Stabilizing Motion Tracking Using Retrieved Motion Priors

Andreas Baak[1], Bodo Rosenhahn[2], Meinard Müller[1], Hans-Peter Seidel[1]
[1]Saarland University & MPI Informatik, Saarbrücken, Germany
[2]Leibniz University Hannover, Institut für Informationsverarbeitung

Abstract

In this paper, we introduce a novel iterative motion tracking framework that combines 3D tracking techniques with motion retrieval for stabilizing markerless human motion capturing. The basic idea is to start human tracking without prior knowledge about the performed actions. The resulting 3D motion sequences, which may be corrupted due to tracking errors, are locally classified according to available motion categories. Depending on the classification result, a retrieval system supplies suitable motion priors, which are then used to regularize and stabilize the tracking in the next iteration step. Experiments with the HumanEVA-II benchmark show that tracking and classification are remarkably improved after few iterations.

1. Introduction

Markerless Motion Capturing (Mocap) is an active field of research in computer vision and graphics [9] with applications in animation (games, avatars), medicine or sports science. The goal is to determine the 3D positions and orientations as well as the joint angles of a human actor from image data. In such a tracking scenario, it is common to assume as input a sequence of multiview images of the performed motion as well as a surface mesh of the actor's body. Since the pose and joint parameters are usually unknown and have to be computed from the image data, one typically has to cope with high-dimensional search spaces (typically more than 30 dimensions for a full body model). To make the tracking problem feasible, the manifold of all virtual possible configurations is often reduced to a lower-dimensional subspace. One possibility is to explicitly prevent self-occlusions and to impose fixed joint angle limits as suggested in [7, 21]. Another option is to directly learn a mapping from the image or silhouette space to the space of pose configurations [1, 17]. A very popular strategy for restricting the search space is dimensionality reduction, either by linear or by nonlinear projection methods. In [18], the low-dimensional space is obtained via PCA and the motion patterns in this space are structured in a binary tree. In [20] it has been suggested to learn a Gaussian mixture from pose configurations. Similarly, in [24], a nonlinear projection is employed, in this case via a Gaussian process model.

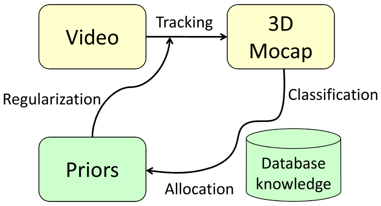

Figure 1. Iterative motion tracking framework.

A recent research strand integrates further sources of information in the motion capture process, e. g., by capturing light sources [2] or by using physical models and forces arising from a ground plane [5, 25]. A common problem with learning-based approaches is that the user needs a good guess on the type of pattern to be expected. For instance, if the user knows that the subject is performing a walking pattern, suited training data is selected and integrated in the tracking system. Current probabilistic learning approaches are limited in their ability to handle large training sets. Only recently, local regression methods have been proposed that allow for coping with a large number of motion patterns within a tracking scenario [23]. In activity recognition, many works rely on 2D descriptors or image silhouettes, such as presented in [8, 22]. To the best of our knowledge, no approach uses activity recognition for stabilizing a tracking framework yet.

In this paper, we introduce a tracking framework that combines methods for markerless motion capturing with a retrieval component in an iterative fashion, see Fig. 1. We start the iteration without applying any prior knowledge on the actions to be performed. The tracking system takes a multiview image sequence ('Video') and returns a sequence of joint positions over time ('3D Mocap'). In our system, we rely on a region-based approach performing joint pose estimation and optimized region splitting similar to [16]. However, our general framework also allows for applying other tracking techniques as presented in [3, 6, 14].

Due to noise, occlusions, and other ambiguities in the image data, tracking may fail for parts of the sequence re-

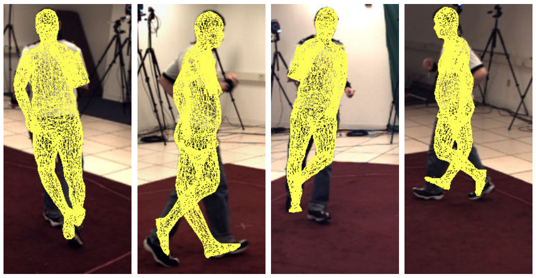

Figure 2. Tracking without priors may lead to pose deformations.

sulting in corrupted poses, see Fig. 2. However, despite of these errors, the overall rough course or at least parts of the motion may still be recognized to a reasonable degree. In the next step, the 3D mocap tracking results are locally classified according to available motion categories. In our approach, these categories are encoded in form of class motion templates (MTs) as introduced in [10]. MTs show a high degree of robustness under spatial and temporal deformations, while revealing consistent aspects of a motion category. In particular, it turns out that typically occurring tracking errors do not have a significant influence on the classification result using MTs. For example, after recognizing a walking cycle in the tracked sequence, this increase of knowledge about the tracked sequence can be used to allocate a simple prior such as 'left foot moves to the front'. Such priors are integrated in the tracking procedure as regularization terms, and the tracking step is repeated to yield an enhanced tracking result. Iterating such a procedure, as shown by our experiments with the HumanEVA-II benchmark, remarkably improves the result after few iterations.

