Hierarchical Particle Filtering for 3D Hand Tracking

Alexandros Makris, Nikolaos Kyriazis, Antonis A. Argyros
Institute of Computer Science, FORTH, Greece
{amakris,kyriazis,argyros}@ics.forth.gr

Abstract

We present a fast and accurate 3D hand tracking method which relies on RGB-D data. The method follows a model based approach using a hierarchical particle filter variant to track the model's state. The filter estimates the probability density function of the state's posterior. As such, it has increased robustness to observation noise and compares favourably to existing methods that can be trapped in local minima resulting in track loses. The data likelihood term is calculated by measuring the discrepancy between the rendered 3D model and the observations. Extensive experiments with real and simulated data show that hand tracking is achieved at a frame rate of $90 fps$ *with less that* $10mm$ *average error using a GPU implementation, thus comparing favourably to the state of the art in terms of both speed and tracking accuracy.*

1. Introduction

Tracking articulated objects in 3D is a challenging task with many applications in various fields such as health care, telepresence, surveillance, and entertainment. The problem of 3D hand tracking poses additional challenges mainly due to its high dimensionality and the frequent and often severe self occlusions. Lately, this problem has received a lot of attention and several works that took advantage of RGB-D sensors advanced the state of the art in terms of tracking accuracy and processing time. Both these factors are very important since the nature of most hand tracking applications (e.g., human-computer interaction, tele-operation, sign language understanding) require high precision that needs to be achieved at interactive frame rates.

In this work we tackle the hand tracking problem with a model based approach. The model is tracked using the Hierarchical Model Fusion framework (HMF), first proposed by Makris et al. [17], which is a particle filter (PF) variant that decomposes the initial problem into smaller and simpler problems and efficiently addresses the implications of the high dimensionality. The filter relies on a likelihood model that measures the discrepancy between a rendered 3D hand model and the observations provided by an RGB-D camera, similarly in spirit to the state of the art approach of Oikonomidis et al [19, 20]. Extensive experiments demonstrate that the proposed method results in improved robustness (i.e. fewer track losses) and smaller tracking errors that are combined with a real time performance.

Figure 1. The proposed approach accepts markerless visual input from an RGB-D camera to track the pose and full articulation of a human hand performing unconstrained 3D motion. By estimating the probability density function of the hand's state posterior it has increased robustness to observation noise and compares favourably to existing methods that can be trapped in local minima resulting in track loses.

1.1. Related work

In the following, we attempt to provide a review of the main approaches for 3D hand tracking by structuring the relevant works with respect to a number of design choices.

Bottom up vs top down approaches: There are mainly two families of methods that are used for hand tracking i.e. model based (top-down) approaches and appearance based (bottom-up) approaches. The appearance based approaches rely on image features and a classifier to find the corresponding pose within a set of predefined hand poses [4, 9, 13, 14, 18, 30, 31]. Model based approaches on the other hand [19, 20, 23, 24, 28] define a parametric model of the hand and search for the optimal solution in the model's continuous parameter space. The optimal solution comes either by local optimization around the previous estimate or by filtering. Bottom-up approaches require big training datasets to capture the large appearance varia-

978-1-4673-6759-2/15/$31.00 ©2015 IEEE

tion that arises due to the highly articulated structure of the hand. To cope with this issue typically a coarse pose quantization is performed. The advantage of these data-driven approaches is that no hand initialization is required. Top-down approaches on the other hand require initialization and do not recover easily from a track failure but provide a more accurate estimation of the hand's pose.

Hybrid approaches: Hybrid methods that combine the model based and appearance based approaches have also been proposed. Hybrid methods typically use some sort of appearance information to guide the model based optimization. In [25, 26], a generative tracker based on color similarity is combined with a part-based discriminative method. The generative tracker uses a gradient descent scheme to find the optimal pose. In each timestep, two initialization poses are used, the previous estimation and the proposal from the discriminative method. In [33], another hybrid method is presented. Random forests are used to form the initial pose proposals while a simple optimization scheme searches for the local optimum around these proposals. In [22], a fast and accurate hand-tracking method using depth data is presented. The method combines stochastic optimization with gradient descent to achieve fast convergence. Finger detection is also used to propose candidate poses for the optimization.

Holistic vs part based approaches: Model based methods can be further classified in holistic and part based depending on the type of model that they use. Part-based approaches define the state as a set of sub-states that each corresponds to a part of the articulated model. Then constraints in the relative position of the parts are typically enforced. In [28], a part-based approach that uses nonparametric belief propagation to approximate the posterior is presented. In [12], the authors propose a part-based method for tracking a hand that interacts with an object. Each part has 6-DOF (position and orientation) for a total of 96-DOF for the whole hand. For each part, the objective function is defined locally and uses depth information after a skin color based hand segmentation. An explicit data driven occlusion modeling is performed that ensures that areas that are significantly closer to the sensor compared to the hypothesis do not contribute to the objective function. Model based approaches that perform local optimization around the previous estimate have been recently proposed. In [5] an approach that captures the motion of two hands interacting with an object is proposed. An iterative optimization scheme that uses the Levenberg-Marquard algorithm is employed. In [19, 20], Particle Swarm Optimization (PSO) is applied for single and two-hands tracking while a custom evolutionary optimization method is proposed in [21] for these two problems. The main drawback of the optimization based methods is that since they perform local optimization are prone to be trapped in local minima of the cost function.

