UNIVERSITY OF COPENHAGEN



Københavns Universitet

Three dimensional monocular human motion analysis in end-effector space

Hauberg, Søren; Lapuyade, Jerome; Engell-Nørregård, Morten Pol; Erleben, Kenny; Pedersen, Kim Steenstrup

Published in:
Energy Minimization Methods in Computer Vision and Pattern Recognition

DOI:
10.1007/978-3-642-03641-5_18

Publication date:
2009

Document version
Peer reviewed version

Citation for published version (APA):
Hauberg, S., Lapuyade, J., Engell-Nørregård, M. P., Erleben, K., & Pedersen, K. S. (2009). Three dimensional monocular human motion analysis in end-effector space. In D. Cremers, Y. Boykov, A. Blake, & F. R. Schmidt (Eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition: 7th International Conference, EMMCVPR 2009, Bonn, Germany, August 24-27, 2009. Proceedings* (pp. 235-248). Springer. https://doi.org/10.1007/978-3-642-03641-5_18

Download date: 21. Mar. 2026

Three Dimensional Monocular Human Motion Analysis in End-Effector Space

Søren Hauberg, Jerome Lapuyade, Morten Engell-Nørregård, Kenny Erleben, and Kim Steenstrup Pedersen

{hauberg, lapuyade, mort, kenny, kimstp}@diku.dk,
The eScience Center, Dept. of Computer Science, University of Copenhagen

Abstract. In this paper, we present a novel approach to three dimensional human motion estimation from monocular video data. We employ a particle filter to perform the motion estimation. The novelty of the method lies in the choice of state space for the particle filter. Using a non-linear inverse kinematics solver allows us to perform the filtering in end-effector space. This effectively reduces the dimensionality of the state space while still allowing for the estimation of a large set of motions. Preliminary experiments with the strategy show good results compared to a full-pose tracker.

1 Introduction

Three dimensional human motion analysis is the process of estimating the configuration of body parts over time from sensor input [1]. One approach to this estimation is to use motion capture equipment where electromagnetic markers are attached to the body and then tracked in three dimensions. While this approach gives accurate results, it is intrusive and cannot be used outside laboratory settings.

Our long-term goal is to use human motion analysis as part of a physiotherapeutic rehabilitation system where the motion of a patient is tracked and analysed during exercise sessions performed both at the hospital and at the patient's home. The motion information will then be used to provide real-time feedback to the patient and to collect statistics on the patient's progress. The system should serve as an aid to the patient while performing a self-training programme at home, in that the patient will get instant response on whether or not the exercise is performed optimally. Furthermore, the system may act as a progress measurement tool for the physiotherapist. It is essential to develop systems that can be used by the patient at home, which rules out marker-based systems and hard-to-calibrate multi-camera solutions. Thus, the focus is on developing monocular vision-based systems.

In this paper, we present a novel approach to three dimensional monocular human motion estimation. The novelty of the system lies in the choice of state space. Monocular motion estimation in three dimensions is an inherently ill-posed problem since the observed images are two dimensional. This manifests

itself in that the distribution of the human pose is multi-modal with an unknown number of modes. To reliably estimate this distribution we need methods that cope well with multi-modal distributions. Currently, the best method for such problems is the particle filter [2], which represents the distribution as a set of weighted samples. Unfortunately, the particle filter is smitten by the curse of dimensionality in that the necessary number of samples grow exponentially with the dimensionality of state space. The consequence is that the particle filter is only applicable to low dimensional state spaces. This is in direct conflict with the fact that the human body has a great number of degrees of freedom.

Our approach is inspired by recent results from the world of animation [3]. Here animators are faced with the task of posing human figures on a frame by frame basis. This time-consuming task has created the need for models that allow animators to create realistic looking figures using few parameters. *Inverse kinematic* models allow the animator to pose only a few selected parts of the human figure, called *end-effectors*, while the remaining parts can be positioned by solving a non-linear least-squares optimisation problem. The end-effectors are most often the head, the hands and the feet of the figure. Although the underlying optimisation problem is hard to solve, recent work [3] has shown that realistic looking animations can be created in real-time using inverse kinematics. In this paper, we investigate the usefulness of estimating human motion in end-effector space rather than full-pose space. This is done using inverse kinematics to infer the full pose from end-effector positions.

To simplify the measurement model of the particle filter, our measurement model is based on a simple Markov random field texture model for each limb of the current pose model. We do not attempt to handle self-occlusions in this paper as focus is on the choice of state space. When necessary, this can be introduced later by appropriately changing the measurement model.

Our main point in this paper is to show the feasibility of tracking in end-effector space compared to tracking in full-pose space and argue that the former approach allows for real-time implementations with reasonable accuracy.

1.1 Related Work

Much work has gone into human motion analysis. The bulk of the work is in locating the position of moving humans in image sequences and classifying their actions. It is, however, beyond the scope of this paper to give a review of this work. The interested reader can consult review papers such as [4].

