6-DOF Model Based Tracking via Object Coordinate Regression

Alexander Krull, Frank Michel, Eric Bracmann, Stefan Gumhold,
Stephan Ihrke, Carsten Rother

TU Dresden, Dresden, Germany

Abstract. This work investigates the problem of 6-Degrees-Of-Freedom (6-DOF) object tracking from RGB-D images, where the object is rigid and a 3D model of the object is known. As in many previous works, we utilize a Particle Filter (PF) framework. In order to have a fast tracker, the key aspect is to design a clever proposal distribution which works reliably even with a small number of particles. To achieve this we build on a recently developed state-of-the-art system for single image 6D pose estimation of known 3D objects, using the concept of so-called 3D object coordinates. The idea is to train a random forest that regresses the 3D object coordinates from the RGB-D image. Our key technical contribution is a two-way procedure to integrate the random forest predictions in the proposal distribution generation. This has many practical advantages, in particular better generalization ability with respect to occlusions, changes in lighting and fast-moving objects. We demonstrate experimentally that we exceed state-of-the-art on a given, public dataset. To raise the bar in terms of fast-moving objects and object occlusions, we also create a new dataset, which will be made publicly available.

1 Introduction

In this paper we address the problem of tracking the pose of a previously known rigid object from a RGB-D video stream in real-time. The object pose is typically expressed relative to the camera and has 6-DOF, three for orientation and three for position. A solution to this problem has great impact in many application fields, ranging from augmented reality, human-computer interaction, to robotics. However, given constraints on high precision, real-time, as well as robustness to real world situations, such as occlusions and changes in lighting conditions, this is still an open problem.

Building on the results of multiple decades of research on object tracking, very recently several researchers have re-investigated pose estimation [14, 21, 4] and tracking approaches [16, 8, 18] in order to exploit new RGB-D sensor technology and to ensure real-time support through GPU based implementations. For the tracking problem the PF framework [12] has become a preferred choice as it can model multi-modal probability densities that are essential for successful tracking of objects with occlusions.

As described in more detail in Sec. 3.2 the PF framework incrementally traces the posterior probability density of the 6D pose through a set of samples from

the 6D Euclidean Group SE(3)[1]. Key ingredients to the PF are a model for the relative motion of the object over time, as well as an observation model for the image formation process. In single image pose estimation approaches, like [4], solely the observation model is used. An important difficulty in the application of the PF framework to 6D pose tracking is the high dimensionality of the state space, as each particle represents a 6D pose estimate. This either demands for a huge number of particles prohibiting real-time systems or for an importance sampling approach (compare [10]) based on a proposal distribution that effectively approximates the posterior probability distribution. Any pose estimation approach can be used to implement a proposal distribution as shown by Stückler *et al.* in [24], where a multi-resolution surfel representation of the tracked object was utilized.

In this work we demonstrate the combination of the PF framework with the recently developed concept of object coordinate regression which has achieved state of the art results for one shot estimation of camera pose [22] or object pose [4]. The object coordinate regression framework is detailed in Sec. 3.1. It exploits random forests to automatically learn optimal features from RGB-D training images. Brachmann *et al.* [4] have shown that such learning based approaches can efficiently deal with changes in illumination and with partial occlusions. Given this distinction between a training and test phase, our system works as follows. Given a potentially large selection of 3D objects, here 4, and 20 in [4], as well as example images of background, we first train a random forest. At test time, we know which object we want to track and use the output of the forest only for this particular object. Note, a straightforward extension, not evaluated in this work, is to jointly do multiple object detection and tracking. In this work, we carefully extend the work [4] to the 6D pose tracking problem. Our main contributions are:

- A new model-based 6D pose tracking framework, based on the concept of predicting 3D object coordinates, which helps to generalize better to real-world settings of fast-moving objects, occlusions and changes in lighting.
- The technically new aspect is a two-way procedure for optimally using the output of the object coordinate regression framework to determine the proposal distribution of the tracker.
- A new, challenging 6D pose tracking data set that will be publicly available.
- A system that exceeds state of the art results on 6D model-based tracking.

2 Related Work

Almost two decades ago the PF has been introduced for 2D visual tracking by Isard and Blake [15]. Based on a statistical observation model and a motion model the PF approximates the posterior distribution of the object's position in a non parametric form, using a set of samples. Ten years later Pupilli *et al.* [20] adapted the framework to 6-DOF camera tracking using edge features. Shortly

[1] The group of rigid body transformations.

later Klein *et al.* [16] presented an implementation that utilized the GPU for the evaluation of its observation model, which usually is the bottleneck in PF applications. The GPU implementation enabled them to deal with hidden edges while allowing a speedup [16].

The number of necessary particles and therefore the runtime can be reduced by guiding particle sampling with a proposal distribution. Ideally the proposal distribution is very close to the posterior distribution. Furthermore, it needs to exploit both the observation model and the motion model in order to improve over the standard PF as well as over one shot pose estimation. Bray *et al.* [5] improved hand pose tracking with a proposal distribution, which was defined as the mixture of two distributions both represented as a particle ensemble. The first particle ensemble is constructed in the default manner by applying the motion model to the sampled posterior distribution of the previous frame. The second ensemble is constructed by moving the resulting particles further to local optima and using them as centers of a mixture of normal distributions. Teuliere *et al.*[26] used a similar approach to edge-based tracking of simple objects from luminance images. Corresponding approaches for 3D pose tracking from RGB-D videos can be found in [9, 7, 18].

A PF that has to operate on the 6D Euclidean group SE(3) brings some theoretical challenges. The definition of probability distributions and calculation of average rotations is not straight forward. A theoretical analysis of these issues can be found in [6] and [17]. While earlier methods relied on simple random walk motion models, Choi *et al.* [9] used autoregressive models that assume a more or less constant pose velocity and were thus able to deal with faster motion.