The remainder of the paper is organized as follows. We first summarize the tracking procedure (Sect. 2) and describe the retrieval component (Sect. 3). In Sect. 4, we explain how to fuse the retrieval results with our tracking procedure. To the best of our knowledge, such an iterative tracking procedure using retrieved motion priors has not yet been considered before and constitutes the main contribution of our paper. Besides stabilization of 3D tracking we also gain a classification which closes the gap between symbolic labels on the motion patterns and the underlying actions. The experiments are presented in Sect. 5. A summary can be found in Sect. 6.

2. Tracking Procedure

The input of our tracking procedure is a data stream of multi-view images (obtained by a set of calibrated and synchronized cameras) as well as a surface mesh of the subject to be tracked (obtained by a body laser scanner). We further assume that the mesh is rigged so that all mesh points are associated in a fixed way to the joints of an underlying kinematic chain. Then, the tracking problem consists of computing the configuration parameters (joint angles as well as root orientation and translation) of the kinematic chain from the given image data. Here, the surface mesh should be transformed with the configuration parameters in such a way that the projection of the mesh covers the observed subject in the images as accurately as possible.

2.1. Kinematic Chains

The subject to be tracked is modeled by a so-called *kinematic chain*, which is generally used to model a flexibly linked rigid body such as a human skeleton [4]. In the following, we use homogeneous coordinates to represent 3D points and exponential functions of twists to represent rigid body motions. The configuration of a kinematic chain can then be described by a consecutive evaluation of exponential functions of twists, see [4]. More precisely, let $x \in \mathbb{R}^3$ be a 3D coordinate of a joint in the neutral configuration (standard pose) of the kinematic chain. Let $X = \binom{x}{1}$ be the respective homogeneous coordinate and define π as the associated projection with $\pi(X) = x$. Furthermore, let ξ be a rigid body motion, which can be represented as $\xi = \exp(\theta\hat{\xi})$ with a twist $\hat{\xi}$ and $\theta \in \mathbb{R}$. The overall configuration of the kinematic chain is specified by a rigid body motion $\xi = \exp(\theta\hat{\xi})$ encoding the root orientation and translation as well as a sequence $\xi_1 = \exp(\theta_1\hat{\xi}_1), \ldots, \xi_n = \exp(\theta_n\hat{\xi}_n)$ of rigid body motions encoding the joint angles. Note that the twists $\hat{\xi}_1, \ldots, \hat{\xi}_n$ are fixed for a specific kinematic chain. Thus, the configuration of a fixed kinematic chain is specified by the following $(6+n)$ free parameters:

$$\chi := (\xi, \Theta) \;\; \text{with} \;\; \Theta := (\theta_1, \ldots, \theta_n). \tag{1}$$

In other words, the configuration parameter vector χ consists of the 6 degrees of freedom for the rigid body motion ξ and the joint angle vector Θ, see also [15]. Now, for a given point x on the kinematic chain, we define $\mathcal{J}(x) \subseteq \{1, \ldots, n\}$ to be the ordered set that encodes the joint transformations affecting x. Then, for a given configuration parameter vector $\chi := (\xi, \Theta)$, the point x is transformed according to

$$Y = \exp(\theta\hat{\xi}) \prod_{j \in \mathcal{J}(x)} \exp(\theta_j\hat{\xi}_j)X. \tag{2}$$

2.2. Pose Estimation

In our setup, the vector χ is unknown and has to be determined from the image data. In the following, instead of regarding points on the kinematic chain, we use points on the surface mesh. As the mesh is rigged, the mesh points are directly associated to a joint. Given a set of 3D surface mesh points x_i, $i \in I$, we assume for the moment that one knows corresponding 2D coordinates of these points within a given image. In Sect. 2.3, we describe how to obtain such correspondences. Furthermore, we represent each 2D point as a reconstructed projection ray given in 3D Plücker form $L_i = (n_i, m_i)$ [13]. For pose estimation, the basic idea is to apply the (unknown) rigid body motions on 3D points x_i according to χ and to claim incidence with the reconstructed projection rays. Due to the properties of Plücker lines, this incidence can be expressed as

$$\left(\pi\left(\exp(\theta\hat{\xi})\prod_{j\in\mathcal{J}(x_i)}\exp(\theta_j\hat{\xi}_j)\right)\times n_i\right)-m_i=0. \quad (3)$$

To simultaneously account for the incidences of all points x_i, $i \in I$, one minimizes the following term in a least-squares sense:

$$\underset{\chi}{\operatorname{argmin}}\sum_i\left\|\left(\pi\left(\exp(\theta\hat{\xi})\prod_{j\in\mathcal{J}(x_i)}\exp(\theta_j\hat{\xi}_j)X_i\right)\times n_i\right)-m_i\right\|_2^2 \quad (4)$$

To solve for the unknown parameters in the exponential functions, we linearize each function by using the first two elements of the respective Taylor series: $\exp(\theta\hat{\xi}) \approx \mathbb{1} + \theta\hat{\xi}$. This leads to three linear equations with $6 + n$ unknowns for each exponential function. In case of many correspondences (i. e., in case there are many mesh points x_i with correspondences), one obtains an over-determined linear system of equations, which can be solved in the least squares sense. The approximation errors introduced by the linearization step are handled by applying an iterative computation scheme, see [15] for details.

2.3. Region-based Pose Tracking

In our framework we use the region-based tracking approach as presented in [16], to which we refer to details. However, alternatively, one may also use other techniques as presented in [3, 6, 14].