In recent years, much work has gone into more detailed visual human motion analysis in three dimensions [1]. The main difficulty in this area is the high number of degrees of freedom in the human body, which gives rise to high dimensional state spaces. To overcome this problem, many researchers reduce the dimensionality of state space using manifold learning [5,6,7]. The basic idea is to learn a manifold from motion capture data and then perform the motion analysis on this manifold. It seems that most researchers taking this route focus on simple low-dimensional motions, such as *walking* [7,8,9,10], *golf swings* [9,10],

tennis playing [11] etc. In this context, end-effector tracking using inverse kinematics can be interpreted as a dimensionality reduction technique that is based on domain specific knowledge about general human motion.

While kinematic skeleton models have been used a lot in motion analysis it seems that inverse kinematics have been given little attention. The approach closest to ours is that of Sminchisescu and Triggs [12] who use inverse kinematics to enumerate possible interpretations of the input, which results in a more efficient sampling scheme for their particle filter. They, however, still perform the motion analysis in full-pose space, whereas we work in end-effector space. This provides a very efficient way of reducing the dimensionality of the state space that still allows us to work with a large class of motions.

1.2 Organisation of the Paper

This paper is organised as follows. In the next two sections the theoretical background is presented. First an introduction to inverse kinematics is given in Sec. 2. A brief introduction to particle filtering is then given in Sec. 2. In Sec. 3 we discuss two choices of the state space for the motion analysis system and in Sec. 4 we describe the measurement system used in our implementation. Preliminary results and a discussion of the pros and cons of the method are presented in Sec. 5 and the paper is concluded in Sec. 6.

2 Posing with Inverse Kinematics

Inverse kinematics is the problem of manipulating the pose of a skeleton in order to achieve a desired pose disregarding inertia and forces. The problem can be posed as a non-linear optimisation problem.

In the context of human modeling a skeleton is often modeled as a collection of rigid bodies connected by rotational joints of 1–3 degrees of freedom such as the one shown in Fig. 1. All joints are constrained in their rotation, as exemplified by joint i in Fig. 1 with l_i and u_i showing the limits of the angle θ_i. To compute the position and orientation of a joint in space we perform a transformation of the bone relative to its parent joint. The transformation consist of a rotation and a translation corresponding to the shape and orientation of the joint relative to its parent. These transformations are then nested to create chains of joints. Each chain ends in an end-effector, which can be regarded as the handle for controlling the chain. Thus, the full transformation of a joint from local space to global space can be performed.

The problem can be formally stated as follows. Given the set of joint parameters $\boldsymbol{\theta}$ we can change the values of $\boldsymbol{\theta}$ and gain explicit control over all joint angles. This in turn controls the position and orientation of the end-effector $\boldsymbol{a} = \boldsymbol{F}(\boldsymbol{\theta})$. This is commonly known as forward kinematics. Given a desired end-effector goal position, $\boldsymbol{g}$, one seeks the value of $\boldsymbol{\theta}$ such that

$$\boldsymbol{\theta} = \boldsymbol{F}^{-1}(\boldsymbol{g}) \ . \tag{1}$$

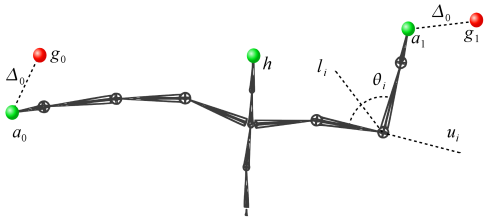


Fig. 1. An illustration of the kinematic model. End-effector positions are shown as green dots, while the desired positions (goals) are shown as red dots.

This is known as inverse kinematics. Closed form solutions exist for models with less than 7 degrees of freedom. However, the human body has a lot more than 7 degrees of freedom and has a large degree of interdependency between joints. Thus, it makes sense to solve the problem globally for the entire skeleton. High performance implies the need for iterative numerical methods. By posing the problem as a constrained least-squares fitting problem we are given a number of possible methods to solve the problem. Given a skeleton containing K kinematic chains, each with exactly one end-effector, we agglomerate the K end-effector functions into one function

$$\boldsymbol{a} = \left[\boldsymbol{a}_1^T \dots \boldsymbol{a}_K^T\right]^T = \left[\boldsymbol{F}_1(\boldsymbol{\theta})^T \dots \boldsymbol{F}_K(\boldsymbol{\theta})^T\right]^T = \boldsymbol{F}(\boldsymbol{\theta}) \ , \tag{2}$$

where $\boldsymbol{a}_j$ is the world coordinate position of the j^{th} end-effector and $\boldsymbol{F}_j(\boldsymbol{\theta})$ is the end-effector function corresponding to the j^{th} kinematic chain. Using the agglomerated end-effector function, we create the objective function

$$f(\boldsymbol{\theta}) = (\boldsymbol{g} - \boldsymbol{F}(\boldsymbol{\theta}))^T(\boldsymbol{g} - \boldsymbol{F}(\boldsymbol{\theta})) \ , \tag{3}$$

where $\boldsymbol{g} = \left[\boldsymbol{g}_1^T \dots \boldsymbol{g}_K^T\right]^T$ is the agglomerated vector of end-effector goals. The optimisation problem is then