With the availability of RGB-D sensors, the question of the best image features for the observation model has sparked recent research. Stückler *et al.* [24] use RGB-D images to learn 3D surfel maps of objects and use them in a PF operating on RGB-D. Choi *et al.* [8] present in their recent work a highly optimized GPU implementation that uses a traditional mesh representation. Their observation model is based on comparing rendered images and observations using color as well as depth features. Learning of the best features in a random forest has been successfully applied to pose estimation of articulated objects [25], camera pose [22] and object poses [4]. In these approaches a random forest is trained to predict part or object probabilities and in the latter two approaches also coordinates in a reference coordinate system of the background scene or the learned objects. During detection the output of this discriminative model is used as input to an optimization procedure for the 3D-pose with respect to an observation model. Brachmann *et al.* [4] improve over this basic concept by beneficially incorporating the predicted object probabilities and coordinates into the observation model, which is formulated in form of an energy.

In this work we build a PF tracking framework with the energy based observation model of [4] and carefully design a proposal distribution that intelligently exploits the input RGB-D image as well as the output of the trained discriminative model. In this way we combine the robustness of the discriminative model with respect to changes in illumination and the robustness of the PF framework

with respect to occlusions. Furthermore, we use a two-way PF that builds on similar ideas as annealed PF approaches as previously proposed in [16] and [2]. The random forest approach is in spirit similar to boosting based approaches that have been examined in the area of tracking in the works [19, 13, 1, 27]. Finally, our approach also uses GPU rendering features to efficiently evaluate the observation model.

3 Method

Given a stream of RGB-D images of a moving object our goal is to estimate in each frame t the object pose H_t. We assume that a 3D model of the corresponding object is available. We define the pose H_t as the transformation that maps a point from the local coordinate system of the object to the coordinate system of the camera. We cast pose estimation as a tracking problem and solve it with a PF framework. In Sec. 3.1 we introduce a regression forest, that predicts object probabilities and local object coordinates. This output is used in several steps of our approach. Then, we briefly reiterate the basic PF framework (Sec. 3.2). We follow with a description of our motion model (Sec. 3.3) and our definition of particle likelihood (Sec. 3.4). Finally, in Sec. 3.5 we describe how we adapt sampling of particles according to a proposal distribution which concentrates on image areas where high particle likelihood is expected. This is the main component to facilitate efficient and robust pose tracking.

3.1 Discriminative Prediction of Object Probabilities and Object Coordinates from a Single RGB-D Image

In order to assess the likelihood of particles (Sec. 3.4), and to concentrate hypotheses sampling on promising image areas (Sec. 3.5) we utilize a discriminative function. This function takes local image patches as input and estimates the following two outputs for the center pixel of each patch: the probability $p(c)$ of the pixel belonging to object c, and its coordinate $\mathbf{y}_c$ in the object coordinate system. We follow exactly the setup of [4] to achieve this mapping densely for each pixel of the current frame t using decision forests.

Note that we train the forest jointly for multiple objects although, in this work, we consider tracking one pre-specified object c only. During test time, discriminative predictions are only calculated for object c. Other objects, which the forest might know, are not considered. However, the same forest can provide predictions for different objects, so that training has to be done only once. In the following we give a short summary on training and prediction.

Training. We train an ensemble $\mathcal{T}$ of decision trees T^j. Each tree is trained separately using a set of object images as well as neutral background images. Object images are segmented and show object c in different poses. Each pixel within a segmentation mask is furthermore annotated with its object coordinate $\mathbf{y}_c$. Trees operate on simple scale-invariant depth and RGB difference features [22]. During training we select for each node one feature out of a randomly generated pool that maximizes the standard information gain defined on a discrete

set of labels. These labels are obtained by separating object c into n_{cell} cells with a 3D grid resulting in n_{cell} discrete labels per object. One additional label is assigned to the background class.

We do not restrict tree depth but stop growing when less than a minimum number of pixels arrive at a node. At each leaf l^j we approximate the object probability $p(c|l^j)$ by the fraction of pixels at that leaf that belong to object c. The estimate of the object coordinate $\mathbf{y}_{c,l^j}$ is found by running mean-shift on all pixels of object c in leaf l^j and taking the largest mode.

Prediction. Each pixel of an input image is run through each tree in $\mathcal{T}$ and arrives at leafs $(l^1, \ldots, l^{|\mathcal{T}|})$. This results in a list of object probabilities $(p(c|l^1), \ldots, p(c|l^{|\mathcal{T}|}))$ and object coordinates $(\mathbf{y}_{c,l^1}, \ldots, \mathbf{y}_{c,l^{|\mathcal{T}|}})$ for each object c. The object probabilities of all trees are combined with Bayes rule over all known objects and the background class to obtain the final $p(c)$ for each pixel [4].

3.2 PF for 3D pose tracking

A PF approximates the current posterior distribution $p(H_t|Z_{1:t})$ of the pose H_t at time t given all previous observations $Z_{1:t}$. In each time step t the posterior distribution is represented by a set of samples $S_t = \{H_t^1, \ldots, H_t^N\}$, which are referred to as particles. Each particle has two velocity vectors $\mathbf{v}_t$ and $\mathbf{e}_t$ attached. They correspond to the previous translational and rotational motion respectively.

