6-DOF Model Based Tracking via Object Coordinate Regression
Supplemental Note

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

TU Dresden, Dresden, Germany

The supplemental material is not necessary to understand the paper. Please find attached three supplemental videos as well as a note discussing them.
As mentioned in the paper this supplemental note discusses the following points:

- Calculation of the velocity vectors
- Details on the implementation
- Details on the modified depth term
- Details on the factor $\phi(\theta^{diff})$ in Eq. (6)
- Details fitting the continuous distribution
- Complete list of parameter settings

1 Calculation of the Velocity Vectors

In our particle filter implementation velocity vectors $\mathbf{v}_t^i$ and $\mathbf{e}_t^i$ are attached to each particle H_t^i. They describe the previous translational and rotational motion for the particle. After each sampling step (see Sec. 3.2) new velocity vectors $\mathbf{v}_{t+1}^i$ and $\mathbf{e}_{t+1}^i$ are calculated from the original particle H_t^i and the newly sampled preliminary particle $\hat{H}_{t+1}^i$. The new translational velocity vector is calculated as:

$$\mathbf{v}_{t+1}^i = \hat{\mathbf{t}}_{t+1}^i - \mathbf{t}_t^i, \tag{1}$$

where $\hat{\mathbf{t}}_{t+1}^i$ and $\mathbf{t}_t^i$ are the 3D vectors corresponding to the translational components of the new preliminary particle $\hat{H}_{t+1}^i$ and the original particle H_t^i, respectively.

The new rotational velocity vector is calculated as the rotation vector of the difference rotation between the rotational components:

$$\mathbf{e}_{t+1}^i = \mathbf{r}(\hat{R}_{t+1}^i {R_t^i}^{-1}), \tag{2}$$

where $\hat{R}_{t+1}^i$ and R_t^i are the matrix representations of the rotational components of the new preliminary particle $\hat{H}_{t+1}^i$ and the original particle H_t^i, respectively. The term $\mathbf{r}(A)$ shall represent the rotation vector corresponding to a rotation matrix A.

2 Details on the Implementation

2.1 Evaluation of the Forest

We found the per frame evaluation time of the forest for the entire 640 by 480 image to be between 100 and 200ms. To increase the speed of our method we evaluate the forest only for every second pixel in x- and y-direction. Additionally we consider only pixels in a square window around the last estimated position of the object projected onto the image. The width of the window in pixels is calculated as

$$w(H) = -1.2 f \delta_c / z, \tag{3}$$

where f is the focal length, δ_c is the objects diameter and z is the object's z-component of the translation from H. We set the object probability (see Eq. (1) in [1]) $p_{c,i} = 0$ for all pixels outside the window and every other second pixel in x and y-direction.

2.2 Energy Evaluation

As in [1] the calculation of the energy $E(H)$ is performed on the Graphics Processing Unit (GPU). Since we evaluate the forest only in every second pixel in x- and y-direction, we can also reduce the resolution of the rendered images by half.

2.3 Sampling

In the construction of our proposal distribution we use a sampling scheme similar to the one in [1] (see Fig. 2(g)). We will now describe in detail where our procedure differs from [1].

The sampling process described in [1] draws a random pixel according to $p_{c,i}$ from the entire image followed by two more in its vicinity. We in contrast, consider only pixels inside a square window around the projected center of $\tilde{H}^{center}$. We calculate the width of the window as

$$w(\tilde{H}^{center}) = -f \delta_c / \tilde{z}, \tag{4}$$

where $\tilde{z}$ is the object's z-component of the translation from $\tilde{H}^{center}$. In [1] an error is calculated for each sampled hypothesis by mapping the predicted object coordinates of the three pixels into camera space using the Hypothesis H and comparing them with the observed 3D camera coordinates. Hypotheses are accepted, if their error is below 5% of the objects diameter δ_c. We use a different error measure, based on the assumption, that the distance between two points in camera and object space should be identical. We calculate the Euclidean distance between each possible pair of points out of the three. We do this in camera and object space and for each pair compute the difference between the two. Our error measure is defined as the maximum of these differences. We accept a hypothesis whenever its error is smaller than the objects diameter δ_c. While in [1] sampling is repeated until 210 hypotheses are accepted, we sample 500 times. In cases where less than 5 hypotheses are accepted, we stop calculation of the global estimate and use only the local estimate.