Bayesian approaches. From the discussion so far it becomes evident that a model-based approach that does not seek only for a single, optimal solution and thus, is less likely to get trapped into local minima, would constitute an attractive alternative for 3D hand tracking compared to local optimization methods. One option in this direction would be to adopt a Bayesian tracking framework that seeks to estimate the posterior distribution of the hand's state. One successful Bayesian approach, the Particle Filters (PF) is now the norm for many tracking applications [3]. However, in the case of tracking articulated objects with many degrees of freedom the standard particle filter fails due to the high dimensionality of the problem. Still, several approaches that follow the Bayesian paradigm have been proposed for articulated objects tracking.

A hierarchical grid-based Bayesian filter is proposed in [27]. The levels of hierarchy follow a coarse to fine space representation. The update algorithm follows the hierarchy so that regions with low likelihood are identified and discarded early in the process. Edge and color cues are used to define the data likelihood. In [32], a PF based tracker performs principal component analysis (PCA) to obtain a 7D subspace that characterizes the hand motion. In that subspace the valid hand configurations lie on a manifold and a dynamic model gnerates hypotheses on that manifold. The method is tested for cases with small global motion and rotation. The works in [7, 8], propose a combination of a PF with Stochastic Meta-Descent (SMD), a gradient based local optimization method. This achieves 26-DOF hand tracking using a depth sensor. The SMD method converges fast to a local minimum which however is not guaranteed to be the global minimum. Therefore, PF is used on top of the SMD to increase the chances of finding it. The paper reports successful hand tracking but the method is slow (30 seconds per frame).

Partitioned Sampling (PS) [15, 16] decomposes the initial state into partitions and uses visual cues that can independently localize the sub-state of each partition. In [16], it has been applied in tracking a 7-DOF hand (position, scale, and 2D orientation of the palm and 3 joint angles). In [34], the partitioned sampling is used to track the contour of a hand in a 14D space (b-spline deformable template). Skin color is used to calculate the observation likelihood.

The annealed particle filter (APF) [10] is another approach to tackle high-dimensional problems by using a series of smoothed likelihood functions and a broad-to-fine searching approach. To do so, it drops the Bayesian framework upon which the traditional PF is based and it only tries to locate the global maximum instead of approximating the posterior probability density function (PDF). The APF based method in [24] estimates the diffusion noise added to the particles between each layer in order to distinguish between the accuracy of the hypotheses distribution and the

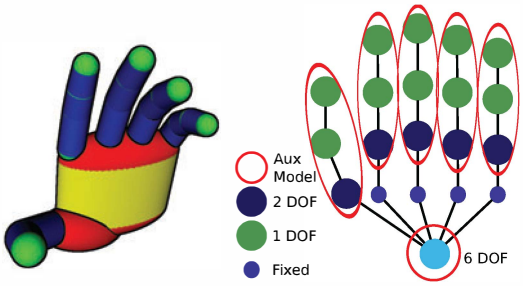
discriminatory capacity of the likelihood function. In [6], the presented articulation tracking method fuses the APF with the PS in order to use the advantages from both strategies. APF is better at finding the local maxima but the number of particles required is still large while PS is more efficient but gets trapped in local maxima. That approach partitions the state and performs annealing steps within these partitions. The method has been applied to full body tracking with satisfactory results but, to the best of our knowledge, it hasn't yet been tested for hand tracking. In [11], another variant of PS that is designed for articulated models is proposed. The method identifies conditionally independent sub-parts of the state that can be updated simultaneously. With a modified resampling strategy better particles are created by swapping independent sub-parts between particles. This method has yet to be tested on hand tracking.

1.2. Our contribution

In this work we adapt the Bayesian Hierarchical Model Framework (HMF) first proposed in [17] to the hand tracking problem. Using that framework which is a PF variant we build a tracker that uses six *auxiliary models* that lie in lower dimensional spaces as proposals for the 26-DOF *main model* of the hand. The auxiliary models track the position and orientation of the palm (6-DOF), and the joint angles of the five fingers (4-DOF each). HMF estimates and propagates the PDF of the posterior (unlike optimization techniques that only look for the local optimum) thus it is more robust to observation noise and it is more likely to recover from a short track loss. A fast, robust, distance metric that measures the discrepancy between the rendered model and the RGB-D observations is proposed. The method is tested qualitatively using real challenging sequences and quantitatively using simulated data. As a baseline for the comparisons we use the method at [19]. The experiments show clear benefit of the proposed approach in terms of speed, accuracy, and robustness. We achieve tracking rates of $90 fps$ with average error less than $10mm$ using a GPU implementation. In summary, the main contributions of the paper are:

- The application of the Bayesian HMF framework to the hand tracking problem. The followed methodology is generic and can be applied in any articulated model. As a PF variant the method estimates the posterior PDF rather than the local optimum which increases the robustness to occlusions and can handle multimodal likelihoods.

- The achievement of a significantly faster performance than the state of the art model based methods for the same tracking accuracy.



(a) (b)

Figure 2. The employed 3D hand model: (a) hand geometry, (b) hand kinematics.

2. Hierarchical Model Framework (HMF)

2.1. Hand model

The hand model consists of a palm and five fingers. Two basic geometric primitives are used to build the hand model, a sphere and a cylinder. The model is depicted in Fig. 2. The kinematics of each finger are modeled using four parameters, two for the base angles and two for the remaining joints. Bounds on the values of these parameters are set based on anatomical studies [1]. The global position of the hand is represented by a fixed point on the palm and the global orientation by a quaternion representation. This results to a 27 parameter representation that encodes the 26-DOF.