The PF requires an observation model and a motion model. The former describing the likelihood $p(Z_{t+1}|H_{t+1})$ of an observation given a pose. The latter describing the probability $p(H_{t+1}|H_t, \mathbf{v}_t, \mathbf{e}_t)$ of a pose given the previous pose H_t as well as the last velocity vectors. With each new frame $t+1$ a new set of particles S_{t+1} is found in three steps:

1. **Sampling:** For each particle H_t^i an intermediate particle $\hat{H}_{t+1}^i$ is sampled according to the motion model $p(\hat{H}_{t+1}^i|H_t^i, \mathbf{v}_t^i, \mathbf{e}_t^i)$. New velocities $\mathbf{v}_{t+1}^i$ and $\mathbf{e}_{t+1}^i$ are calculated (see supplemental note for details) using H_t^i and $\hat{H}_{t+1}^i$.
2. **Weighting:** Each intermediate particle is assigned a weight π_{t+1}^i, which is proportional to the likelihood $p(Z_{t+1}|\hat{H}_{t+1}^i)$ of the observed data given the pose $\hat{H}_{t+1}$.
3. **Resampling:** Finally, the set $S_{t+1} = \{H_{t+1}^1, \ldots, H_{t+1}^N\}$ of unweighted particles (with their attached velocities), is randomly drawn from $\{\hat{H}_{t+1}^1, \ldots, \hat{H}_{t+1}^N\}$ using probabilities proportional to the weights π_{t+1}^i.

The number of particles required to approximate the 6D posterior distribution can be drastically reduced if the sampling is concentrated in areas where one expects the true pose of the object. This is done using a proposal distribution $q(H_{t+1}|H_t, \mathbf{v}_t, \mathbf{e}_t)$ in the sampling step instead of the original motion model. To compensate for the fact that we sample from a different distribution, the calculation of weights has to be adjusted according to Eq. (19) in [10]:

$$\pi_{t+1}^i \propto p(Z_{t+1}|\hat{H}_{t+1}^i) \frac{p(\hat{H}_{t+1}^i|H_t^i, \mathbf{v}_t^i, \mathbf{e}_t^i)}{q(\hat{H}_{t+1}^i|H_t^i, \mathbf{v}_t^i, \mathbf{e}_t^i)} \tag{1}$$

In the following we describe the specifics of our implementation of the PF framework: the motion model, the observation likelihood, and finally, our main contribution, the construction of our proposal distribution. An overview of our tracking pipeline can be found in Fig. 1.

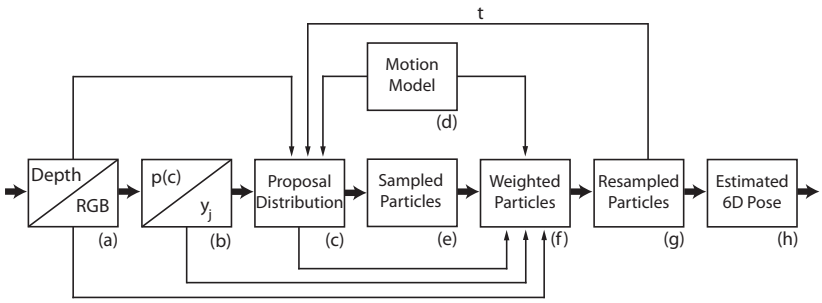

Fig. 1. Our tracking pipeline. For each frame t the RGB-D image(a) is processed by the forest to predict object probabilities and local object coordinates(b). We use the observed depth from the original image, the forest predictions and the particles from the last frame together with our motion model(d) to construct our proposal distribution(c). Particles are sampled(e) according to the proposal distribution, then weighted(f) and resampled(g). Our final pose estimate is calculated as mean of the resampled particles(h).

3.3 Our Motion Model

The motion model describes which movements of the object are plausible between two frames. Generally speaking, we assume our object roughly continues its previous motion, and that an additional random normally distributed translation is applied to our pose together with a random rotation around the center of the object. More specifically we assume the rotation to be around a uniformly chosen random axis and the angle of the rotation to be normally distributed. We will introduce a continuous probability distribution on SE(3) representing such a random motion. It will be reused in the context of our proposal distribution as described in Sec. 3.5.

Let R_t and T_t be the homogeneous 4x4 matrix representations of the rotational and translational component of the pose $H_t = T_t R_t$. We model the change of the pose between two time steps t and $t+1$ separately for both components: $H_{t+1} = T_{t+1} R_{t+1}$ with $T_{t+1} = \Delta_t^T T_t$, $R_{t+1} = \Delta_t^R R_t$ where Δ_t^T and Δ_t^R are 4x4 translation and rotation matrices, representing the change in the rotational and translational component respectively. Note that Δ_t^R contains the relative rotation around the object center and not the camera center. Translational change is defined as:

$$\Delta_t^T = T\big(\lambda_T \mathbf{v}_t + \boldsymbol{\omega}_T\big) \tag{2}$$

The vector $\boldsymbol{\omega}_T$ contains independent zero centered Gaussian noise with σ_T^2 variance. The symbol λ_T stands for a damping parameter. It determines how much the previous translation, described by the velocity vector $\mathbf{v}_t$ is continued. Finally,

$T(\mathbf{v})$ shall be the translation matrix corresponding to the translation vector $\mathbf{v}$. The rotational change is defined as:

$$\Delta_t^R = R(\boldsymbol{\omega}_R \theta) R(\lambda_R \mathbf{e}_t) \tag{3}$$

The symbol $\boldsymbol{\omega}_R$ stands for a random unit vector that defines the rotation axis of the random movement. Here θ is a Gaussian distributed zero centered random variable determining the rotation angle[2]. Its variance is σ_R^2. The symbol λ_R stands for a damping parameter. It determines how much the previous rotation, described by the rotational velocity vector $\mathbf{e}_t$ is continued. Finally, $R(\boldsymbol{e})$ shall stand for the rotation matrix corresponding to the rotation vector[3] $\boldsymbol{e}$.

