | Max error PFT | Min error PFT | STD PFT | Estimate STD PFT |
| All joints 1 | 25.53 | 14.80 | 32.83 | 19.29 | 16.08 | 13.13 | 4.29 | 1.08 | 3.48 | 0.81 |
| 2D lateral 2 | 29.91 | 19.95 | 54.31 | 21.10 | 16.89 | 15.72 | 8.00 | 1.81 | 3.80 | 1.42 |
| All joints 1 | 69.57 | 23.48 | 118.22 | 48.98 | 32.15 | 12.60 | 22.53 | 10.07 | 29.15 | 7.24 |
| 2D frontal 2 | 71.34 | 27.57 | 149.36 | 51.38 | 29.37 | 17.37 | 39.63 | 12.02 | 27.57 | 8.36 |
| Three joints 1 | 40.87 | 19.70 | 55.30 | 39.55 | 17.03 | 10.14 | 9.40 | 6.65 | 20.15 | 2.64 |
| 2D lateral 2 | 36.42 | 19.68 | 64.59 | 36.60 | 19.22 | 10.75 | 10.05 | 7.49 | 30.43 | 3.73 |
| Three joints 1 | 77.08 | 24.42 | 169.43 | 41.21 | 35.17 | 12.87 | 40.82 | 10.43 | 57.44 | 7.54 |
| 2D frontal 2 | 78.00 | 24.82 | 165.19 | 51.51 | 38.02 | 14.65 | 33.89 | 12.71 | 61.83 | 6.29 |
well within the tested action class, and that it is robust against incomplete and ambiguous measurements.

Several tests have been carried out varying the number of joints considered and the viewpoint used to update the predicted postures from the dynamic model. In the worst tested case, results point out that our tracker is able to estimate the 3D configuration of a full-body model providing only the known 2D positions of three joints from a totally frontal viewpoint as measurements to compute the likelihood of the predicted postures. Therefore, future work relies in combining our tracker with approaches aimed to track or identify these 2D body parts, i.e. the head, hands and feet [36,48,29,41,26,27,19].

On the contrary, the approach has the following limitations. First, general non-constrained motion cannot be tracked, since the use of a strong motion prior limits the approach to the subset of actions learnt beforehand. Second, the body's absolute position and orientation estimation has not been addressed in this paper. Then, the PCA-based posture representation has proven to be effective in reducing the dimensionality of the state space, since walking is a highly correlated motion, specially between arms and legs. However, more complex motions with less linear correlation, like break dancing, would not benefit from this amount of reduction. Finally, although the probabilistic matching step provides some flexibility, it is assumed that
the framerate of both training and testing sequences do not differ too much.

Therefore, future research lines also include the estimation of the absolute body orientation from the image sequence, and dealing with the initialization of the tracker for practical applications [33]. Currently, we are exploring solutions that learn a mapping between body silhouettes obtained by background subtraction techniques [24] and the viewpoint at a given height of the camera W.T., the subject [36]. Additionally, the overall tracking approach will be trained for other kinds of actions, and add multiple action support by selecting appropriate training sets and dealing with transitions between actions. Finally, although it is out of the scope of this paper, current work is being done on the integration of the estimated 3D body postures by our tracker within the HSE scheme for scene understanding [13].

Acknowledgments

This work is supported by EC Grants IST-027110 for the HERMES project and IST-045547 for the VIDI-video project, and by the Spanish MEC under Projects TIN2006-14606, TIN2007-67896, and CONACYT-UNDER-INGENIO 2010 MIPRCV/CVIAR/CORES. I. Varona also acknowledges the support of a Ramon y Cajal (co-funded by the European Social Fund) Postdoctoral fellowship from the Spanish MEC.

References

participated as a WP-Leader in the European projects HERMES and VIDI-Video. He co-founded the research lab on Image Sequence Evaluation at the Computer Vision Center in Barcelona.

About the Author—JAVIER VARONA is a Ramon y Cajal research fellow of the Spanish Government in the Universitat de les Illes Balears (UIB). He received his doctoral degree in Computer Engineering from the Universitat Autònoma de Barcelona (UAB) and the Computer Vision Center (CVC), in 2001. His PhD thesis was on robust visual tracking. His research interests include visual tracking and human motion analysis. Currently, he is the leader of a research project for building natural interfaces based on computer vision founded by the Spanish Government (TIN2007-67896). He is member of the ACM.

About the Author—XAVIER ROCA received the degree in Computer Sciences in 1990 from the Universitat Autònoma de Barcelona (UAB) and the PhD degree in Computer Sciences in 1998 from the UAB. Since 1993 he is an associate professor at the Computer Sciences Department of the UAB and a staff researcher of the CVC. He has participated in several projects in computer vision, with private and public funds. In 2002, with other CVC members, a spin-off was created from the results of research projects. He has published 30 papers in national and international conferences and journals. From 1998 to 2003 he was marketing director of the CVC and from 2003 to 2004 vice-director. From the beginning of 2006 is Director of the Computer Science Department of UAB.