The concept is to estimate pose parameters χ such that the projection of the resulting surface mesh optimally splits the image into a foreground (subject) and a background region. Here, the splitting is regarded as optimal if suitable image features (color, texture) are maximally dissimilar in the two regions with regard to estimated density functions, see [16]. Starting with a first estimate χ, the transformed mesh points y_i (see Eq. (2)) are projected onto image points p_i yielding correspondences in a natural way. One then considers only the visible image points p_i that lie on the contour line separating foreground and background. Next, these points are shifted inwards or outwards (orthogonal to the contour line) according to force vectors so that the resulting points, say q_i, better explain the color distributions of the foreground and background regions, see Fig. 3. Finally, using the points q_i with the correspondent mesh points x_i (obtained from the transformed mesh points y_i) one is then in the situation for applying the minimization (4) to obtain an improved estimation of the pose parameters. The entire process is iterated until convergence, see [16] for details.

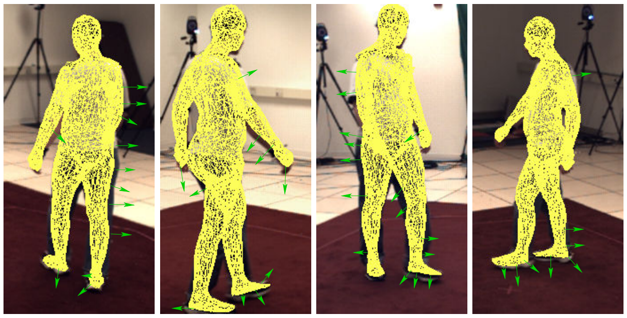

Figure 3. Example forces (enlarged force vectors, green) acting on the contour line of the projected surface mesh.

3. Retrieval Component

In this section, we review the concept of motion templates (MT) in Sect. 3.1, explain our MT-based classification procedure in Sect. 3.2 and finally describe the allocation of priors in Sect. 3.3.

3.1. Motion Templates

We now summarize the main idea of motion templates referring to [10] for details. As underlying feature representation, we use the concept of relational features that capture semantically meaningful Boolean relations between specified points of the kinematic chain underlying the mocap data, see [11]. In the following, we use a set of $f = 41$ relational features where the first 39 features are defined as in [10], and the last two features express whether the right/left foot is moving to the front. Then, a given mocap sequence is converted into a sequence of f-dimensional Boolean feature vectors in a framewise fashion. Denoting the number of frames by K, we think of the resulting sequence as a *feature matrix* $X \in \{0,1\}^{f\times K}$. An example is shown in Fig. 4 (b), where, for the sake of clarity, we display a subset comprising only 6 of the $f = 41$ features.

In our scenario, we assume that each motion category is given by a class $\mathcal{C}$ consisting of $\gamma \in \mathbb{N}$ logically related example motions. To learn a motion class representation that grasps the essence of the class, we compute a semantically meaningful average over the γ feature matrices of training examples. Here, to cope with temporal variations in the example motions, we use an iterative warping and averaging algorithm [10], which converges to an output matrix $X_{\mathcal{C}}$ referred to as a *motion template* (MT) for the class $\mathcal{C}$. After a subsequent quantization step, one obtains a *quantized MT* with values over the set $\{0, 1, *\}$ as indicated by Fig. 4 (a). The length (number of columns) of the MT corresponds to the average length of the training motions. The black/white regions in a class MT indicate periods in time (horizontal axis) where certain features (vertical axis) consistently assume the same values zero/one in all training motions, respectively. By contrast, gray regions (corresponding to the

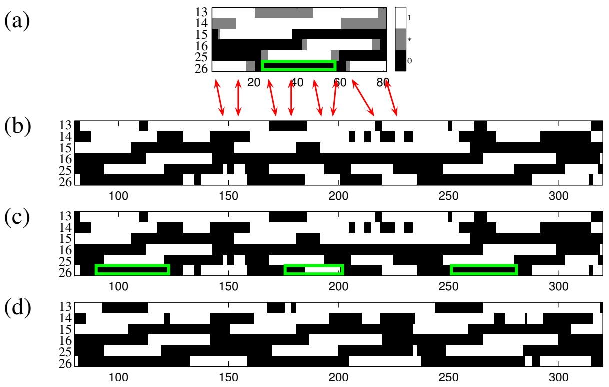

Figure 4. **(a)**: Marked motion template for motion class 'walk2StepsLstart'. The feature numbers correspond to the features used in [10]. The annotation for the relational constraint 'leftFootOnGround' is indicated by the green rectangle. **(b)**: Feature matrix for the subsegment of D_{EVA} consisting of the first three walking cycles. The second walking cycle (frames 160 to 240) has not been tracked correctly. **(c)**: Feature matrix (b) with allocated priors (green rectangles). **(d)**: Feature matrix after regularized tracking using the priors of (c).

wildcard character $*$) indicate inconsistencies mainly resulting from variations in the training motions. In other words, the black/white regions encode characteristic aspects that are shared by all motions, whereas the gray regions represent the class variations coming from different realizations. For further details, we refer to [10].