Based on the model described above we can calculate the probability density $p(H_{t+1}|H_t, \mathbf{v}_t, \mathbf{e}_t)$ for a transition to pose H_{t+1} given the previous pose H_t. We approximate our motion model using the following density function $f(H_{t+1}; H_t^{pred}, \Sigma^{mm}, \kappa^{mm})$, which is described at the end of this section:

$$p(H_{t+1}|H_t, \mathbf{v}_t, \mathbf{e}_t) \approx f(H_{t+1}; H_t^{pred}, \Sigma^{mm}, \kappa^{mm}) \tag{4}$$

This function describes the probability of an arbitrary pose H_{t+1} with respect to the predicted pose H_t^{pred}, which is found by extrapolating previous motion given by the velocity vectors $\mathbf{v}_t$ and $\mathbf{e}_t$. The probability distribution depends on the variance in the translational component through Σ^{mm} and the variance of the rotational component through κ^{mm}. The diagonal matrix $\Sigma^{mm} = I\sigma_T^2$, the term $\kappa^{mm} = 1/\sigma_R^2$ and

$$H_t^{pred} = T(\lambda_T \mathbf{v}_t) T_t R(\lambda_R \mathbf{e}_t) R_t \tag{5}$$

The density function $f(H; H', \Sigma, \kappa)$ corresponds to the random motion described in Eqs. (2) and (3):

$$f(H; H', \Sigma, \kappa) = f_n(\mathbf{v}^{diff}; \mathbf{0}, \Sigma) f_{vm}(\theta^{diff}; 0, \kappa) \phi(\theta^{diff}) \tag{6}$$

It consists of a zero centered multivariate normal distribution $f_n(\mathbf{v}^{diff}; \mathbf{0}, \Sigma)$ and a zero centered von Mises distribution $f_{vm}(\theta^{diff}; 0, \kappa)$. While the vector $\mathbf{v}^{diff}$ denotes the translational difference between H' and H, the symbol θ^{diff} stands for the angle of the difference rotation. The normal distribution models the translational noise $\boldsymbol{\omega}_T$ introduced in Eq. (2). The von Mises distribution in Eq. (6) is a close approximation for a wrapped normal distribution, it models the Gaussian rotational noise introduced by θ in Eq 3. Note that the random rotation axis $\boldsymbol{\omega}_R$ of Eq. (3) has no influence on the density, since each rotation axis has equal probability. Also note that an additional factor $\phi(\theta^{diff})$ is necessary to map the 1D density over angles onto the group of rotations SO(3). A more detailed discussion of $\phi(\theta^{diff})$ can be found in the supplemental note.

[2] Please note, that because of its circular nature, applying rotations with the normally distributed angles θ will result in angles distributed in the interval between 0 and 2π according to a wrapped normal distribution. Such a distribution is difficult to handle and we will use a von Mises distribution as approximation.

[3] Direction and length of a rotation vector correspond to rotation axis and rotation angle, respectively.

3.4 Our Observation Likelihood

We use a likelihood formulation based on the energy $E(H)$ from [4]:

$$p(Z_{t+1}|H) \propto \exp(-\alpha E(H)) \tag{7}$$

where α is a control parameter determining how harshly larger energy values should be punished. The energy term is a weighted sum of three components $E^{\text{depth}}(H)$, $E^{\text{coord}}(H)$ and $E^{\text{obj}}(H)$. Detailed equations can be found in [4]. The depth component $E^{\text{depth}}(H)$ punishes deviations between the observed and rendered depth images within the object mask. The other two components are concerned with the output of the forest. The coordinate component $E^{\text{coord}}(H)$ punishes deviations between each pixels true object coordinates for pose H and the object coordinates predicted by the forest. The object component $E^{\text{obj}}(H)$ punishes deviation between the ideal segmentation obtained by rendering and the soft segmentations predicted by the trees in form of $p(c|l^j)$.

In contrast to [4], we use a simple modification of the depth term, that copes better with occlusion. Depth values that lie in front of the object can be explained by occlusion. This is not the case for depth values that lie behind the object. Our modified depth term accounts for this by reducing the threshold of possible punishment for values in front of the object. A detailed description can be found in the supplemental note.

3.5 Our Proposal Distribution

Our proposal distribution allows our method to cope with unexpected motion and occlusion, while maintaining high accuracy. It allows us to approximate the posterior distribution $p(H_t|Z_{1:t})$ more accurately with a small number of particles. The construction of our proposal distribution is described in the following, and subsumed in Fig. 2.

A proposal distribution describes the sampling of a new particle $\hat{H}^i_{t+1}$ on the basis of an old particle H^i_t. We define the proposal distribution $q(H_{t+1}|H_t, \mathbf{v}_t, \mathbf{e}_t)$ for the particle H^i_t as a mixture of two parts (Fig. 2(o)):

$$\begin{aligned} q(H^i_{t+1}|H^i_t, \mathbf{v}^i_t, \mathbf{e}^i_t) = (1-\alpha^{prop}) f(H^i_{t+1}; H^{i,pred}_t, \Sigma^{prop}, \kappa^{prop}) \\ +\alpha^{prop} f(H^i_{t+1}; H^{est}_{t+1}, \Sigma^{prop}, \kappa^{prop}) \end{aligned} \tag{8}$$

The mixture is governed by the weight α^{prop}. Both parts reuse the density function defined in Eq. (6) with variance parameters Σ^{prop} and κ^{prop}, which can be found in the supplemental note. The first part of Eq. (8) is centered on $H^{i,pred}_t$ which is the extrapolation of the current particle H^i_t according to our motion model as described in Eq. (5). Hence, $H^{i,pred}_t$ differs for each particle H^i_t. The second part of Eq. (8) is centered on a preliminary estimate H^{est}_{t+1} which is found based on the output of our discriminative function (Sec. 3.1). It does not depend on H^i_t, but is shared among all particles.