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \quad \text{s.t.} \quad \boldsymbol{l} \leq \boldsymbol{\theta} \leq \boldsymbol{u} \ . \tag{4}$$

Here $\boldsymbol{l}$ is a vector containing the minimum joint limits and $\boldsymbol{u}$ is a vector of the maximum joints limits. For a more thorough description of the inverse kinematics problem and the constraint model see [13,3]. Any constrained non-linear optimisation method may be used to solve the problem. In this paper we use a simple, yet effective, gradient projection method with line search [14]. To compute the gradient we need the Jacobian matrix of $f(\boldsymbol{\theta})$ denoted $\boldsymbol{J}$. For rotational joints this can be easily computed. For each chain, the Jacobian matrix contains a 3×1 entry for each rotational degree of freedom. This entry is computed as the cross product of the rotational axis and the vector from the joint to the end-effector as shown in Fig. 2. Given this Jacobian matrix the gradient can be computed as $(\boldsymbol{g} - \boldsymbol{a})\boldsymbol{J}^T$. A more thorough description of the calculation of the Jacobian can be found in [15].

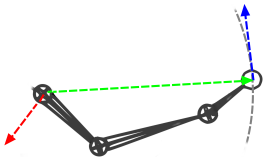


Fig. 2. Finding the rotational derivative of a joint. The derivative is found as the cross product of the rotational axis (red) and the vector from joint to end-effector (green) the resulting tangent vector is shown in blue.

Bayesian Filtering Bayesian filtering is concerned with estimating the unobserved state of a system from observations. In terms of human motion analysis, it is concerned with estimating a sequence of human poses given a sequence of images. This section provides a brief overview of the topic. For more details, the interested reader should consult papers such as [2] and the references therein.

In Bayesian filtering it is assumed that the observation I_t at time t is solely governed by a hidden variable $\boldsymbol{s}_t$. The distribution of this variable is in turn assumed to form a Markov chain, such that

$$p(\boldsymbol{s}_{1:T}, I_{1:T}) = p(\boldsymbol{s}_1)p(I_1|\boldsymbol{s}_1) \prod_{t=1}^{T-1} p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t)p(I_{t+1}|\boldsymbol{s}_{t+1}) \ . \tag{5}$$

Here $\boldsymbol{s}_{1:T} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T\}$ denotes a sequence of state variables and likewise for $I_{1:T}$.

The objective of estimating the current state given all observations is expressed as estimating the *filtering distribution* $p(\boldsymbol{s}_t|I_{1:t})$. This can be estimated recursively as [2]

$$p(\boldsymbol{s}_t|I_{1:t}) \propto \int p(\boldsymbol{s}_{t-1}|I_{1:t-1})p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})p(I_t|\boldsymbol{s}_t)\mathrm{d}\boldsymbol{s}_{t-1} \ . \tag{6}$$

Unfortunately, this integral can only be computed exactly in simple cases. For instance, if the state is finite valued the corresponding algorithm is called Baum-Welsh filtering [16] and if the model is linear and Gaussian, it can be computed by the Kalman filter [17]. In more general cases approximations are necessary. In recent years the particle filter [2] has seen growing popularity. It represents the filtering distribution as a set of weighted samples, which allows for multi-modal distributions and non-linear processes.

In general the filtering objective is to estimate moments of the filtering distribution instead of estimating the distribution itself. In particle filtering the basic idea is to draw samples, also called *particles*, from the distribution and estimate the moments using these, i.e.

$$\int h(\boldsymbol{s}_t)p(\boldsymbol{s}_t|I_{1:t})\mathrm{d}\boldsymbol{s}_t \approx \frac{1}{N}\sum_{j=1}^{N} h(\boldsymbol{s}_t^{(j)}) \ , \tag{7}$$

where $\boldsymbol{s}_t^{(j)}$ are the samples, and $h(\cdot)$ is any function of interest. In practice, we cannot draw samples from the filtering distribution as it is unknown. We therefore draw samples from an instrumental distribution $q(\boldsymbol{s}_t|I_t, \boldsymbol{s}_{t-1})$. These samples are then weighted such that the weighted sum of the samples provides an unbiased estimate of the moments. It can be proved [2] that these weights can be recursively updated as

$$\omega_t^{(j)} \propto \omega_{t-1}^{(j)} \cdot \frac{p(I_t|\boldsymbol{s}_t^{(j)})p(\boldsymbol{s}_t^{(j)}|\boldsymbol{s}_{t-1}^{(j)})}{q(\boldsymbol{s}_t^{(j)}|I_t, \boldsymbol{s}_{t-1}^{(j)})} \quad . \tag{8}$$

Most often the instrumental distribution is chosen to be the predictive distribution $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$ as this simplifies the weight update. The resulting algorithm is known as the Bootstrap filter, which we are employing in this paper. This algorithm performs the following iterative steps for all particles

- Draw new samples $\boldsymbol{s}_t^{(j)}$ from $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}^{(j)})$;
- Compute normalised weights $\omega_t^{(j)} \propto \omega_{t-1}^{(j)} p(I_t|\boldsymbol{s}_t^{(j)})$.