3.2. Classification Procedure

Given a mocap sequence D and a specific motion class $\mathcal{C}$, we now define a distance function that reveals all motion subsegments of D correlating to $\mathcal{C}$. Let $X \in \{0,1,*\}^{f \times K}$ be a quantized class MT of length K and $Y \in \{0,1\}^{f \times L}$ the feature matrix of D of length L. We define for $k \in [1:K]$ and $\ell \in [1:L]$ a local cost measure $c^Q(k,\ell)$ between the k-th column $X(k)$ of X and the ℓ-th column $Y(\ell)$ of Y. Let $I(k) := \{i \in [1:f] \mid X(k)_i \neq *\}$, where $X(k)_i$ denotes the i^{th} entry of the k^{th} column of X. Then, if $|I(k)| > 0$, we set

$$c^Q(k,\ell) = \frac{1}{|I(k)|} \sum_{i \in I(k)} |X(k)_i - Y(\ell)_i|, \tag{5}$$

otherwise we set $c^Q(k,\ell) = 0$. In other words, $c^Q(k,\ell)$ only accounts for the consistent entries of X with $X(k)_i \in \{0,1\}$ and leaves the other entries unconsidered. Based on this local distance measure and a subsequence variant of dynamic time warping (DTW), one obtains a distance function $\Delta_{\mathcal{C}} : [1:L] \to \mathbb{R} \cup \{\infty\}$ as described in [10] with the following interpretation: a small value $\Delta_{\mathcal{C}}(\ell)$ for some $\ell \in [1:L]$ indicates the presence of a motion subsegment of D that is similar to the motions in $\mathcal{C}$ starting at a suitable frame index $a_\ell < \ell$ and ending at frame index ℓ . Here, the starting frame index a_ℓ can be recovered by a simple backtracking within the DTW procedure. In other words, looking for all local minima in $\Delta_{\mathcal{C}}$ below a suitable matching threshold $\tau > 0$ one can identify all subsegments of D that are similar to the class MT. As example, Fig. 5 (a) shows a distance function based on the quantized MT for the class 'walk2StepsLstart' for $D = D_{\mathrm{EVA}}$. Note that each of the first five local minima (frames 0 to 400) reveals the end of a walking cycle starting with the left foot.

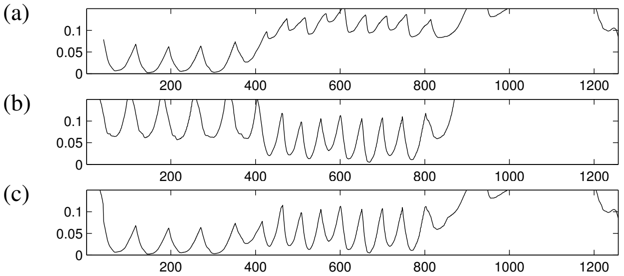

Figure 5. Distance function $\Delta_{\mathcal{C}}$ for D_{EVA} with respect to the class **(a)** 'walk2StepsLstart', **(b)** 'jog2StepsLstart', and **(c)**: Combined distance function $\Delta^{\min}$ obtained by minimizing (a) and (b).

Now, let $\mathcal{C}_1, \ldots \mathcal{C}_P$ be the available motion classes, where $p \in [1:P]$ denotes the class label of class $\mathcal{C}_p$. Then, given a mocap sequence D of length L, the classification task is to identify all motion subsegments within D that belong to one of the P classes. To this end, we compute a distance function $\Delta_p := \Delta_{\mathcal{C}_p}$ for each class $\mathcal{C}_p$ and minimize the resulting functions over $p \in [1:P]$ to obtain a single function $\Delta^{\min} : [1:L] \to \mathbb{R} \cup \{\infty\}$:

$$\Delta^{\min}(\ell) := \min_{p \in [1:P]} \Delta_p(\ell), \tag{6}$$

$\ell \in [1:L]$. Furthermore, we store for each frame the minimizing index $p \in [1:P]$ yielding a function $\Delta^{\arg} : [1:L] \to [1:P]$ defined by:

$$\Delta^{\arg}(\ell) := \operatorname*{argmin}_{p \in [1:P]} \Delta_p(\ell). \tag{7}$$

The function $\Delta^{\arg}$ yields the local classification of the mocap sequence D by means of the class labels $p \in [1:P]$. Fig. 5 shows an example-based on $P = 2$ motion classes.

3.3. Allocation of Priors

We now explain how to generate suitable motion priors, which can then be used to regularize the tracking process. Recall that a class motion template $X_{\mathcal{C}}$ explicitly encodes characteristic motion aspects (corresponding to black/white regions) that are typically shared by motions of class $\mathcal{C}$. We select some of these aspects by marking suitable entries within the template $X_{\mathcal{C}}$. These entries are also referred to as *MT priors*. As an example, consider Fig. 4 (a), where the

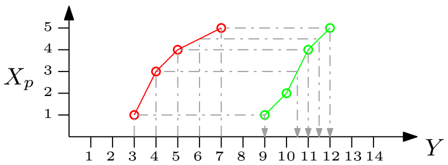

Figure 6. Pose priors are allocated by looking for subsequences in Y that all align to the same MT X_p.

entries of row 26 between columns 22 and 58 are marked by the green rectangle. Feature 26 expresses whether the right foot rests (black, value 0) or assumes a high velocity (white, value 1). Since all entries have the value 0 within the green rectangle, this MT prior basically expresses that the right foot rests (stays on the ground) during this phase of the motion. Note that the MT priors are part of the database knowledge and do not depend on the sequence to be tracked.