Regarding H^{est}_{t+1}, we will discuss two different ways to quickly obtain a good estimate: One way finds a local estimate H^{local}_{t+1}, the other way finds a global

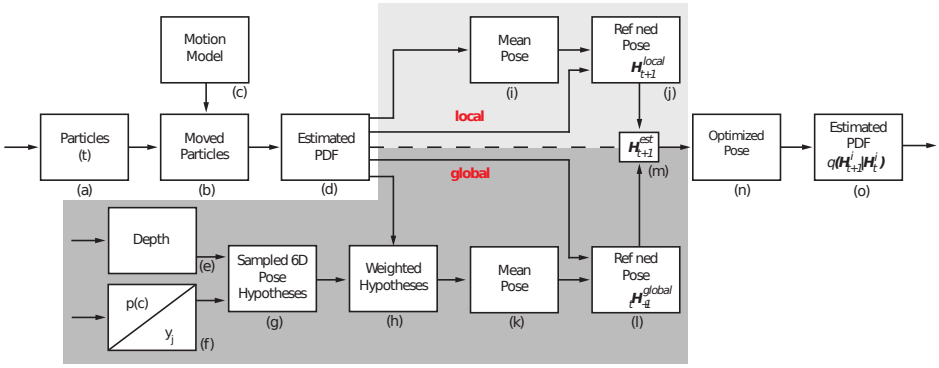

Fig. 2. To construct our proposal distribution we first calculate a continuous representation of the prior distribution for the pose at the current frame (a-d). Next we determine two pose estimates H_{t+1}^{local} (light gray) and H_{t+1}^{global} (dark gray). The local estimate is calculated based on a local search on propagated solutions via the motion model (i-j). The global estimate is based on the pose sampling scheme from [4] (e-l). We choose the particle with the lower energy and use it as starting point for a final optimization (n). Our final proposal distribution $q(H_{t+1}^i|H_t^i)$ for particle H_t^i is a mixture of two components (o): one centered on H_t^i and one on the newly found particle in (n). This figure represents component (c) of Fig. 1.

estimate H_{t+1}^{global}. While a proposal distribution based on a local estimate is sufficient in most cases, it may fail in situations with fast unexpected motion. The global estimate on the other hand depends on the quality of the discriminative prediction and may at times give noisy results. As a consequence, we apply a combination of the two approaches:

$$H_{t+1}^{est} = \text{argmin}_{H\in\mathcal{H}} E'(H); \mathcal{H} = \{H^{local}, H^{global}\} \tag{9}$$

Note, that this is our main technical contribution. Energy $E'(H)$ will be defined below. The preliminary estimate H_{t+1}^{est} is optimized (Fig. 2(n)) using a general purpose optimizer (details can be found in the supplemental note).

The remainder of this section is concerned with the calculation of H_{t+1}^{local} and H_{t+1}^{global}. First, however, we discuss how we represent prior knowledge used in both estimates.

Prior Knowledge. The proposal distribution should be an estimate of the posterior $p(H_{t+1}|Z_{1:t+1})$. Both of our estimates should thus include not only knowledge taken from the current observation i.e. the likelihood and results from the discriminative function, but also information from the previous particle set i.e. the prior. To include the prior we perform the following preparatory steps. We take the last set of particles $S_t = \{H_t^1, \ldots, H_t^N\}$ and move each particle according to the motion model (Fig. 2(a-c)). The result is an extrapolated set of particles $\tilde{S}_{t+1} = \{\tilde{H}_{t+1}^1, \ldots, \tilde{H}_{t+1}^N\}$. In order to obtain a continuous representation we reuse the distribution $f(H; H^{center}, \Sigma, \kappa)$ (Eq. (6)). We fit $f(H; H^{center}, \Sigma, \kappa)$ to the set $\tilde{S}_{t+1}$ (Fig. 2(d)). The resulting parameters are $\tilde{H}^{center}, \tilde{\Sigma}, \tilde{\kappa}$. For details on the fitting procedure, please refer to the supplemental note. The distribution

$f(H; \tilde{H}^{center}, \tilde{\Sigma}, \tilde{\kappa})$ is a representation of the knowledge we have about the pose at the current time $t+1$ without considering the current observation Z_{t+1}. It is a representation of the prior.

Local Estimate. To find H_{t+1}^{local}, we use $\tilde{H}^{center}$ (Fig. 2(i)) as starting point for refinement as described in [4] (Fig. 2(j)). This refinement is done by repeatedly finding inlier pixels. Their predicted object coordinates together with the observed depth values enable a rough but quick optimization using Kabsch algorithm. In order to include prior knowledge in the refinement we change the objective function to:

$$E'(H) = \alpha E(H) - \ln f(H; \tilde{H}^{center}, \tilde{\Sigma}, \tilde{\kappa}) \tag{10}$$

Because of this adjustment of the objective function the resulting H_{t+1}^{local} becomes a local maximum a posteriori (MAP) estimate.

Global Estimate. Calculation of the global estimate H_{t+1}^{global} is based on a sampling scheme similar to the one in [4]. We sample a set of m particles $\check{H}^i$ (Fig. 2(e-g)). Details can be found in the supplemental note. Then, the particles $\check{H}^i$ are weighted using the distribution $f(H; \tilde{H}^{center}, \tilde{\Sigma}, \tilde{\kappa})$ (Fig. 2(h)). Finally, their weighted mean is calculated (Fig. 2(k)) and used as initialization for the refinement (Fig. 2(l)), again using $E'(H)$ from Eq. (10) as objective function. This yields H_{t+1}^{global}.

4 Experiments

Some RGB-D object tracking datasets have been published in recent years. For example, Fanelli *et al.* [11] recorded a dataset to track human head poses using a Kinect camera. Song and Xiao [23] used a Kinect camera to record 100 RGB-D video sequences of arbitrary objects but do only provide 2D bounding boxes as ground truth. For our purpose, we found only one relevant dataset from Choi and Christensen [8]. It consists of 3D object models and synthetic test sequences. For further evaluation, we recorded a new more challenging and realistic dataset on which we compared our approach. Additionally, we conduct experiments to demonstrate that our proposal distribution achieves superior results when unexpected object motion occurs.