2.4 Optimization

As the final step in the construction of our proposal distribution (Sec. 3.5) we perform optimization using a general purpose optimization algorithm. We use the Constrained Optimization BY Linear Approximations (COBYLA) algorithm by Powell [2]. We use the implementation from the NLopt library [3].

3 Details on the Modified Depth Term

To cope better with occlusion, we use a simple modification of the depth term in our energy. 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 modification accounts for this by reducing the threshold of possible punishment for values in front of the object.

For each pixel j Brachmann et al. [1] use a robust error function

$$f(d_j, d_j^*(H)) = \min\left(||\mathbf{x}(d_j) - \mathbf{x}(d_j^*(H))||, \tau_d\right)/\tau_d, \tag{5}$$

in the depth term. It limits the punishment of depth deviation. We use the following modification:

$$f^{occ}(d_j, d_j^*(H)) = \begin{cases} \min\left(||\mathbf{x}(d_j) - \mathbf{x}(d_j^*(H))||, \tau_d\right)/\tau_d & \text{if } d_j < d_j^*(H) \\ \min\left(||\mathbf{x}(d_j) - \mathbf{x}(d_j^*(H))||, \tau_d^{occ}\right)/\tau_d & \text{else} \end{cases}, \tag{6}$$

where τ_d^{occ} is an additional threshold.

4 Details on the Factor $\phi(\boldsymbol{\theta^{diff}})$ in Eq. (6)

In order to map the probability density (up to a constant factor) from the one dimensional von Mises distribution to the 3D group of rotations SO(3) we require a factor

$$\phi(\theta^{diff}) = \frac{1}{16\pi \sin^2 \frac{\theta^{diff}}{2}}, \tag{7}$$

depending on the angular difference θ^{diff}.

We will now discuss this factor and its origin in more detail. The factor compensates two mappings: Firstly a coordinate transform from spherical coordinates to Cartesian coordinates inside the tangential space around the source location and secondly the transport to the tangential space of the target location H^{center}.

4.1 Mapping from Spherical Coordinates to Tangential Space

Let us now discuss the first of the two mappings. In the 3D tangential space a vector corresponds to a rotation. The direction defines the axis of rotation and the length of the vector is equal to the angle of rotation. Since all axis are assumed to have equal probability, each particular angle θ^{diff} corresponds to a sphere of radius θ^{diff} centered at the origin of the tangential space at H^{center}. Our von Mises distribution is defined in

the space of possible angles, it is thus a distribution of these radii. Since the surface of these spheres depends quadratically on the radius θ^{diff}, we have to apply the quadratic factor

$$\hat{\phi}(\theta^{diff}) = \frac{1}{4\pi\theta^{diff^2}} \tag{8}$$

to transform the densities from our von Mises distribution, to this tangent space. The larger the difference angle θ^{diff} the smaller will be the resulting densities in the tangent space.