From a practical point of view this approach is not numerically stable as all but one of the weights tends towards zero. To overcome this issue it is common to only keep samples with large weights, which is done by resampling the samples. In the Bootstrap filter a sample is kept in the next iteration with a probability equal to its weight. This can result in the same sample appearing several times after resampling.

To model the dynamics of the state process it is necessary to include either second order information or velocity. This can be done by either extending the state with a velocity vector or by changing the state distribution from a first to a second order Markov chain. The latter approach keeps the dimensionality of the state space at a minimum allowing for a computationally efficient filter, while the former approach allow us to estimate velocities. Since this is not needed in the current application we choose the latter approach. This corresponds to drawing new samples from $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}^{(j)}, \boldsymbol{s}_{t-2}^{(j)})$ instead of $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}^{(j)})$, while the weight update remains unchanged.

Although the particle filter in general is able to cope with multi-modal distributions and non-linear state changes it is not without flaws. Unless very good predictive models are available the filter generally needs $\mathcal{O}(N^D)$ samples to reliably estimate the moments [2], where D is the dimensionality of the state space.

3 Human Motion Analysis

As mentioned in the introduction, we wish to infer the full pose of the human in the scene from the image sequence. To simplify matters, we will restrict ourselves to working on the upper body, i.e. torso, head and hands. We will also assume known limb-sizes. This effectively reduces the full-pose space to being the space of joint angles. The most straight-forward choice of state space is then the space of angles, which will be discussed in Sec. 3.1. An alternative low dimensional state space will be discussed in Sec. 3.2.

3.1 Full-Pose Motion Analysis

We have seen in Sec. 2 that Bayesian motion analysis can be performed using a particle filter. The obvious way of realising such a filter is to perform the filtering in the space of all poses $\boldsymbol{\theta}$. This only requires that we provide a predictive distribution $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}, \boldsymbol{\theta}_{t-2})$ and a likelihood function $p(I_t|\boldsymbol{\theta}_t)$.

The likelihood function will be described in detail in Sec. 4, so here we focus on prediction. This can simply be performed by extrapolating the two previous states and adding noise. In more detail we define

$$p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}, \boldsymbol{\theta}_{t-2}) = \mathcal{N}\left(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1} + \Delta_{t-1}, \boldsymbol{\Sigma}\right) \quad , \tag{9}$$

where $\Delta_{t-1} = \boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_{t-2}$ represents the current displacement and $\boldsymbol{\Sigma}$ is the covariance matrix of the prediction noise. In absence of additional knowledge about the motion, we choose the least committed model and define $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}$, with σ being a parameter and $\boldsymbol{I}$ being the identity matrix.

This simple setup provides a full system for Bayesian motion analysis. The problem with this approach is the high dimensionality of the state space. In this paper we restrict ourselves to studying an upper body model that has 47 degrees of freedom. Since the necessary number of particles grows exponentially with the dimensionality of the state space, we cannot expect to reliably use a particle filter in this state space. We therefore seek a more low dimensional state space with a similar amount of expressive power.

3.2 End-Effector Motion Analysis

As an alternative to the full-pose state space, we propose to use inverse kinematics to reduce the dimensionality of the state space. This is essentially done by performing the tracking in end-effector space. Here we define the end-effectors as the head and the hands. The end-effector space is thus the three dimensional positions of the three end-effectors, i.e. $\mathbb{R}^9$. The strategy is then to use inverse kinematics to infer the full pose. Once the full pose has been infered it can be measured just like the system working in full-pose space (see Sec. 4). This allows us to compare results from both systems.

In more details we define $\boldsymbol{x}_{head}$, $\boldsymbol{x}_{hand_0}$ and $\boldsymbol{x}_{hand_1}$ as the three dimensional positions of the end-effectors. To simplify matters, we will assume that the hands are conditionally independent given the position of the head. That is,

$$p(\boldsymbol{x}_{head}, \boldsymbol{x}_{hand_0}, \boldsymbol{x}_{hand_1}) = p(\boldsymbol{x}_{head}) p(\boldsymbol{x}_{hand_0}|\boldsymbol{x}_{head}) p(\boldsymbol{x}_{hand_1}|\boldsymbol{x}_{head}) \quad . \tag{10}$$

This simply means we represent the hands relative to the position of the head. Thus we define the state as $\boldsymbol{s} = (\boldsymbol{x}_{head}, \boldsymbol{x}_{hand_0} - \boldsymbol{x}_{head}, \boldsymbol{x}_{hand_1} - \boldsymbol{x}_{head})$. This factorisation has the consequence that we can treat the end-effectors separately in the filtering.