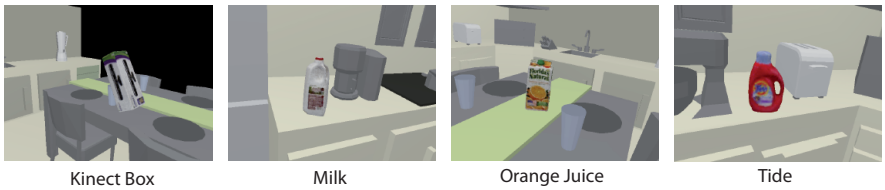

Fig. 3. Example images of the dataset provided by Choi and Christensen [8].

Dataset of Choi and Christensen [8]. The dataset of Choi and Christensen [8] provides four textured 3D models and four synthetic test sequences (1000

RGB-D frames). To generate the test sequences, each of the four objects was placed in a static texture-less 3D scene and the camera was slowly moved around the object. The authors provide the ground truth camera trajectory which is error-free since it was generated through rendering. Fig. 3 shows one image of each sequence.

To gather the training data for our random forest we rendered RGB-D images of each model. We sampled the full view sphere of the model regularly with fixed distance and including in-plane rotation. For the background set we used renderings from multiple 3D scenes from Google warehouse. We trained three decision trees with a maximum feature patch size of 20 pixel meter and $n_{cell} = 125$ discrete labels per object. We trained the trees for all 4 objects jointly. For each testing sequence the object to be tracked is assumed to be known, and predictions are only made for this object. Our PF uses 70 particles. The complete list of parameters is included in the supplemental note.

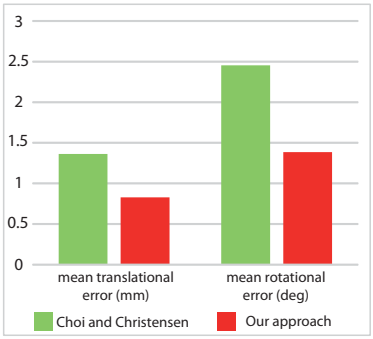

Fig. 4. Averaged translation and rotation RMSE on the dataset of [8].

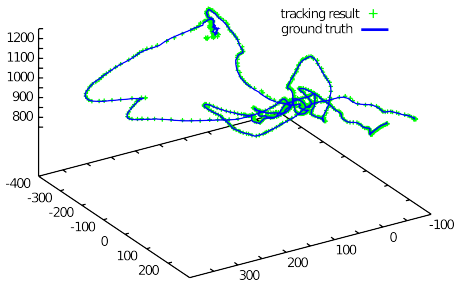

Fig. 5. Reconstructed motion trajectory (green) for one sequence of our dataset (Cat, sequence 1). Ground truth is depicted blue for comparison.

While testing we follow the evaluation protocol of Choi and Christensen [8] and compute the Root Mean Square Error (RMSE) of the translation parameters X,Y and Z and the rotation parameters Roll, Pitch and Yaw. We average the translational RMSE over three test runs, the coordinates (X,Y and Z), as well as over the four objects to obtain one translational error measure. We do the same for rotational RMSE. We compare to the numbers provided in [8] which also include results for the tracking implementation of the Point Cloud Library (PCL) [3]. We base our comparison on the results in [8] achieved with 12800 particles, for which the lowest error is reported.

Our method results in an average translational RMSE of 0.83 mm compared to 1.36 mm for [8], i.e. we achieve a 38% lower translational error (PCL: 18.7 mm). For the average rotational RMSE we report 1.38 deg compared to 2.45 deg in [8], which is 43% lower (PCL: 29.6 deg). We achieve these results while keeping the computation time on our system[4] comparable to the one reported

[4] Intel Core i7-3820 CPU @ 3.6GHz with a Nvidia GTX 550 TI GPU

in [8]. Figure 4 depicts the average RMSE over all objects. Detailed results including run-times can be found in Table 1.

Objects	Tracker	RMSE						
		X (mm)	Y (mm)	Z (mm)	Roll (deg)	Pitch (deg)	Yaw (deg)	Time (ms)
Kinect Box	PCL	43.99	42.51	55.89	7.62	1.87	8.31	4539
	Choi and Christensen	1.84	2.23	1.36	6.41	0.76	6.32	166
	Our	**0.83**	**1.67**	**0.79**	**1.11**	**0.55**	**1.04**	**143**
Milk	PCL	13.38	31.45	26.09	59.37	19.58	75.03	2205
	Choi and Christensen	0.93	1.94	1.09	3.83	**1.41**	3.26	**134**
	Our	**0.51**	**1.27**	**0.62**	**2.19**	1.44	**1.90**	135
Orange Juice	PCL	2.53	2.20	1.91	85.81	42.12	46.37	1637
	Choi and Christensen	0.96	1.44	1.17	1.32	**0.75**	1.39	**117**
	Our	**0.52**	**0.74**	**0.63**	**1.28**	1.08	**1.20**	129
Tide	PCL	1.46	2.25	0.92	5.15	2.13	2.98	2762
	Choi and Christensen	0.83	1.37	1.20	**1.78**	**1.09**	**1.13**	**111**
	Our	**0.69**	**0.81**	**0.81**	2.10	1.38	1.27	116

Table 1. Comparison of the translation error (X,Y,Z), rotation error (Roll, Pitch, Yaw) and computation time on the synthetic dataset of Choi and Christensen [8] with results from our method, [8] and the PCL.

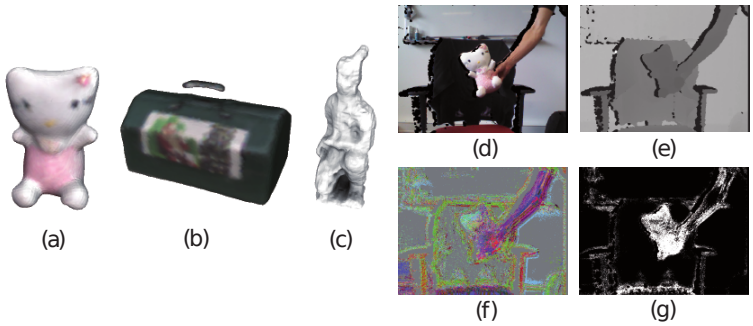

Fig. 6. (a)-(c) Our objects from left to right Cat, Toolbox, Samurai. (d) color frame and (e) depth frame recorded with the commercially available Kinect camera.(f) Probability map and (g) predicted 3D object coordinates from a single tree mapped to the RGB-cube for the object Cat.