4.2 Mapping from Source to Target Tangential Space

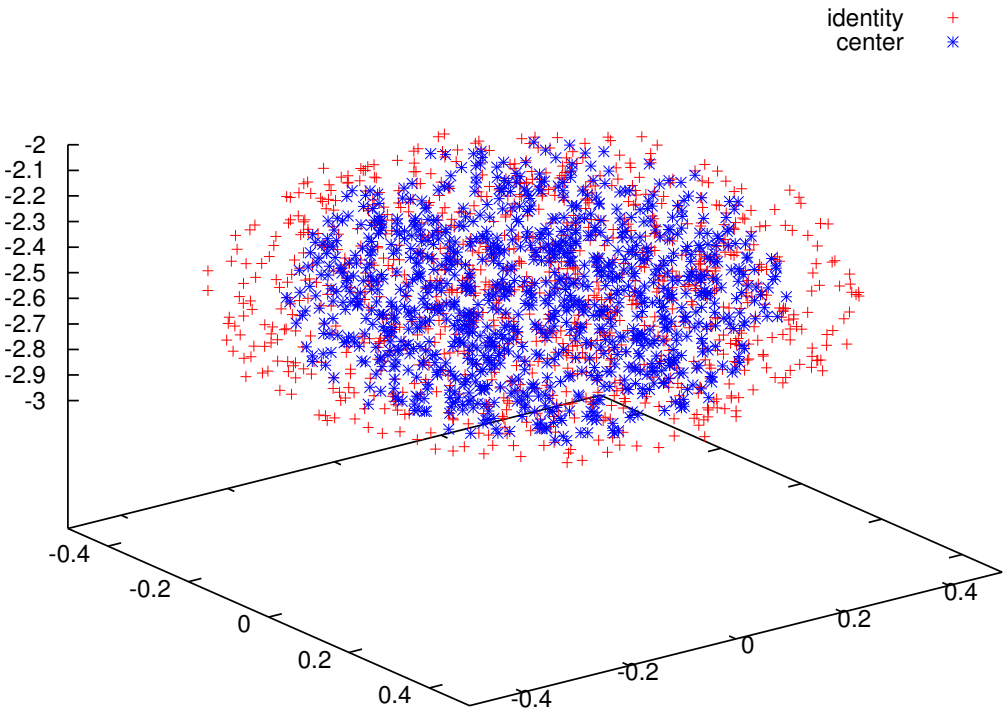

Fig. 1. The same set of rotations depicted once in the tangential space around the identity rotation (red) and once in the tangential space around the center of the set (blue). The density in the latter case is larger. The data points were shifted to lie on top of each other for comparison.

Calculating a density at a specific position x on the manifold of rotations SO(3) should be done in the tangential space at the position x. Using a different tangential space will yield different density estimates (see Fig. 1). To map the density calculated in our source tangential space to the tangential space around the target position, we have

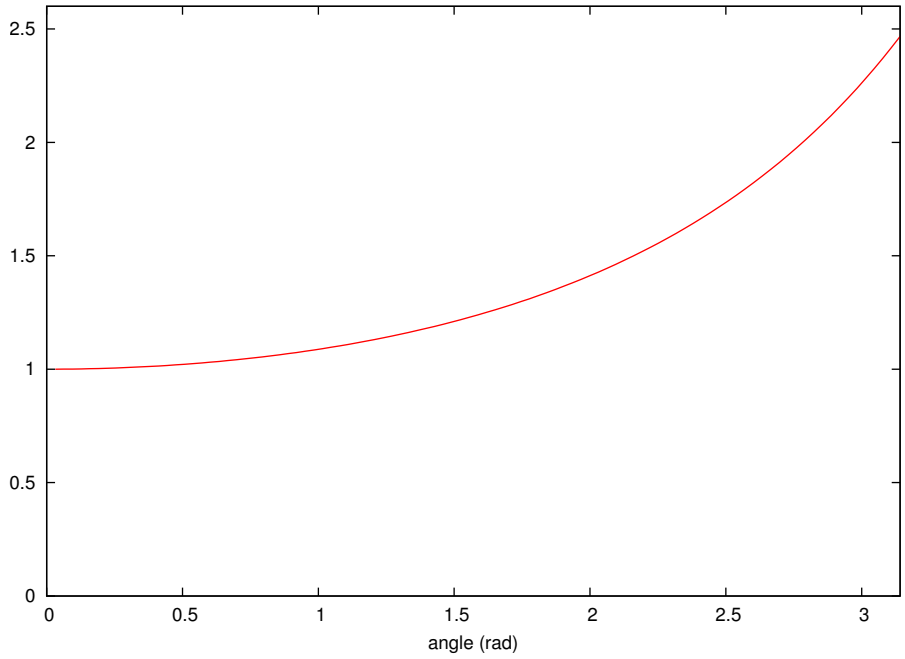

Fig. 2. The fuction $\check{\phi}(\theta^{diff})$ that maps densities between tangential spaces. Note that the difference angle θ^{diff} can only have values between 0 and π.