To be able to make measurements we need to be able to compute $p(I_t|\boldsymbol{s}_t)$. As described in Sec. 2 we can compute the full pose $\boldsymbol{\theta}_t$ from the state $\boldsymbol{s}_t$. This allows

us to perform the measurement in full-pose space rather than simply measuring the position of the head and the hands. So we define $p(I_t|\boldsymbol{s}_t) \equiv p(I_t|\boldsymbol{\theta}_t)$.

The prediction can be performed much like the full-pose system (9). This boils down to linear extrapolation of the end-effectors followed by addition of Gaussian noise. In more detail

$$p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}, \boldsymbol{s}_{t-2}) = \mathcal{N}\left(\boldsymbol{s}_t|\boldsymbol{s}_{t-1} + \Delta_{t-1}, \sigma^2\boldsymbol{I}\right) \ , \tag{11}$$

where $\Delta_{t-1} = \boldsymbol{s}_{t-1} - \boldsymbol{s}_{t-2}$ represents the current displacement.

4 Visual Measurements

In this section we present a method for computing the likelihood of a full pose, i.e. $p(I_t|\boldsymbol{\theta}_t)$. To avoid notational clutter we will drop the t subscript in the following.

The basic idea is to assume that the individual limbs are independent, i.e.

$$p(\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\theta^{(n)}) \quad \text{and} \quad p(\boldsymbol{\theta}|I) = \prod_{n=1}^{N} p(\theta^{(n)}|I) \ , \tag{12}$$

where $\theta^{(n)}$ are the parameters of the n^{th} limb. From this assumption we see that

$$p(I|\boldsymbol{\theta}) = \frac{p(I)p(\boldsymbol{\theta}|I)}{p(\boldsymbol{\theta})} \tag{13}$$

$$= p(I)^{-(N-1)} \prod_{n=1}^{N} \frac{p(I)p(\theta^{(n)}|I)}{p(\theta^{(n)})} \tag{14}$$

$$\propto \prod_{n=1}^{N} p(I|\theta^{(n)}) \ . \tag{15}$$

That is, we only need to be able to evaluate the likelihood $p(I|\theta^{(n)})$ of individual limbs.

The basic assumption in the measurement model is that limbs can be treated independently. However, if one limb occludes another this assumption no longer holds. In this paper we do not attempt to model this situation as we are concerned with the choice of state space rather than the visual measurements.

4.1 Modeling the Likelihood of a Limb

We use a simple Markov random field (MRF) model to describe the appearance of a limb, in which the limb appearance statistics is described by histograms of a set of descriptive features capturing texture and colour information. The likelihood function $p(I|\theta^{(n)})$ thus takes the form of a Gibbs distribution with an energy functional consisting of terms for texture, colour and background. Each of these are considered independent and we thus define the likelihood as

$$\begin{aligned} -\log p(I|\theta^{(n)}) = \alpha_T d_T^2(H^{T_m}, H^T|\theta^{(n)}) + \alpha_C d_C^2(H^{C_m}, H^C|\theta^{(n)}) \\ + \alpha_B d_B^2(H^{B_m}, H^B|\theta^{(n)}) + \text{constant} \ , \end{aligned} \tag{16}$$

where $d_T^2(H^{T_m}, H^T|\theta^{(n)})$ is the distance between a texture model H^{T_m} and the observed texture H^T. $d_C^2(\cdot)$ and $d_B^2(\cdot)$ are the similar counterparts of the colour and background models. The α parameters control the relative importance of the individual terms.

The texture and colour models are based on the same principle, which boils down to computing a normalised histogram of a descriptive feature within the limb and comparing that with a normalised model histogram. When constructing these histograms we use a Gaussian aperture to weigh the individual pixel contributions. This aperture will be described in the next section. When comparing the histograms we are using the earth mover's distance [18], which takes into account small perturbations and shifts of the histograms.

The background model is based on a simple thresholding of the absolute difference between a background image and the current image. This binary image B is then compared to a simple rendering R of the pose. The rendered pose is created by thresholding the Gaussian apertures used in the histogram creation. We thus define $d_B^2(H^{B_m}, H^B|\theta^{(n)})$ as the number of pixels where the two binary images do not agree.

The final objective of tracking is to provide an estimate of the current pose. When using Bayesian filtering, one such estimate is the maximum a posterior solution (MAP), which corresponds to finding the maximum of the filtering distribution. With our specific choice of MRF measurement model, the MAP solution will correspond to minimising the energy functional (16) with respect to the pose. Hence, the objective is to find the pose for which the earth mover's distance between the model histograms and the actual observation histograms is minimised.

4.2 Gaussian Limb Aperture

In the kinematic model each limb correspond to a line segment as was illustrated in Fig. 1. Hence, in order to create texture and colour histograms we need to define a spatial extend of each limb. This is done by forming a Gaussian aperture around each limb line segment. This aperture is used to weight individual pixel contributions in the histograms.