2.2. HMF tracking algorithm

In the following we describe the HMF tracking framework [17] and its adaptation to the hand tracking problem. The HMF uses several auxiliary models that are able to provide information for the state of the main model which is to be estimated. In the hand tracking problem the main model is a full 26-DOF model of the hand configuration. Each of the auxiliary models tracks a distinct part of the hand; we use one for the palm with 6-DOF for its 3D position and orientation and one for each finger with 4-DOF for the joint angles (Fig. 2b). This selection of auxiliary models encodes certain design choises that are crucial in order to build an efficient tracker. The first is the ability to create meaningful likelihoods for each auxiliary model which is satisfied by the current model selection since the state of each one directly affects the position of an observable part of the hand. The second concerns the expected computational gain from the dimensionality reduction. The auxiliary models that we selected have at most 6-DOF which is much less than the original 26-DOF. Of course, the main model still has 26-DOF but since it exploits the information from the auxiliary models the search in its high dimensional space is significantly narrowed. We define the full state $\mathbf{x}_t$ at a time step t as the concatenation of the sub-states that

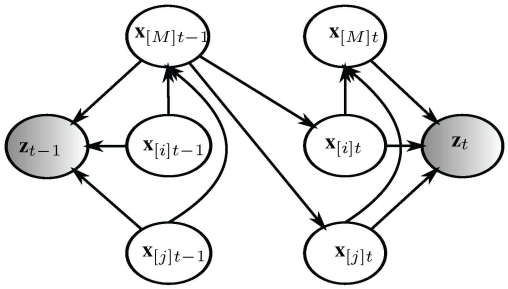


Figure 3. Dynamic Bayesian network of the proposed model depicting two auxiliary models j , i and the main model for the slices $t-1$ and t of the temporal dimension.

correspond to the M auxiliary models and the main model $\mathbf{x}_{[0:M]t}$. By $\mathbf{z}_t$ we denote the measurements at time step, t. The state decomposition and the independence assumptions that we make are graphically represented using the dynamic Bayesian model of Fig. 3. The observations at each time step are independent from previous states and observations given the current state. The state of the main model depends on the states of the auxiliary models at a given time step since they encode the pose of its parts. The state evolution is modeled by linking the main model at $t-1$ (which contains the most refined pose estimation for that time step) with the auxiliary models at t.

The framework follows the Bayesian approach for tracking [3]. By $\mathbf{x}_{0:t}$ we denote the state sequence $\{\mathbf{x}_0 ... \mathbf{x}_t\}$ and accordingly by $\mathbf{z}_{1:t}$ the set of all measurements $\{\mathbf{z}_1 ... \mathbf{z}_t\}$ from time step 1 to t. The tracking consists of calculating the posterior $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ at every step, given the measurements up to that step and a prior, $p(\mathbf{x}_0)$. Using the state decomposition the solution is expressed as:

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) \propto p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \prod_i p(\mathbf{z}_t|\mathbf{x}_{[i]t}) p(\mathbf{x}_{[i]t}|Pa(\mathbf{x}_{[i]t})), \quad (1)$$

where by $Pa(\mathbf{x}_{[i]t})$ we denote the parent nodes of $\mathbf{x}_{[i]t}$ (see Fig. 3). In (1) we make the approximation that the observation likelihood is proportional to the product of individual model likelihoods:

$$p(\mathbf{z}_t|\mathbf{x}_t) \propto \prod_i p(\mathbf{z}_t|\mathbf{x}_{[i]t}). \quad (2)$$

To efficiently approximate the posterior given the above state decomposition we use a particle filter that sequentially updates the sub-states. The algorithm approximates this posterior by propagating a set of hypotheses (particles) based on the importance sampling method [3]. The particles are drawn from the proposal distribution which is selected as the product of the dynamics of the auxiliary and main models:

$$q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \prod_i p(\mathbf{x}_{[i]t}|Pa(\mathbf{x}_{[i]t})). \quad (3)$$

Algorithm 1 The HMF algorithm for 3D hand tracking

Input: $\{\mathbf{x}^{(n)}_{[0:M]t-1}, \mathbf{w}^{(n)}_{t-1}\}^N_{n=1}$, $\mathbf{z}_t$.
for each model $i = 1$ to M **do**
  for each particle $n = 1$ to N **do**
    Sample $\mathbf{x}^{(n)}_{[i]t}$ from $p(\mathbf{x}_{[i]t}|Pa(\mathbf{x}_{[i]t})^{(n)})$.
    Update its weight $\mathbf{w}^{(n)}_t$ using $p(\mathbf{z}_t|\mathbf{x}^{(n)}_{[i]t})$.
  end for
  Normalize the particle weights.
  Resample the particle set according to its weights.
end for
Output: $\{\mathbf{x}^{(n)}_{[0:M]t}, \mathbf{w}^{(n)}_t\}^N_{n=1}$.

The particles are then weighted by the importance weights:

$$w_t = \frac{p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})}{q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})} = w_{t-1} \prod_i p(\mathbf{z}_t|\mathbf{x}_{[i]t}). \quad (4)$$

Using these factorizations for the proposal distribution and the weights, the algorithm can sequentially update the substates by sampling from the factor of the proposal that corresponds to the i-th sub-state and subsequently updating the weights with the i-th factor of the likelihood.