to use the following factor

$$\check{\phi}(\theta^{diff}) = \left(\frac{\frac{\theta^{diff}}{2}}{\sin\left(\frac{\theta^{diff})}{2}\right)} \right)^2 . \tag{9}$$

The larger the angular distance between source and target rotation the larger will be the resulting factor (see Fig. 2). Together with supplementary Eq. (8) we can calculate

$$\phi(\theta^{diff}) = \hat{\phi}(\theta^{diff})\check{\phi}(\theta^{diff}) = \frac{1}{16\pi \sin^2 \frac{\theta^{diff}}{2}} . \tag{10}$$

In our experiments we use an approximate mapping, by assuming $\phi(\theta^{diff}) = 1$. The benefits of using supplementary Eq. (10) instead are yet to be investigated. A discussion of tangential spaces for rotations can be found in [4].

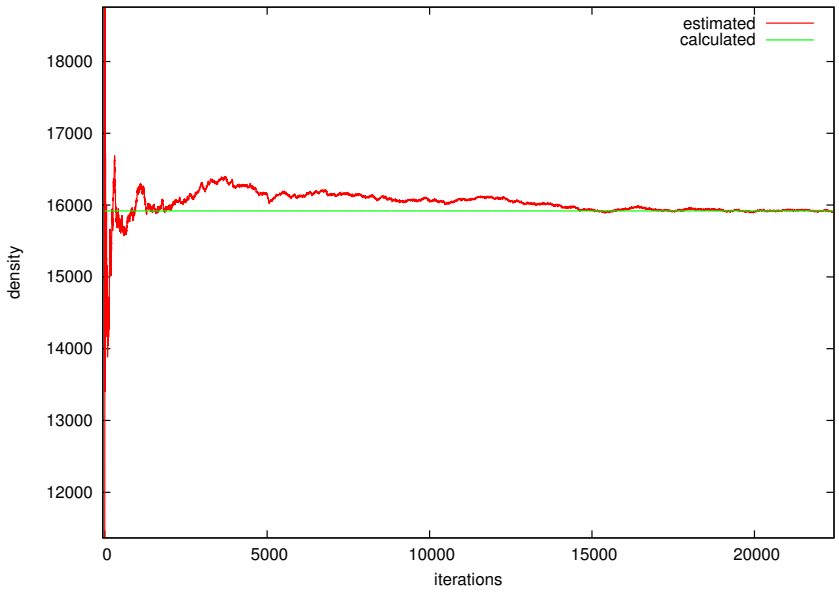

Fig. 3. Comparison of the numerically estimated density (red) and the density calculated with supplementary Eq. (10) (green). We generated angles from a one dimensional circular uniform distribution and picked random rotation axis to sample rotations. These samples were than multiplied with a rotation of $\theta = 0.05$rad around the x-axis. We estimated the normalized density of samples at the origin in the tangent space around zero by counting the number of samples that fell inside a small sphere of radius $\epsilon = 0.0015$rad, and dividing the number by the sphere volume and by the total number of samples we used. The estimated density converges towards the vale calculated by supplementary Eq. (10).

5 Details on Fitting the Continuous Distribution

During the construction of our proposal distribution we fit a continuous distribution $f(H; H^{center}, \Sigma, \kappa)$ to a set of extrapolated particles $\tilde{S}_{t+1}$. We will now describe the fitting process in detail.

We calculate the translational component of $\tilde{H}^{center}$ as the mean translation of $\tilde{S}_{t+1}$. We then use it to calculate the covariance matrix $\tilde{\Sigma}$ of the translational components of $\tilde{S}_{t+1}$.

To find our estimate $\tilde{\kappa}$ for the concentration parameter we calculate the angle $\tilde{\theta}^{diff,i}$ of the difference rotation $\tilde{R}^{i\,-1}\tilde{R}^{center}$ between the rotational component $\tilde{R}^{center}$ of $\tilde{H}^{center}$ and the rotational component $\tilde{R}^{i}$ of each $\tilde{H}^{i} \in \tilde{S}_{t+1}$. Since we do not know the direction of the rotations we apply a random sign to each $\tilde{\theta}^{diff,i}$ and finally use Eq. (4) from [5] to compute $\tilde{\kappa}$. Note that we add an additional constant $c = 1e-4$ to the denominator for numerical stability.