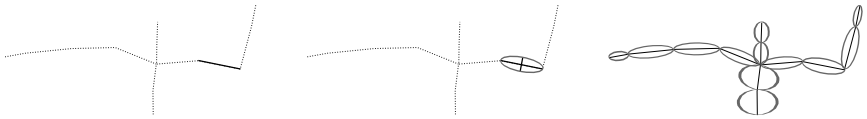
To compute the aperture, the line segment is projected onto the image plane and its mean point $\boldsymbol{\mu}_n$ is computed. From the orientation vector $\boldsymbol{v}_1$ along the projected line segment and its perpendicular counter-part $\boldsymbol{v}_2$ a covariance matrix

$$\boldsymbol{\Sigma}_n = \lambda_1 \boldsymbol{v}_1 \boldsymbol{v}_1^T + \lambda_2 \boldsymbol{v}_2 \boldsymbol{v}_2^T \tag{17}$$

is formed. The Gaussian aperture can then be defined as

$$L_n(\boldsymbol{x}|\theta^{(n)}) = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_n)^T \boldsymbol{\Sigma}_n^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_n)\right) \quad . \tag{18}$$

The first eigenvalue λ_1 is chosen such that $2.5\sqrt{\lambda_1} = d/2$, where d is the length of the projected line segment. The second eigenvalue is computed as $\lambda_2 = w_n \lambda_1$, where w_n controls the width of the n^{th} limb. The entire process is illustrated in Fig. 3.



(a) (b) (c)

Fig. 3. An illustration of the Gaussian apertures. (a) The kinematic skeleton with one high-lighted limb. (b) The direction of the limb is computed and a Gaussian aperture stretching in this direction is formed. (c) All Gaussian apertures of the same pose. Notice that the apertures of different limbs can have different widths.

4.3 Texture and Colour Features

In each pixel we compute both colour and texture features. Specifically, we compute colour saturations and gradient orientations. The colour saturation is computed as the S-channel of the HSV representation of the image. Each pixel entry in the histograms are weighted with the value of the Gaussian limb aperture in the pixel. In more detail, we collect the colour saturation histogram as

$$H_k^C(C\{I\}|\theta^{(n)}) = \sum_{\boldsymbol{x}=1}^{M} L_n(\boldsymbol{x}|\theta^{(n)})\mathbf{1}_{\Delta_k}[C\{I\}(\boldsymbol{x})], \quad k = 1, \ldots, K \ , \tag{19}$$

where k is the index of the histogram bin, $\mathbf{1}_{\Delta_k}$ is the indicator function of the bin interval Δ_k and $C\{I\}$ is the colour saturation of the image I.

The texture model is computed in much the same manner, except the gradient orientations are also weighted with the gradient lengths. Specifically, the histogram is computed as

$$H_k^T(\psi\{I\}|\theta^{(n)}) = \sum_{\boldsymbol{x}=1}^{M} \beta(\boldsymbol{x}) L_n(\boldsymbol{x}|\theta^{(n)})\mathbf{1}_{\Delta_k}[\psi\{I\}(\boldsymbol{x})], \quad k = 1, \ldots, K \ , \tag{20}$$

where β is the gradient length and ψ is its orientation. To make the features independent of the orientation of the limb, we compute the gradient in the principal coordinate system of the Gaussian aperture.

To control the relative importance of the features we compute the parameters α_T and α_C in an *ad-hoc* manner as

$$\alpha_T = \frac{\mathcal{H}(H^{C_m})}{\mathcal{H}(H^{T_m}) + \mathcal{H}(H^{C_m})} \quad \text{and} \quad \alpha_C = \frac{\mathcal{H}(H^{T_m})}{\mathcal{H}(H^{T_m}) + \mathcal{H}(H^{C_m})} \ , \tag{21}$$

where $\mathcal{H}(\cdot)$ is the entropy of a histogram. The intuition behind this choice is that more peaked model histograms should have a greater influence on the likelihood. The importance α_B of the background model has been selected manually.

The training of the model requires collecting texture and colour histograms for each limb. This is done by letting the human in front of the camera take a known pose, which allows us to collect the histograms from the first frame. The known starting pose also provides us with an initial state for the motion analysis.

5 Discussion

In the previous sections two different motion analysis systems have been described: one working on full-pose space and one working in end-effector space. Both use the same measurement system and their prediction systems are as similar as possible. This enables a comparison of the two trackers. It should be noted that the visual measurements are by far the computationally most expensive part of the tracking. Since each particle requires one visual measurement, the number of particles is proportional to the final computational time.