Our dataset. The dataset which was provided by [8] is problematic for several reasons: testing sequences are generated synthetically without camera noise, and without occlusion. The objects are placed in a texture-less and static environment. In a static scene, a tracking method can in theory use the entire image to estimate the motion of the camera instead of estimating the motion of the object. Furthermore, the camera is moved around the object. The statistics of object motion when the camera is moved are very different from a situation where the camera is static and the object is moved. E.g., a complete vertical flip of the object is unlikely in the first scenario.

To address these issues we introduce our own dataset which consists of three objects. The objects were scanned in using Kinect, and six RGB-D testing sequences were recorded (350+ frames each). The objects are moved quickly in front of a static camera and are at times strongly occluded. Ground truth poses

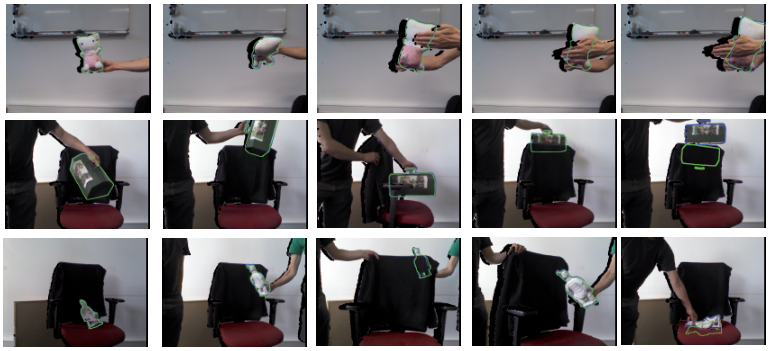

Fig. 7. Example images from our dataset. Blue object silhouettes depict ground truth and green silhouettes depict the estimated poses. The first four columns show correctly estimated poses and the last column missed poses.

were manually labeled by hand annotation followed by ICP. In Fig. 7 five images of each object are shown. For our dataset we trained decision trees as discussed in the previous section, but with renderings of our scanned-in objects, and a set of arbitrary RGB-D office images as background set. We keep all other training parameters as in the previously described experiment. We compare our approach to

Objects	Sequence	Ratio of frames used	Method		
			Our Approach		Brachmann *et al.*
			full proposal distribution	local proposal distribution	
			Accuracy	Accuracy	Accuracy
Cat	1	100%	100%	89%	66.8%
		33%	91.5%	90.4%	
	2	100%	99.4%	100%	44.2%
		33%	94.9%	87.8%	
Samurai	1	100%	96.3%	96.7%	72%
		33%	68.6%	52.4%	
	2	100%	92.3%	98.1%	33.7%
		33%	55.4%	29.1%	
Toolbox	1	100%	88.8%	88.5%	54.7%
		33%	81.2%	68.2%	
	2	100%	100%	100%	59.4%
		33	89.9%	34.5%	

Fig. 8. Accuracy measured on our dataset. Comparison of our full proposal distribution to the local proposal distribution and to [4]. Evaluation is done based on all frames and on every third frame of the image sequences.

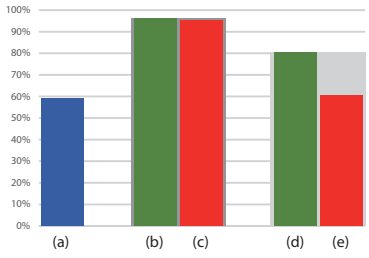

Fig. 9. Average accuracy over all sequences for (a) [4]. Our full proposal distribution using (b) all frames and (d) every third frame of the sequence. Our local proposal distribution using (c) all frames (e) every third frame of the sequence.

the one shot pose estimation from [4]. We exactly adhere to their training setup and parameters (3 trees, maximum patch size 20 pixel meter, 210 hypotheses per frame, Gaussian noise on training data). We measure accuracy as in [4] as the fraction of frames where the object pose was estimated correctly.

While our approach achieves 96.2% accuracy on average over all sequences the approach of [4] only estimates 58.9% of the poses correctly. Even though [4] is inherently robust to occlusion, the heavy occlusions in our dataset still cause it to fail. In contrast, our approach is able to estimate most poses correctly by using information from previous frames. The results are depicted in Fig. 9 (a)

and (b). Additionally, we computed rotational and translational distances to the ground truth for both methods. The distribution of errors for one sequence is depicted in Fig. 10. The plots again show the large number of outlier estimations of [4] (rightmost bins). The plots also reveal that concerning correct poses, our approach leads to much more precise estimations.

To show that our full proposal distribution (Sec. 3.5) increases the robustness of our method we conducted the following experiment: We define a simplified variant of the proposal distribution, which is based only on H_{t+1}^{local} to which we also apply the final optimization. We term this variant *local proposal distribution*. We use it together with 120 particles since it needs less computation time. For

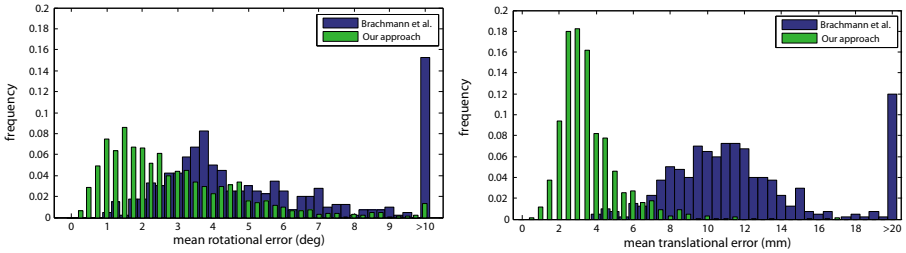

Fig. 10. Histogram of rotational and translational errors for our approach in comparison to [4], which is a single frame pose estimation framework.

this experiment, we artificially increase motion in our test sequences by using only every third frame. As before, we measure the number of correctly estimated poses. In this challenging setup the full proposal distribution achieves 80.3% accuracy on average while the local proposal distribution achieves only 60.4% accuracy. The results are depicted in Fig. 9 (b)-(e). Table 8 provides detailed information for the achieved accuracy on all sequences.