Now, let D be a mocap sequence of length L obtained from some previous tracking procedure and let $Y \in \{0,1\}^{f\times L}$ be the corresponding feature matrix. The goal is to automatically transfer suitable MT priors to the tracked sequence to obtain what we refer to as *tracking priors*. Let $\mathcal{C}_1,\ldots\mathcal{C}_P$ be the available motion classes with corresponding motion templates $X_p = X_{\mathcal{C}_p}$, $p \in [1:P]$, each equipped with suitable MT priors. We compute the functions $\Delta^{\min}$ and $\Delta^{\arg}$ as described in Sect. 3.2. Recall that a local minimum $\ell \in [1:L]$ of $\Delta^{\min}$ close to zero indicates the presence of a motion subsegment of D (starting at a suitable frame index $a_\ell < \ell$ and ending at frame index ℓ) that corresponds to motion class $\Delta^{\arg}(\ell) \in [1:P]$. Therefore, we fix a quality threshold $\tau > 0$ and look for all essential local minima $\ell \in [1:L]$ with $\Delta^{\min}(\ell) < \tau$. (Here, *essential* means that we only consider one local minimum within a suitable temporal window to avoid local minima being too close to each other.) Using the same DTW procedure as in Sect. 3.2, we then derive an alignment between the motion template X_p with $p := \Delta^{\arg}(\ell)$ and the feature subsequence of Y ranging from a_ℓ to ℓ. Fig. 4 shows an example, where the alignment is indicated by the red arrows. Note that such an alignment establishes temporal correspondences between semantically related frames and thus allows for transferring the MT priors within X_p to corresponding regions within Y, see Fig. 4 (c). These regions, in the following referred to as *tracking priors*, are then used for regularization in the tracking procedure (Sect. 4).

As an additional stabilizing factor, we take further advantage out of several subsequences of Y that all align to the same MT X_p. The idea is to use X_p as a kind of mediator to generate additional priors from the multiply aligned subsequences. We explain this idea by means of a simple example consisting of two subsegments as indicated by Fig. 6. Here, each circle denotes a correspondence (x_ℓ, ℓ) between frame x_ℓ in X_p and frame $\ell \in [1:L]$ of Y. In this example, essential local minima were found for frames 7 and 12 (matching to the subsequences ranging from frames 3 to 7 and from 9 to 12). Now, suppose that the green alignment has a cost close to zero ($\Delta^{\min}(12) \approx 0$). In practice, such an alignment corresponds to a subsequence of Y that does not contain tracking errors. By contrast, suppose that the subsequence corresponding to the red alignment contains some tracking errors resulting in higher alignment cost. Then, the idea is to use the poses of the "green subsequence", referred to as *pose priors*, to stabilize the tracking of the "red subsequence". The correspondence of poses between the subsequences is established via the alignments to X_p, see Fig. 6. For example, frame 9 of Y yields a pose prior for frame 3 of Y since both frames are aligned to the first frame of X_p (indicated by the dashed arrow line). To put it in simple words, we first detect the presence of repetitions within Y by means of the MT-based local classification and then generate pose priors from the established correspondences.

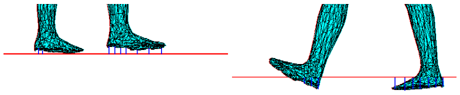

Figure 7. Integration of the constraint *foot on floor*. Points on the sole are pushed onto the floor.

4. Integration of Allocated Priors in Tracking

As explained in Sect. 3.3, a retrieval component is used to allocate two types of priors to the tracking sequence. In the following, we show how the allocated priors are integrated into a subsequent tracking iteration.

Tracking priors provide information about certain movement behaviors of body parts within a certain motion context. As an example, we consider the tracking prior "left foot should be on the floor for a certain frame". We use soft constraints to integrate this information in the tracking framework, where the influence of a prior can be controlled by a weighting parameter. In particular, soft constraints are formulated as additional equations that are included in the minimization step (4). To implement the example tracking prior, all points $y_s, s \in S \subset I$, on the sole of the foot (the plantar) are projected onto the ground plane, yielding the points z_s. Then, we claim incidence $y_s - z_s = 0$ for $s \in S$, to push the sole onto the ground plane, see Fig. 7. Using Eq. (2) to express y_s by the underlying kinematic chain, we integrate the set of equations

$$\pi\Big(\exp(\theta\hat{\xi})\prod_{j\in\mathcal{J}(y_s)}\exp(\theta_j\hat{\xi}_j)X_s\Big) - z_s = 0, \qquad s \in S \tag{8}$$

into the minimization step (4). Note that the unknowns are the same as for (4), as z_s are considered as constants for one frame. In a similar manner it is straightforward to integrate motion dynamics like arms swinging forward or backwards.

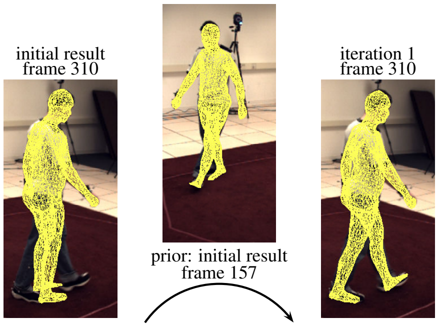

Figure 8. Using pose priors to regularize tracking.

Unlike tracking priors, *pose priors* denote that a certain joint angle configuration Θ at frame $\ell_1 \in [1 : L]$ should also be assumed in frame $\ell_2 \in [1 : L] \setminus \{\ell_1\}$. For example, consider Fig. 8 where the generated pose prior suggests to take Θ of frame $\ell_1 = 157$ for frame $\ell_2 = 310$. To this end, equations similar to Eq. (8) can be integrated into the minimization step (4) to regularize the joint angle configuration at frame ℓ_2 towards Θ in the subsequent tracking iteration.