The steps to estimate the posterior at time t given the previous estimate are shown in Algorithm 1. The input of the algorithm is the set of N weighted particles from the previous time step $\{\mathbf{x}^{(n)}_{[0:M]t-1}, \mathbf{w}^{(n)}_{t-1}\}^N_{n=1}$ and the current observations. The normalization step of the algorithm modifies the weights so as to sum up to one. The resampling step randomly chooses particles according to their weights so that particles with low weights are discarded and particles with high weights are selected multiple times. The output of the algorithm is the current weighted particle set $\{\mathbf{x}^{(n)}_{[0:M]t}, \mathbf{w}^{(n)}_t\}^N_{n=1}$. This way, we replace the search in a high dimensional space with a series of easier problems of lower dimensionality. Additionally, the main model uses a proposal distribution that depends on the previously updated auxiliary models. Therefore, we exploit the current observations to form the proposal and this guides the particles in high likelihood areas. Finally, the track estimate of the algorithm at each step is the weighted average of the main model particles.

2.3. Model dynamics

The state evolution for the main model exploits the state of the updated auxiliary models:

$$p(\mathbf{x}_{[M]t}|Pa(\mathbf{x}_{[M]t})) \equiv p(\mathbf{x}_{[M]t}|\mathbf{x}_{[0:M-1]t}) = N(\mathbf{x}_{[M]t}; \mathbf{x}_{[0:M-1]t}, \Sigma_M), \quad (5)$$

where by $N(y; m, \Sigma)$ we denote the normal distribution over y with mean m and covariance matrix Σ. The above

distribution encodes the fact that the main model is expected to be around the estimated position of its parts.

For the auxiliary models we define the state evolution using the main model at the previous time step:

$$\begin{aligned} p(\mathbf{x}_{[i]t}|Pa(\mathbf{x}_{[i]t})) &\equiv p(\mathbf{x}_{[i]t}|\mathbf{x}_{[M]t-1}) \\ &= N(\mathbf{x}_{[i]t}; a(\mathbf{x}_{[M]t-1}, i), \Sigma_i), \end{aligned} \tag{6}$$

where the operator $a(\mathbf{x}_{[M]t-1}, i)$ gives the part of the state of the main model $\mathbf{x}_{[M]t-1}$ that corresponds to the i-th auxiliary model.

2.4. Observation likelihood

The observation likelihood measures the degree of matching between a model pose and the observations. It is based on the approach described in [20]. The input of the method is an RGB-D image and a model pose. A preprocessing step uses the estimated hand position in the previous frame as reference and keeps only the observations that are within a predefined range around it. Within that window, skin color is detected using the method described in [2]. The observation consists of the resulting 2D depth and skin color maps $\mathbf{z} = \{z_d, z_s\}$. To calculate the likelihood for a given hypothesis of an auxiliary or of the main model we perform rendering given the camera calibration. In the following we drop for clarity the state subscripts that define the time and the model number. We thus refer to the state of a hypothesis by $\mathbf{x}$. This state contains the 27 parameters required to perform the rendering. The state of an auxiliary model i has less than 27 parameters and thus to render it we complete it by using the states of the already updated auxiliary models (1 to $i-1$) and the state of the main model at $t-1$. The result of rendering is a 2D depth map and the corresponding skin color map $\{r_d(\mathbf{x}), r_s(\mathbf{x})\}$. Let P_i be the set of pixels that are labelled as skin in both the observation and the hand model defined as $P_i = \{z_s \wedge r_s\}$ and P_u be the set of pixels that are labeled as skin in either the hand model or the observation $P_u = \{z_s \vee r_s\}$. We denote as λ the ratio of the number of elements of these two sets: $\lambda = \mid P_i \mid / \mid P_u \mid$. The following function $D(\mathbf{z}, \mathbf{x})$ is then used to evaluate the discrepancy between a hypothesis $\mathbf{x}$ and the observation $\mathbf{z}$:

$$D(\mathbf{z}, \mathbf{x}) = \lambda \frac{\sum_{p \in P_i} min(|z_{d,p} - r_{d,p}|, d_M)}{d_M \mid P_i \mid} + (1 - \lambda)\,. \tag{7}$$

This ranges from 0 for a perfect match to 1 for a mismatch. The intuition for this definition is that we weigh by the clamped depth difference the part of the pixels that overlap in the model and observation (P_i) whereas the rest of the pixels influence negatively the total distance. The clamping threshold d_M is required so that a few pixels with big depth differences should not influence an otherwise reasonable fit. Pixels that have depth difference above d_M are considered mis-matched. The definition for the distance guarantees that these mismatched pixels will penalize $D(\mathbf{z}, \mathbf{x})$ with the maximum value thus in exactly the same way as the pixels that are not in P_i. This is justified because in both these cases the corresponding 3D observation and hand model points are considered to be far from each other. Using $D(\mathbf{z}, \mathbf{x})$ the likelihood function is then given by:

$$p(\mathbf{z}|\mathbf{x}) = \exp\left\{-\frac{D^2(\mathbf{z}, \mathbf{x})}{2\sigma_l^2}\right\}. \tag{8}$$

3. Experiments

We performed extensive experiments to assess the performance of the proposed approach and to compare it with the approach of [19] as well as to PF variants. We used real data obtained by RGB-D sensors to qualitatively evaluate the methods. For quantitative evaluations we used synthetic data since real world annotated data are difficult to obtain. It should be noted that certain works (e.g., [22, 29]) have released datasets with ground truth. However, different choices in the modeling of the hand affect the measured ground truth. Furthermore, both of these works use bottom up information which means that temporal continuity is not required. Our approach requires temporal continuity and ensures that this will always be the case by operating on high framerates. Therefore those datasets are not relevant to our work.