6 Complete List of Parameter Settings

The following settings were used in all experiments:

Training Parameters	
maximum feature offset:	20 pixel meters
number of features generated at each node:	1000
ratio of 'da-d' to 'da-rgb' features	0.5
number of trees $\lvert\mathcal{T}\rvert$:	3
random pixels per image to learn tree structure:	1000
random pixels per image to learn leaf distributions:	5000
stopping criterion: minimum number of pixels per node:	50

General Testing Parameters	
number of particles N:	70
control parameter α:	20
depth comp. weight λ^{depth}:	10
coordinate comp. weight λ^{coord}:	2
object comp. weight λ^{obj}:	10
threshold τ_d used in E_c^{depth}:	50 mm
additional threshold τ_d^{occ} used in E_c^{depth}:	30 mm
threshold τ_y used in E_c^{coord}:	$(0.2 \cdot \delta_c)^2$
threshold τ_{pc} used in E_c^{coord}:	10^{-8}
number of Hypothesis to be sampled:	500
threshold used during sampling of poses:	$0.05 \cdot \delta_c$
inlier threshold used in refinement:	20 mm

Proposal Distribution Parameters	
covariance matrix for prop. dist. Σ^{prop}:	$I\sigma_T^{prop}$
std of random translation in prop.dist. σ_T^{prop}:	2mm
std of random rotation in prop.dist. $\sigma_R^{prop} = 1/\sqrt{\kappa^{prop}}$:	0.01rad
concentration parameter in prop.dist. $\kappa^{prop} = 1/{\sigma_R^{prop}}^2$:	10000
maximum iterations for refinement of H_{t+1}^{global}:	15
maximum iterations for refinement of H_{t+1}^{local}:	15
maximum iterations for final optimization of H_{t+1}^{est}:	30
mixture weight in prop. dist. α^{prop}:	0.5

The following motion model parameters were used in all experiments performed on the dataset of Choi and Christensen [6]:

Motion Model Parameters (Dataset from Choi and Christensen [6])	
damping parameter for translation λ_T:	0.7
damping parameter for translation λ_R:	0.6
std of random translation σ_T:	14mm
std of random rotation in motion model $\sigma_R = 1/\sqrt{\kappa^{mm}}$:	0.05rad
concentration parameter in motion model $\kappa^{mm} = 1/\sigma_R^2$:	400

The following motion model parameters were used in all experiments performed on our data set:

Motion Model Parameters (Our Dataset)	
damping parameter for translation λ_T:	0.7
damping parameter for translation λ_R:	0.6
std of random translation σ_T:	20mm
std of random rotation in motion model $\sigma_R = 1/\sqrt{\kappa^{mm}}$:	0.15rad
concentration parameter in motion model $\kappa^{mm} = 1/\sigma_R^2$:	$44.\bar{4}4$

References

1. Brachmann, E., Krull, A., Michel, F., Shotton, J., Gumhold, S., Rother, C.: Learning 6d object pose estimation using 3d object coordinates. In: Proceedings of the 14th European Conference on Computer Vision. ECCV (2014)
2. Powell, M.: A direct search optimization method that models the objective and constraint functions by linear interpolation. In: Advances in optimization and numerical analysis: proceedings of the Sixth Workshop on Optimization and Numerical Analysis, Oaxaca, Mexico, Kluwer Academic Pub (1994) 51
3. Johnson, S.G.: The NLopt nonlinear-optimization package. (http://ab-initio.mit.edu/nlopt) Accessed: 2014-09-30.
4. Grassia, F.S.: Practical parameterization of rotations using the exponential map. Journal of Graphics Tools **3** (1998) 29–48
5. Sra, S.: A short note on parameter approximation for von mises-fisher distributions: and a fast implementation of $I_s(x)$. Computational Statistics **27** (2012) 177–190

6. Choi, C., Christensen, H.I.: RGB-D object tracking: A particle filter approach on GPU. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE (2013) 1084–1091