5. Experiments

In our experiments, we used the Human EVA-II benchmark [19]. Here, a surface model, calibrated multiview image sequences of four cameras, and background images are provided. Note, that our region-based pose tracking does not rely on background subtraction and therefore the background information is not used in our method. Instead, we rely on the image data, projection matrices and a mesh model. Due to color similarities and the sparse number of cameras, tracking is challenging and the results are likely to be corrupted if no priors are involved. Tracking results (as 3D marker positions) can be uploaded to a server at Brown University for evaluation. As the sequence has been captured in parallel with a marker-based tracking system (using a Vicon system), an automated script can evaluate the accuracy of a tracking result in terms of relative errors in millimeters. In the Human EVA-II sequence $S4$, three different actions are performed consecutively, lasting for 6.7 s (400 frames at 60 Hz) each. A non-professional actor walks in a circle, jogs in a circle, and then balances on each foot. We decided for this sequence for several reasons: Firstly, it is a public available benchmark, which allows a quantitative comparison to other existing approaches. Secondly, the sequence contains three different patterns and we want to test, whether our system is able to classify and single out the involved motions correctly (walking and jogging). Thirdly, walking and jogging are similar patterns, which allows us to get a good feeling about the sensitivity of our approach in classifying similar patterns. Fourthly, the balancing part is not in the database, which means the algorithm should not perform a classification at all, so that the tracking is only driven from the image data without any priors. All these aspects can be covered by this sequence.

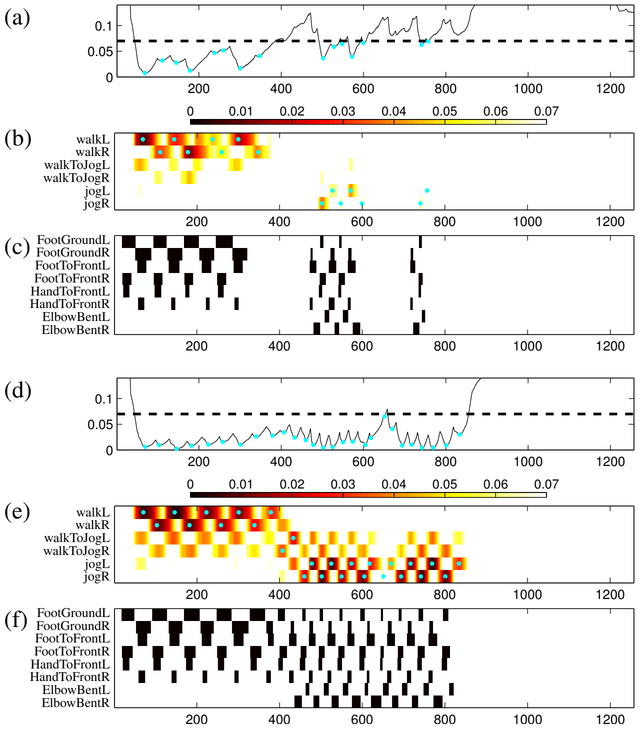

Figure 9. **(a)**: $\Delta^{\min}$ for the initial tracking result. Essential local minima are marked by a cyan dot. **(b)**: Corresponding distance functions Δ_C for six MTs shown in a color coded fashion. Values greater than $\tau = 0.7$ are drawn in white. **(c)**: Allocated tracking priors. **(d)**-**(f)**: Corresponding plots after the fourth iteration.

The database knowledge that is used by the retrieval system is generated in a preprocessing step. To this end, we assembled a total of 232 short 3D motion capture clips, which we manually cut out from the freely available HDM05 mocap database [12] (obtained from a Vicon system). The mocap clips of an average length of 1.1 s were categorized into $P = 6$ different motion categories, which are 'walk two steps', 'jog two steps', and 'change from walk to jog', each for starting with the left and right foot, respectively.

After extracting the relational features for each example motion at a sampling rate of 60 Hz, we computed a quantized motion template classifier for each of the P motion classes and marked suitable regions in the quantized MTs as MT priors, see Fig. 4 (a). In our scenario, we marked MT priors corresponding to 'left/right foot is on ground', 'left/right foot moves to front', 'left/right hand moves to front', and 'left/right elbow is bent'. Note that the set of motion templates along with the MT priors, which constitutes our database knowledge, is independent of the sequence to be tracked and has to be generated only once.

In the initialization step, tracking is performed without

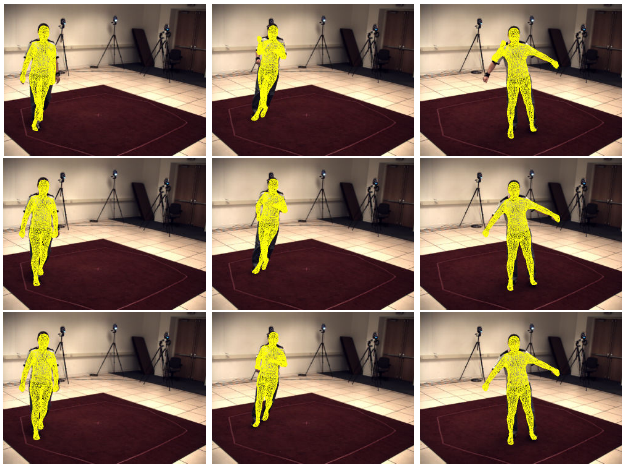

Figure 10. Improvements obtained by our iterative tracking procedure. **Top:** Result without prior knowledge (initialization). **Middle:** Result after the first iteration. **Bottom:** Result after the third iteration. The frames from left to right show examples of the walking, jogging, and balancing part (frames 210, 750 and 1170). Several tracking errors (see arms and legs) are corrected.