The focus of the experiments is on high accuracy (average error lower than $10mm$) combined with real-time operation (framerates equal or higher than 30 fps). Both these factors are crucial for most of the applications that rely on 3D hand tracking. New RGB-D sensors deliver high framerates (e.g., 60 fps for Asus Xtion) so methods that can take advantage of these rates are required. The methods that have been included in our comparative evaluation are:

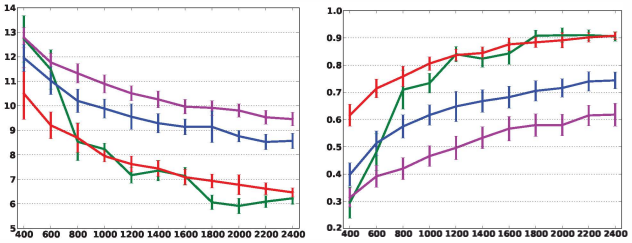
PSO The method of [19] using the PSO optimizer.

PFS The Sampling Importance Resampling algorithm [3].

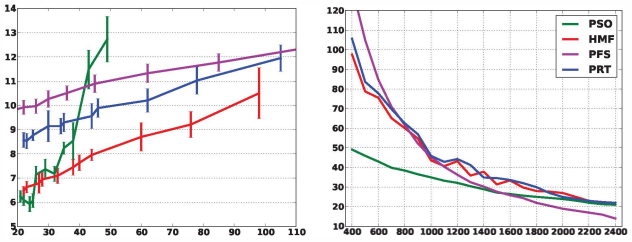
PRT The partitioned samling filter described in [16] with 6 partitions that correspond to the parameters of the palm and each of the five fingers.

HMF The proposed approach.

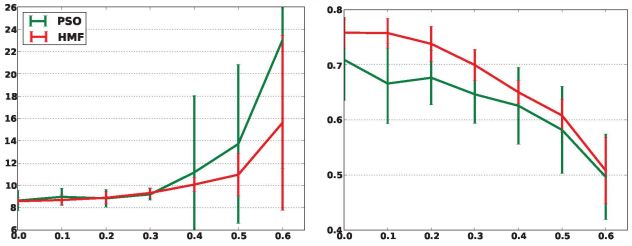
For all the methods we used the same distance metric of (7) to link the model with the observations. It has been confirmed experimentally that the performance of the methods using that metric is similar to the one achieved by using the metric of [20]. However, for the latter metric this performance is achieved only with the right value of the parameter



(a) $E(mm)$ - Budget (b) C - Budget



(c) $E(mm)$ - Processing rate(fps) (d) Processing rate(fps) - Budget



(e) $E(mm)$ - Noise Ratio (f) C - Noise Ratio

Figure 4. Quantitative results obtained in off-line experiments.

that specifies the weights of its two terms i.e. the depth difference and the silhouette matching whereas the proposed metric has no parameters. The methods were tested on a computer equipped with an Intel i7-4770 CPU (quad-core 3.4GHz), and an Nvidia GTX 580 GPU.

The synthetic dataset that we used for the evaluation consists of 700 frames of free hand movement. The initialization of the methods is performed using the grountruth position of the first frame. The comparison metric E measures the average distance between corresponding phalanx endpoints over a sequence similarly to [12]. For each experiment and for each method we perform thirty runs, so we measure and report the mean error as well as its standard deviation in all runs. A high error standard deviation is an indication that the method is not robust, and is typically caused when it completely looses the target in some of the runs. As an additional metric we also use the success rate C which is the ratio of the frames of the sequence for which the average error is below a threshold. Since we want to focus on high precision tracking we use as threshold the value of $10mm$ which is approximately the diameter of a finger.

The most computationally demanding part of the methods is the likelihood evaluation (see (8)). Therefore a reasonable criterion for quantifying the computational cost is the total number of required such evaluations. In the following we refer to this number as the computational budget. In practice however, the speed of the methods for a given budget varies significantly due to the parallel GPU implementation. In contemporary GPU architectures it is common that computational throughput increases with the number of parallel independent tasks. Therefore, for any fixed amount of computations it is highly preferable to execute a few and large batches rather than more and smaller ones. For the **PSO** method the algorithm is such that the minimum number of batches is equal to the number of generations (28 in our experiments). In contrast, all the PF variants require less batches, 1, 6, and 7 for the **PFS**, **PRT**, and **HMF**, respectively. This induces significant speed differences, in favor of the PF variants, which only saturates as the amount of total computations becomes excessively high, in which case computations are serialized.

We test the methods both off-line and on-line. During the off-line experiments we provide each method with all the frames of a sequence independently of its actual *processing framerate* which may be much slower. In on-line experiments the methods have to drop frames if the processing framerate is lower than the *image acquisition* framerate. Since we want to present results that are independent of our current implementation/hardware we firstly compare the methods with respect to the computational budget. Additionally we present the speed in terms of processed frames per second of the methods with respect to each budget.