Fig. 5 shows the estimated object motion path for one sequence with fast motion. The plot illustrates precision and robustness of our approach.

5 Conclusion

We have introduced a novel method applying the concept of 3D object coordinate regression to 6-DOF pose tracking. We utilize predicted object coordinates in a proposal distribution, making our method very robust with regard to fast movements in combination with occlusion. We have evaluated our method on the dataset by Coi and Christensen and demonstrated that it yields superior results. The method was additionally evaluated using a new dataset, specially designed for RGB-D 6-DOF pose tracking, which will be made available to the community.

Acknowledgement. This work has partially been supported by the European Social Fund and the Federal State of Saxony within project VICCI (#100098171).

We thank Daniel Schemala for development of the manual pose annotation tool, we used to generate ground truth data.

References

1. Avidan, S.: Ensemble tracking. IEEE Trans. on PAMI **29** (2007) 261271
2. Azad, P., Munch, D., Asfour, T., Dillmann, R.: 6-DoF model-based tracking of arbitrarily shaped 3D objects. In: IEEE ICRA (2011) 5204–5209
3. Bersch, C., Pangercic, D., Osentoski, S., Hausman, K., Marton, Z.C., Ueda, R., Okada, K., Beetz, M.: Segmentation of textured and textureless objects through interactive perception. In: RSS Workshop on Robots in Clutter: Manipulation, Perception and Navigation in Human Environments (2012)
4. Brachmann, E., Krull, A., Michel, F., Shotton, J., Gumhold, S., Rother, C.: Learning 6d object pose estimation using 3d object coordinates. In: ECCV (2014)
5. Bray, M., Koller-Meier, E., Van Gool, L.: Smart particle filtering for 3D hand tracking. In: IEEE International Conference on Automatic Face and Gesture Recognition (2004) 675680
6. Chiuso, A., Soatto, S.: Monte carlo filtering on lie groups. In: 39th IEEE Conference on Decision and Control Volume 1 (2000) 304–309
7. Choi, C., Christensen, H.I.: 3D textureless object detection and tracking: An edge-based approach. In: IEEE/RSJ International Conference on IROS (2012) 38773884
8. Choi, C., Christensen, H.I.: RGB-D object tracking: A particle filter approach on GPU. In: IEEE/RSJ International Conference on IROS (2013) 1084–1091
9. Choi, C., Christensen, H.: Robust 3D visual tracking using particle filtering on the SE(3) group. In: 2011 IEEE ICRA (2011) 4384–4390
10. Doucet, A., Godsill, S., Andrieu, C.: On sequential monte carlo sampling methods for bayesian filtering. Statistics and Computing **10** (2000) 197–208
11. Fanelli, G., Weise, T., Gall, J., Van Gool, L.: Real time head pose estimation from consumer depth cameras. In: Pattern Recognition. Springer (2011) 101110
12. Gordon, N., Salmond, D., Smith, A.: Novel approach to nonlinear/non-gaussian bayesian state estimation. In: IEEE Radar and Signal Processing. Volume 2 (1993) 107–113
13. Grabner, H., Bischof, H.: Online boosting and vision. In: IEEE CVPR Volume 1 (2006) 260–267
14. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G.R., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: ACCV (2012) 548–562
15. Isard, M., Blake, A.: Contour tracking by stochastic propagation of conditional density. In: ECCV (1996) 343356
16. Klein, G., Murray, D.W.: Full-3D edge tracking with a particle filter. In: BMVC (2006) 11191128
17. Kwon, J., Choi, M., Park, F.C., Chun, C.: Particle filtering on the euclidean group: framework and applications. Robotica **25** (2007) 725–737
18. McElhone, M., Stuckler, J., Behnke, S.: Joint detection and pose tracking of multi-resolution surfel models in RGB-D. In: IEEE ECMR (2013) 131137
19. Okuma, K., Taleghani, A., De Freitas, N., Little, J.J., Lowe, D.G.: A boosted particle filter: Multitarget detection and tracking. In: Computer Vision-ECCV (2004) 2839
20. Pupilli, M., Calway, A.: Real-time camera tracking using known 3d models and a particle filter. In: IEEE ICPR Volume 1 (2006) 199203
21. Rios-Cabrera, R., Tuytelaars, T.: Discriminatively trained templates for 3d object detection: A real time scalable approach. In: IEEE ICCV (2013) 2048–2055

22. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: IEEE CVPR (2013) 29302937
23. Song, S., Xiao, J.: Tracking revisited using rgbd camera: Unified benchmark and baselines. In: ICCV (2013) 233–240
24. Stckler, J., Behnke, S.: Multi-resolution surfel maps for efficient dense 3D modeling and tracking. Journal of Visual Communication and Image Representation **25** (2014) 137–147
25. Taylor, J., Shotton, J., Sharp, T., Fitzgibbon, A.W.: The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In: IEEE CVPR (2012) 103–110
26. Teuliere, C., Marchand, E., Eck, L.: Using multiple hypothesis in model-based tracking. In:IEEE ICRA (2010) 4559–4565
27. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. International Journal of Computer Vision **75** (2007) 247–266