using any regularizing priors. The resulting tracking sequence is then locally classified according to the precomputed MTs. In our experiments, a quality threshold of $\tau = 0.07$ turned out to be a robust choice. In Fig. 9 (a), we show the resulting $\Delta^{\min}$. Essential local minima below $\tau = 0.07$ are marked by a cyan dot. Note that for the walking part of the benchmark sequence (frames 1 to 400), $\Delta^{\min}$ assumes lower values than for the jogging part (frames 400 to 800), which indicates that the walking part contains less tracking errors than the jogging part. Note also that for the balancing part (frames 830 to 1200), $\Delta^{\min}$ is far above τ revealing a strong difference to walking or jogging patterns. The function $\Delta^{\arg}$ assigns the essential local minima to appropriate motion categories, see Fig. 9 (b). Furthermore, each motion subsequence induced by a local minimum is aligned to the corresponding MT. Based on these local alignments, suitable tracking priors are allocated for the next iteration, see Fig. 9 (c). Fig. 9 (d)-(f) show the corresponding distance functions and allocated priors in the fourth iteration. Note that the local minima of $\Delta^{\min}$ shown in Fig. 9 (d) have gained a substantial qualitative boost. Furthermore, the occurrence of the different motion categories are revealed in a much more distinctive way in the fourth iteration, compare (e) and (b) of Fig. 9. This all indicates a stabilization of the tracking procedure over the iterations.

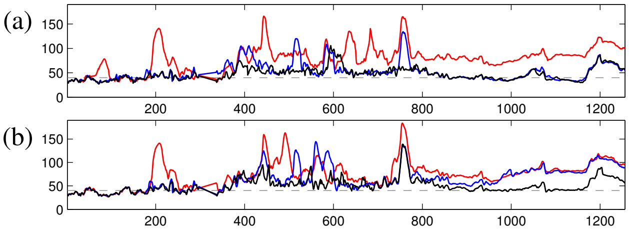

Figure 11. Framewise tracking error (in millimeters) in the initialization (red), first (blue), and third iteration (black). **(a)**: Without image noise. **(b)**: With Gaussian noise (40 pixels standard deviation) added to each frame.

	no additional noise			+40 px image noise		
	∅	σ	max	∅	σ	max
Initial step	79.1	26.2	165.9	75.2	28.2	184.0
Iteration 1	51.8	18.5	134.1	65.0	24.2	144.4
Iteration 3	47.9	12.8	105.8	51.0	15.5	139.0

Table 1. Improvements of the tracking quality over various iterations. Average errors, standard deviations, and maximal errors (in millimeter) over all 1257 frames of the sequence are shown.

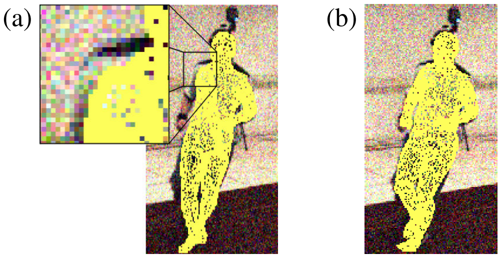

Figure 12. Gaussian noise has been added to each image. Pose overlay of frame 500 after the initial tracking **(a)** and after the third iteration **(b)**. During the iterations, several tracking errors (see right arm and both legs) are corrected.

We now discuss the actual improvements in the tracking results achieved by our novel iterative approach. Fig. 10 shows representative poses overlayed with the tracking result (indicated by the yellow meshes) after the initial step, the first iteration, and the third iteration. As seen, the initial tracking result contains various serious tracking errors such as a swap of legs or an incorrect inflection of the arms. These errors are corrected within few iterations. The absolute difference of the 3D joint positions of the tracking result and the ground truth positions are indicated by Table 1 and by Fig. 11. These numbers were obtained by the automated evaluation system supplied by Brown University [19]. During the iterations, the average error is reduced from 79 mm to 48 mm after few iterations, see Table 1. The significant improvements are also indicated by Fig. 11 (a).

In a second experiment we added Gaussian noise (standard deviation: 40 pixels) to each frame. Two example frames after the initialization and the third iteration are shown in Fig. 12. During iterations, the average error dropped from 75 mm to 51 mm, see Table 1 and Fig. 11 (b). These results demonstrate the stabilizing effects achieved by our iterative tracking approach. Note that our framework requires that a sequence is tracked several times. Currently, our tracking implementation requires 7 s per frame resulting in 2.5 h for the entire 1257 frames. After tracking, the classification and allocation steps require 15 s.

6. Summary

In this paper, we introduced an iterative tracking approach that dynamically integrates motion priors retrieved from a database to stabilize tracking. Intuitively, our idea is to pursue a joint bottom-up and top-down strategy in the sense that we start with a rough initial tracking which is then improved by incorporating high-level motion cues. These motion cues are allocated upon a local classification of the initial tracking result. In addition to stabilization, the local classification can also be used for automatic motion annotation. By means of the HumanEVA-II benchmark, we showed that even simple motion priors lead to significant improvements in the tracking. There are still limitations in our approach. In particular, the presence of strong tracking errors may lead to a confusion in the local classification; misallocated priors may then worsen the tracking error. In future work, we plan to develop techniques that can cope with such situations, e. g., by integrating statistical confidence measures for the classification and by simultaneously considering alternative motion priors.