3.1. Off-line experiments

The first series of quantitative experiments compare the accuracy of the methods for different computational budgets. The results are shown in Figs. 4a,4b. For each budget we perform 30 runs for each method and we plot the average error and success rate as well as the error bars that correspond to one standard deviation. It can be verified that the proposed **HMF** approach results in significantly smaller errors especially for lower budgets. For budgets larger than 800, **HMF** and **PSO** converge to around the same error value ($6mm$) which is significantly lower than the error of the other PF variants ($9mm$). From another viewpoint, the accuracy achieved by **PFS** and **PRT** for a computational budget of 2400, is achieved by **HMF** with almost one fourth of that budget. This means that the **PRT** and **PFS** methods do not achieve an acceptable error rate and are heavily outperformed by the **HMF** and **PSO** methods even with idealized simulated data. For this reason, for the rest of the experimental evaluation we focus our attention to the **HMF** and **PSO** methods.

Fig. 4d shows the processing framerates achieved by the methods as a function of their computational budget. The

comparison is much more favourable for the **HMF** method if we consider these processing framerates. For example, for a budget of 800, **HMF** process at $60fps$, while **PSO** less than $40fps$ and they have the same average error. Additionally, from Fig. 4c we note that while for a processing framerate of $30fps$ **PSO** and **HMF** have about the same error, for higher processing framerates the error of **PSO** is significantly higher. As an example, to achieve an acceptable average error of less than $10mm$ **PSO** cannot process more than $40fps$ while for the same accuracy, **HMF** manages to process at $90fps$.

The next experiment that we performed was designed to assess the methods in the presence of noisy observations. We wanted to simulate the imperfect data that come from a real depth sensor, imperfect skin color detection, and possible occlusions from unknown objects. To do so, we randomly generate disk shaped regions. Inside each such region we either: (i) completely remove the depth and skin observation; (ii) label the area as skin and give it a depth value that is modeled by a Gaussian around the real average depth. Fig. 5 shows some example frames from a sequence with and without noise. This type of noise has the double effect of removing useful data while introducing observations with depth values that are close to the actual depth of the target. Thus, such artifacts cannot be discarded by some type of pre-processing. We vary the ratio of the image area that is replaced by noise and we plot the results in Figs. 4e,4f. For this experiment we use the methods with budget 800 since for that value both **HMF** and **PSO** fulfil the requirements of high accuracy ($E < 10mm$) and real-time speed (*processing* $fps > 30$). As we observe from the figure both methods perform well when the noise ratio is below 0.3. For larger amounts of noise contaminations the accuracy of the methods is affected with the **HMF** being more robust and having an acceptable accuracy up to a noise ratio of 0.5.

For **HMF** we observe that apart from the lower average error E, the error variance among different runs is lower, especially for high noise ratios. This is attributed to the algorithm's ability to model the PDF of the posterior which allows it to recover from temporary track losses. **PSO** tracker on the other hand cannot easily recover from such track losses and this is reflected in the plots as larger error variances.

3.2. On-line experiments

Given our interest on real-time applications we also performed on-line simulation experiments. A method that operates in real time but cannot reach the sensor's frame rate will unavoidably drop frames. This is expected to have a negative effect on the performance of the method since for each dropped frame the search window around the previous position should increase to compensate for the sparser temporal sampling and the decreased temporal continuity. We perform the on-line experiments with a $60fps$ simulated sequence. We also add noise to the sequence to be as close as possible to a real world scenario.

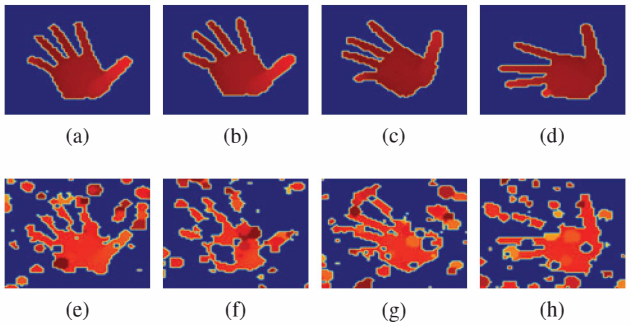
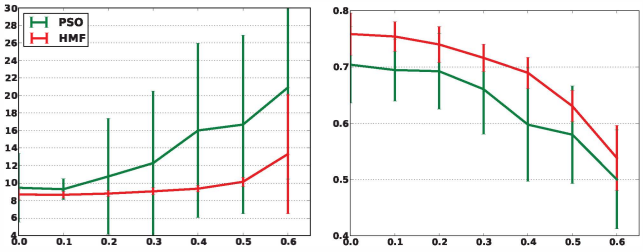


Figure 5. Depth maps for 4 frames of a sequence without noise (top row) and with noise ratio 0.4 (bottom row).



(a) $E(mm)$ - Noise Ratio. (b) C - Noise Ratio.

Figure 7. On-line experiments ($60fps$ sequence): Plot of error E and success rate C (10mm threshold) for different noise ratios. **HMF** is processing the input at $60fps$ while **PSO** at $30fps$.