Fig. 4 shows selected frames from an image sequence with tracking results superimposed.[1] The result is computed as the mean of all particles, as this seems to stabilise the estimate when only few particles are used. First, we tracked the motion in full-pose space using 100 particles. Here the system quickly looses track of one arm and produces a large amount of "jitter" in the motion estimation. The computations took approximately five minutes on standard PC hardware. We then increased the number of particles to 5000, which resulted in a successful tracking with only little jitter. Unfortunately, this required more than 10 hours of computation time. As a final experiment we ran the tracking in end-effector space using 25 particles. The result of this experiment is a successful tracking that is comparable in quality with the previous experiment, but with somewhat more jitter. This jitter is a direct consequence of the small amount of particles. If this is increased the jittering decreases. The computations took approximately five minutes.

The results show that tracking in end-effector space is possible and that large speed-ups can be achieved using this approach. When creating a tracker for use in a physiotherapeutic rehabilitation programme it is essential to have real-time performance in order to provide feedback to the patient. Tracking in end-effector space makes this requirement more plausible compared to tracking in full-pose space.

Tracking in end-effector space does provide speed-ups while it still allows for a large set of motions. The resulting tracker is thus more versatile than low dimensional trackers that are tuned towards very specific motion types, such as walking, golf or tennis playing and so forth. However, the tracking will most often be less precise when it is performed in end-effector space. When infering the full pose from the end-effectors, the inverse kinematics solver finds one out of several minima. Hence, a particle can get a low weight even if it is in the correct part of end-effector space due to limitations of the inverse kinematics solver. This problem will result in a loss of accuracy.

The choice of state space basically boils down to a choice between accuracy, versatility and speed. Tracking in full-pose space allows for a high accuracy and versatility, but sacrifices speed. By learning manifolds in full-pose space it is possible to track on these. This allows for high accuracy and speed, but sacrifices versatility as the manifolds can only describe single types of motion. Tracking in

[1] The sequences are available on-line at `http://humim.org/emmcvpr2009/`.

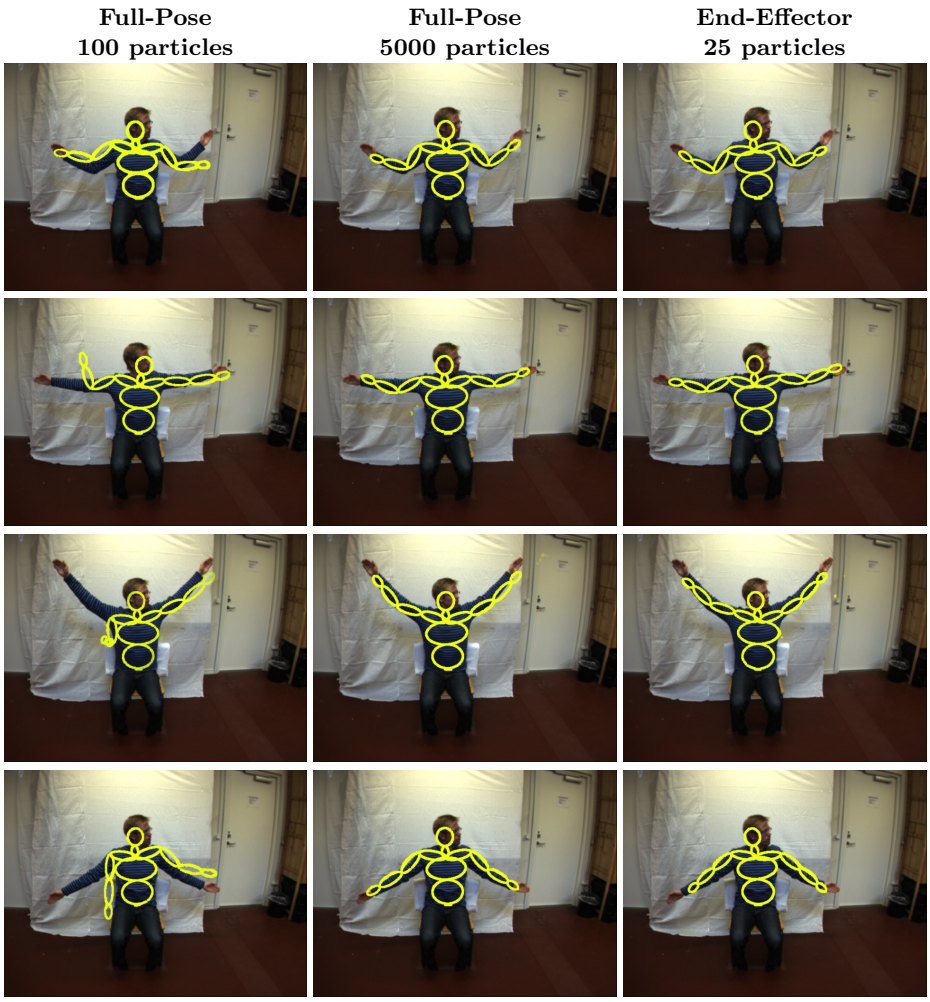


Fig. 4. Images from tracking sessions. The first column corresponds to tracking using 100 particles in full-pose space and the second column corresponds to 5000 particles in the same space. The third column corresponds to 25 particles in end-effector space. The rows correspond to the 32^{nd}, 94^{th}, 126^{th} and the 196^{th} frame of the sequence.

end-effector space allows for great versatility and speed, but comes with a loss of accuracy.