Acknowledgments. The first and second author are funded by the German Research Foundation (DFG CL 64/5-1, RO 2497/6-1), the third author by the Cluster of Excellence on Multimodal Computing and Interaction.

References

[1] A. Agarwal and B. Triggs. Recovering 3D human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):44–58, Jan. 2006.

[2] A. Balan, L. Sigal, M. Black, and H. Haussecker. Shining a light on human pose: On shadows, shading and the estimation of pose and shape. In *Proc. Intern. Conf. on Computer Vision*, 2007.

[3] M. Bray, P. Kohli, and P. Torr. Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In A. Leonarids, H. Bishof, and A. Prinz, editors, *Proc. 9th European Conference on Computer Vision, Part II*, volume 3952 of *LNCS*, pages 642–655, Graz, May 2006. Springer.

[4] C. Bregler, J. Malik, and K. Pullen. Twist based acquisition and tracking of animal and human kinetics. *International Journal of Computer Vision*, 56(3):179–194, 2004.

[5] M. Brubaker, D. J. Fleet, and A. Hertzmann. Physics-based person tracking using simplified lower-body dynamics. In *Conference of Computer Vision and Pattern Recognition (CVPR)*, Minnesota, 2007. IEEE Computer Society Press.

[6] S. Dambreville, A. Yezzi, R. Sandhu, and A. Tannenbaum. Robust 3d pose estimation and efficient 2d region-based segmentation from a 3d shape prior. In *Proc. 10th European Conference on Computer Vision*, LNCS. Springer, 2008.

[7] L. Herda, R. Urtasun, and P. Fua. Implicit surface joint limits to constrain video-based motion capture. In T. Pajdla and J. Matas, editors, *Proc. 8th European Conference on Computer Vision*, volume 3022 of *LNCS*, pages 405–418, Prague, 2004. Springer.

[8] J. Liu and M. Shah. Learning human actions via information maximization. In *Conference of Computer Vision and Pattern Recognition (CVPR)*, Anchorage, 2008. IEEE Computer Society Press.

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

[10] M. Müller and T. Röder. Motion templates for automatic classification and retrieval of motion capture data. In *Proc. of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 137–146. ACM Press, 2006.

[11] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. *ACM Trans. Graph.*, 24(3):677–685, 2005.

[12] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation: Mocap Database HDM05. Computer Graphics Technical Report CG-2007-2, Universität Bonn, June 2007. `http://www.mpi-inf.mpg.de/resources/HDM05`.

[13] R. Murray, Z. Li, and S. Sastry. *Mathematical Introduction to Robotic Manipulation.* CRC Press, Baton Rouge, 1994.

[14] B. Rosenhahn, T. Brox, and J. Weickert. Three-dimensional shape knowledge for joint image segmentation and pose tracking. *International Journal of Computer Vision*, 73(3):243–262, 2007.

[15] B. Rosenhahn, C. Schmaltz, T. Brox, J. Weickert, D. Cremers, and H.-P. Seidel. Markerless motion capture of man-machine interaction. In *Conference of Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society Press, 2008.

[16] C. Schmaltz, B. Rosenhahn, T. Brox, D. Cremers, J. Weickert, L. Wietzke, and G. Sommer. Region-based pose tracking. In J. Martí, J. M. Benedí, A. M. Mendonça, and J. Serrat, editors, *Pattern Recognition and Image Analysis*, volume 4478 of *LNCS*, pages 56–63, Girona, Spain, June 2007. Springer.

[17] G. Shakhnarovich, P. Viola, and T. Darell. Fast pose estimation with parameter sensitive hashing. In *Proc. International Conference on Computer Vision*, pages 750–757, Nice, France, Oct. 2003.

[18] H. Sidenbladh, M. J. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Proc. European Conference on Computer Vision*, volume 2353 of *LNCS*, pages 784–800. Springer, 2002.

[19] L. Sigal and M. Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical Report CS-06-08, Brown University, USA, 2006. Available at `http://vision.cs.brown.edu/humaneva/` .

[20] C. Sminchisescu and A. Jepson. Generative modeling for continuous non-linearly embedded visual inference. In *Proc. International Conference on Machine Learning*, 2004.

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

[22] D. Tran and A. Sorokin. Human activity recognition with metric learning. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *10th European Conference on Computer Vision, ECCV (1)*, volume 5302 of *Lecture Notes in Computer Science*, pages 548–561. Springer, 2008.

[23] R. Urtasun and T. Darrell. Local probabilistic regression for activity-independent human pose inference. In *Conference of Computer Vision and Pattern Recognition (CVPR)*, Anchorage, 2008. IEEE Computer Society Press.

[24] R. Urtasun, D. J. Fleet, and P. Fua. 3D people tracking with Gaussian process dynamical models. In *Conference of Computer Vision and Pattern Recognition (CVPR)*, pages 238–245. IEEE Computer Society Press, 2006.

[25] M. Vondrak, L. Sigal, and O. C. Jenkins. Physical simulation for probabilistic motion tracking. In *Conference of Computer Vision and Pattern Recognition (CVPR)*, Anchorage, 2008. IEEE Computer Society Press.