Using the results of the off-line experiments we chose the suitable methods to track the sequences. More specifically, by combining the information of Fig.4d and Fig.4c we see that the **HMF** method with a budget of 800, can process the sequence at its original framerate ($60fps$) and results in an error that is below the limit of $10mm$. **PSO** cannot reach this framerate with an acceptable error. Therefore, we choose the best algorithm configuration with respect to error which is that of a budget of 1200 that permits a processing framerate of $30fps$ (i.e., dropping every second frame of the original sequence). Fig. 7 shows the results of the simulation. Both the average error as well as the error variance for different runs is much higher for **PSO** compared to the respective values for the **HMF** method. This experiment underlines the importance of having a fast method that can on-line process the data at high framerates.

In Fig. 6, we plot the error as it evolves throughout the sequence. We show 3 plots for low (0.1), mid (0.4), and high (0.6) noise ratios. As it can be verified, **PSO** loses the target for increasingly big intervals as the noise level increases.

The proposed **HMF** method has also been qualitatively

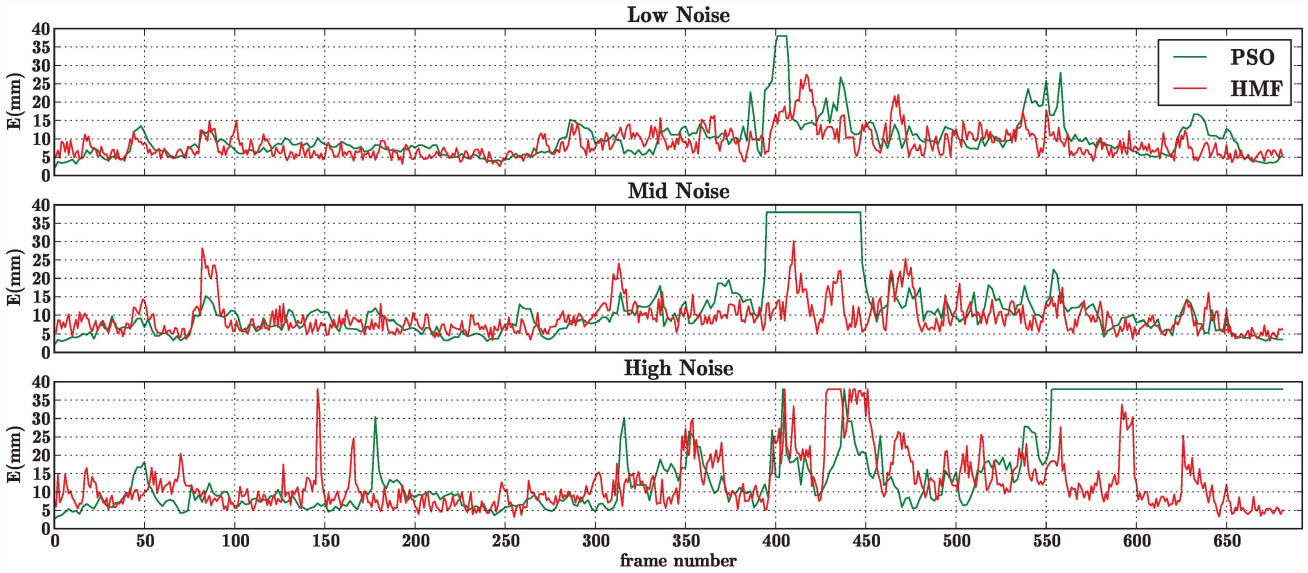


Figure 6. On-line Experiments: Plot of error E over the frames of the test sequence for different noise ratios (0.1, 0.4, and 0.6). **HMF** is procesing the input at $60fps$ while **PSO** at $30fps$. For each algorithm the run with the median average error was chosen as a representative one. For clarity of the figure we clip the error at $38mm$.

Figure 8. Sample tracked frames from a real world RGB-D sequence acquired at $60fps$. **Top row: HMF** manages to process the input at $60fps$. **Bottom row: PSO** processes the input at $30fps$ and therefore has to drop half of the frames.

verified in several challenging real-world sequences. Fig. 8 shows comparative screenshots from on-line tracking on a $60fps$ sequence. The methods operate with the same budgets as in the previous experiment and therefore process the sequence at $60fps$ and $30fps$ respectively. The fast motion and hand rotation trap the slower **PSO** method to a sub-optimal solution. A complete such run appears in the supplementary video material accompanying this paper.

4. Conclusions

In this work we proposed a 3D hand tracking method that is based on the well-established and solid framework of particle filtering. The presented method uses the Hierarchical Model Fusion (HMF) approach which decomposes the original, high dimensional problem into a number of smaller, manageable ones. HMF is shown to cope effectively and efficiently with the high dimensionality of the 3D articulations tracking problem where other PF variants are shown to fail. The required likelihood function employs rendering of the 3D hand model thus the problem of self-occlusions is naturally addressed. Additionally, the performed experiments show a clear benefit of the proposed approach compared to the state of the art Particle Swarm Optimization method, both in terms of speed and robustness. Accurate hand tracking is achieved on challenging sequences at a rate of $90fps$. The method's increased robustness that translates to more stable operation is mainly due to the fact that the HMF can better cope with multi-modality since it maintains an estimate of the posterior PDF rather than a single maximum. The achieved result with respect to the accuracy/computational performance trade off is very important as it enables the deployment of the proposed method in real world applications that involve fast motions of human hands observed at cameras with high acquisition frame rate.

Acknowledgements

This work was partially supported by the EU FP7-ICT-2011-9-601165 project WEARHAP and by the EU IST-FP7-IP-288533 project RoboHow.Cog.