6 Conclusion and Future Work

In this paper we presented a low dimensional state space — the end-effector space — suitable for fast and versatile human motion estimation. Experiments with tracking in this space shows that good results can be achieved using only few particles. Due to the use of an inverse kinematics solver the approach can be expected to have less accuracy than when working in full-pose space, but makes real-time tracking of humans feasible.

In the immediate future we plan on improving the measurement model such that it can handle self-occlusions. The work of Sidenbladh and Black [19] and Roth et. al. [20] seems like good sources of inspiration. Also, a more detailed experimental validation will be performed. Specifically, we will validate our method on the ground truth data set of Knossow et.al. [21] and compare with their method, which should allow us to quantitatively evaluate the accuracy of our approach.

References

1. Poppe, R.: Vision-based human motion analysis: An overview. Computer Vision Image Understing **108**(1-2) (2007) 4–18
2. Cappé, O., Godsill, S.J., Moulines, E.: An overview of existing methods and recent advances in sequential Monte Carlo. Proceedings of the IEEE **95**(5) (2007) 899–924
3. Engell-Nørregård, M., Erleben, K.: A Projected Non-linear Conjugate Gradient Method for Interactive Inverse Kinematics. In: MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling. (2009)
4. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Comput. Vis. Image Underst. **104**(2) (November 2006) 90–126
5. Wang, J.M., Fleet, D.J., Hertzmann, A.: Gaussian Process Dynamical Models for Human Motion. Pattern Analysis and Machine Intelligence, IEEE Transactions on **30**(2) (2008) 283–298
6. Urtasun, R., Fleet, D.J., Fua, P.: 3D People Tracking with Gaussian Process Dynamical Models. In: CVPR '06: Proceedings of the 2006 IEEE Computer Society , Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, IEEE Computer Society (2006) 238–245
7. Lu, Z., Carreira-Perpinan, M., Sminchisescu, C.: People Tracking with the Laplacian Eigenmaps Latent Variable Model. In Platt, J., Koller, D., Singer, Y., Roweis, S., eds.: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA (2008) 1705–1712
8. Sidenbladh, H., Black, M.J., Fleet, D.J.: Stochastic tracking of 3d human figures using 2d image motion. In: Proceedings of ECCV'00. Volume II of LNCS 1843., Springer (2000) 702–718
9. Elgammal, A.M., Lee, C.S.: Tracking People on a Torus. IEEE Transaction on Pattern Analysis and Machine Intelligence **31**(3) (March 2009) 520–538

10. Urtasun, R., Fleet, D.J., Hertzmann, A., Fua, P.: Priors for people tracking from small training sets. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. Volume 1. (2005) 403–410
11. Loy, G., Eriksson, M., Sullivan, J., Carlsson, S.: Monocular 3D Reconstrunction of Human Motion in Long Action Sequences. In: Proceedings of ECCV'04. LNCS 3024, Springer (2004) 442–455
12. Sminchisescu, C., Triggs, B.: Kinematic Jump Processes for Monocular 3D Human Tracking. In: In IEEE International Conference on Computer Vision and Pattern Recognition. (2003) 69–76
13. Engell-Nørregård, M., Erleben, K.: Estimation of Joint types and Joint Limits from Motion capture data. In: WSCG 2009: 17-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision. (2009)
14. Nocedal, J., Wright, S.J.: Numerical optimization. Springer Series in Operations Research. Springer-Verlag, New York (1999)
15. Zhao, J., Badler, N.I.: Inverse kinematics positioning using nonlinear programming for highly articulated figures. ACM Trans. Graph. **13**(4) (1994) 313–336
16. Baum, L.E.: An inequality and associated maximization technique in statistical estimation of probabilistic functions of markov processes. Inequalities **3** (1972) 1–8
17. Kalman, R.: A new approach to linear filtering and prediction problems. Transactions of the ASME-Journal of Basic Engineering **82**(D) (1960) 35–45
18. Rubner, Y., Tomasi, C., Guibas, L.: A metric for distributions with applications to image databases. In: Sixth International Conference on Computer Vision, 1998. (1998) 59–66
19. Sidenbladh, H., Black, M.J.: Learning image statistics for bayesian tracking. In: Proc. of International Conference on Computer Vision. Volume II., Vancouver, Canada (2001) 709–716
20. Roth, S., Sigal, L., Black, M.J.: Gibbs likelihoods for bayesian tracking. In: CVPR 2004. (2004)
21. Knossow, D., Ronfard, R., Horaud, R.P.: Human motion tracking with a kinematic parameterization of extremal contours. International Journal of Computer Vision **79**(2) (September 2008) 247–269