References

[1] I. Albrecht, J. Haber, and H.-P. Seidel. Construction and animation of anatomically based human hand models. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 98–109, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[2] A. A. Argyros and M. I. A. Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. In T. Pajdla and J. Matas, editors, *Computer Vision - ECCV 2004*, number 3023 in Lecture Notes in Computer Science, pages 368–379. Springer Berlin Heidelberg, Jan. 2004.

[3] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb. 2002.

[4] V. Athitsos and S. Sclaroff. Estimating 3d hand pose from a cluttered image. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings*, volume 2, pages II–432–9 vol.2, June 2003.

[5] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision ECCV 2012*, number 7577 in Lecture Notes in Computer Science, pages 640–653. Springer Berlin Heidelberg, Jan. 2012.

[6] J. Bandouch and M. Beetz. Tracking humans interacting with the environment using efficient hierarchical sampling and layered observation models. In *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2040–2047, Sept. 2009.

[7] M. Bray, E. Koller-Meier, and L. Van Gool. Smart particle filtering for 3d hand tracking. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings*, pages 675–680, May 2004.

[8] M. Bray, E. Koller-Meier, and L. Van Gool. Smart particle filtering for high-dimensional tracking. *Computer Vision and Image Understanding*, 106(1):116–129, Apr. 2007.

[9] T. de Campos and D. Murray. Regression-based hand pose estimation from multiple cameras. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 782–789, June 2006.

[10] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *IEEE Conference on Computer Vision and Pattern Recognition, 2000. Proceedings*, volume 2, pages 126–133 vol.2, 2000.

[11] C. Gonzales and S. Dubuisson. Combinatorial resampling particle filter: An effective and efficient method for articulated object tracking. *International Journal of Computer Vision*, pages 1–30, Sept. 2014.

[12] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool. Tracking a hand manipulating an object. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1475–1482, Sept. 2009.

[13] C. Keskin, F. Kirac, Y. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1228–1234, Nov. 2011.

[14] C. Keskin, F. Kra, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision ECCV 2012*, number 7577 in Lecture Notes in Computer Science, pages 852–863. Springer Berlin Heidelberg, Jan. 2012.

[15] I. Laptev and T. Lindeberg. Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features. In M. Kerckhove, editor, *Scale-Space and Morphology in Computer Vision*, number 2106 in Lecture Notes in Computer Science 2106, pages 63–74. Springer Berlin Heidelberg, Jan. 2001.

[16] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In D. Vernon, editor, *Computer Vision ECCV 2000*, number 1843 in Lecture Notes in Computer Science, pages 3–19. Springer Berlin Heidelberg, Jan. 2000.

[17] A. Makris, D. Kosmopoulos, S. Perantonis, and S. Theodoridis. A hierarchical feature fusion framework for adaptive visual tracking. *Image and Vision Computing*, 29(9):594–606, Aug. 2011.

[18] S. Malassiotis and M. G. Strintzis. Real-time hand posture recognition using range data. *Image and Vision Computing*, 26(7):1027–1037, July 2008.

[19] I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. pages 101.1–101.11. British Machine Vision Association, 2011.

[20] I. Oikonomidis, N. Kyriazis, and A. Argyros. Tracking the articulated motion of two strongly interacting hands. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1862–1869, June 2012.

[21] I. Oikonomidis, M. Lourakis, and A. Argyros. Evolutionary quasi-random search for hand articulations tracking. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3422–3429, June 2014.

[22] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1106–1113, June 2014.

[23] J. M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Proceedings of the Third European Conference-Volume II on Computer Vision - Volume II*, ECCV '94, pages 35–46, London, UK, UK, 1994. Springer-Verlag.

[24] G. Ros, J. del Rincon, and G. Mateos. Articulated particle filter for hand tracking. In *2012 21st International Conference on Pattern Recognition (ICPR)*, pages 3581–3585, Nov. 2012.

[25] S. Sridhar, A. Oulasvirta, and C. Theobalt. Interactive markerless articulated hand motion tracking using RGB and depth data. In *2013 IEEE International Conference on Computer Vision (ICCV)*, pages 2456–2463, Dec. 2013.

[26] S. Sridhar, H. Rhodin, H.-P. Seidel, A. Oulasvirta, and C. Theobalt. Real-time hand tracking using a sum of anisotropic gaussians model. 2014.

[27] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1372–1384, Sept. 2006.

[28] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Visual hand tracking using nonparametric belief propagation. In *Conference on Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04*, pages 189–189, June 2004.

[29] D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3786–3793, June 2014.

[30] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, 33(5):169:1–169:10, Sept. 2014.

[31] R. Wang, S. Paris, and J. Popovi. 6d hands: Markerless hand-tracking for computer aided design. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 549–558, New York, NY, USA, 2011. ACM.

[32] Y. Wu, J. Lin, and T. Huang. Capturing natural hand articulation. In *Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings*, volume 2, pages 426–432 vol.2, 2001.

[33] C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *2013 IEEE International Conference on Computer Vision (ICCV)*, pages 3456–3462, Dec. 2013.

[34] D. Zhou, Y. Wang, and X. Chen. Hand contour tracking using condensation and partitioned sampling. In Z. Pan, X. Zhang, A. E. Rhalibi, W. Woo, and Y. Li, editors, *Technologies for E-Learning and Digital Entertainment*, number 5093 in Lecture Notes in Computer Science, pages 343–352. Springer Berlin Heidelberg, Jan. 2008.