This ACCV 2020 paper, provided here by the Computer Vision Foundation, is the author-created version.
The content of this paper is identical to the content of the officially published ACCV 2020
LNCS version of the paper as available on SpringerLink: https://link.springer.com/conference/accv

A Sparse Gaussian Approach to Region-Based 6DoF Object Tracking

Manuel Stoiber[1,2][0000−0002−0762−9288], Martin Pfanne[1][0000−0003−2076−4772], Klaus H. Strobl[1][0000−0001−8123−0606], Rudolph Triebel[1,2][0000−0002−7975−036X], and Alin Albu-Schäffer[1,2][0000−0001−5343−9074]

[1] German Aerospace Center (DLR), 82234 Wessling, Germany
manuel.stoiber@dlr.de
[2] Technical University of Munich (TUM), 80333 Munich, Germany

Abstract. We propose a novel, highly efficient sparse approach to region-based 6DoF object tracking that requires only a monocular RGB camera and the 3D object model. The key contribution of our work is a probabilistic model that considers image information sparsely along correspondence lines. For the implementation, we provide a highly efficient discrete scale-space formulation. In addition, we derive a novel mathematical proof that shows that our proposed likelihood function follows a Gaussian distribution. Based on this information, we develop robust approximations for the derivatives of the log-likelihood that are used in a regularized Newton optimization. In multiple experiments, we show that our approach outperforms state-of-the-art region-based methods in terms of tracking success while being about one order of magnitude faster. The source code of our tracker is publicly available.[1]

1 Introduction

Tracking a rigid object and estimating its 6DoF pose is an essential task in computer vision that has a wide range of applications, from robotics to augmented reality. The aim is to estimate both the rotation and translation of an object relative to the camera from consecutive image frames. Typical challenges include partial occlusions, object ambiguities, appearance changes, motion blur, background clutter, and real-time requirements. To address those issues, many approaches have been proposed. Based on surveys [1, 2], as well as recent developments, methods are typically differentiated by their use of key-points, explicit edges, template images, deep learning, depth information, and image regions.

While methods based on key-points [3, 4] are very popular, rich texture is required, which limits the range of suitable objects. This is also the case for template-based approaches [5, 6]. Methods that use explicit edges [7–9], while more suitable for texture-less objects, have the disadvantage of struggling with background clutter. Deep-learning-based approaches [10–12] have also been used but are either computationally expensive or do not reach state-of-the-art results.

[1] https://github.com/DLR-RM/RBGT

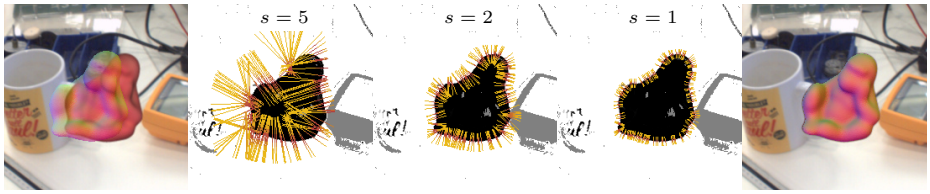

Fig. 1. Example of the optimization process for the *ape* object on the *RBOT* dataset [19]. The outermost images display a rendered overlay before and after the optimization. The images in the middle visualize pixel-wise posteriors that describe the probability of a pixel belonging to the background. White pixels indicate $p_b = 1$. Also, the images show orange correspondence lines that converge toward the final pose with decreasing scale s. High probabilities for the contour position are illustrated in red. Notice that during the operation of the tracker, pixel-wise posteriors are only calculated along lines.

Another recent development are methods that use depth cameras [13–16], which provide good results but depend heavily on depth image quality. Because of their capability to track texture-less objects in cluttered scenes, using only a monocular RGB camera, region-based methods [17–21] have become increasingly popular. However, while they reach state-of-the-art results, most methods feature computationally expensive, dense formulations that hardly run in real-time.

To overcome this issue, we propose a novel, sparse approach to region-based 6DoF object tracking that is based on correspondence lines (see Fig. 1). Also, we prove that the developed probabilistic model follows a Gaussian distribution and use this information in a regularized Newton optimization. Finally, in multiple experiments on the *RBOT* dataset [19], we show that our algorithm outperforms state-of-the-art methods both in terms of runtime and tracking success.

2 Related Work

In general, region-based methods differentiate between a foreground area that corresponds to the object and a background area. To model the membership of each pixel, differences in image statistics, such as color, are used. Based on the 3D geometry of the object, the goal is to find the pose that best explains the two regions. While early approaches [22, 23] treated segmentation and pose tracking as independent problems that are solved sequentially, [24] combined both stages to increase tracking robustness. Building on this approach and including the pixel-wise posterior membership suggested by [25], [17] developed *PWP3D*, a real-time capable algorithm that uses a level-set pose embedding.

Based on the concepts of *PWP3D*, multiple enhancements to incorporate additional information, extend the segmentation model, or improve efficiency were suggested. To fuse depth information, [26] added a term based on the *Iterative Closest Point (ICP)* approach to the energy function. Another method, introduced by [18], maximizes the probability of a model that tightly couples region and depth information. Recently, methods that incorporate texture information

were suggested by both [21] and [27]. To better handle occlusion, learning-based object segmentation was proposed by [28]. The incorporation of orientation information from an inertial sensor was presented by [29]. To improve the segmentation model, [30] introduced a boundary term to consider spatial distribution regularities of pixels. Another approach, that was inspired by [31], proposed the use of local appearance models [32]. The idea was later enhanced with the development of temporally consistent local color histograms [33, 19]. The most recent method introduced by [20], which is based on [27], uses polar-based region partitioning and edge-based occlusion detection to further improve tracking.

With respect to efficiency, enhancements such as a hierarchical rendering approach with Levenberg-Marquardt optimization [29], a simplified signed distance function [21], or a Gauss-Newton approach [19] were suggested. Another idea by [26] suggests to precompute contour points and use a sparse calculation of the energy function along rays. Starting from those ideas, this work focuses on the development of a highly efficient, sparse approach to region-based tracking. To keep complexity at a minimum, we use the global segmentation model of *PWP3D* and do not consider additional information. Notice, however, that our formulation is general enough to incorporate most of the discussed ideas.

3 Probabilistic Model

In this section, basic mathematical concepts are defined. This is followed by an introduction to our sparse probabilistic model. Finally, we extend this model and develop a discrete scale-space formulation to improve computational efficiency.

3.1 Preliminaries

In this work, 3D model points are defined by $\boldsymbol{X}_i = \begin{bmatrix} X_i & Y_i & Z_i \end{bmatrix}^\top \in \mathbb{R}^3$ or the corresponding homogeneous form $\widetilde{\boldsymbol{X}}_i = \begin{bmatrix} X_i & Y_i & Z_i & 1 \end{bmatrix}^\top$. We denote a color image by $\boldsymbol{I} \colon \boldsymbol{\Omega} \to \{0, \dots, 255\}^3$, with the image domain $\boldsymbol{\Omega} \subset \mathbb{R}^2$. The RGB values $\boldsymbol{y}_i$ at image coordinate $\boldsymbol{x}_i = \begin{bmatrix} x_i & y_i \end{bmatrix}^\top \in \mathbb{R}^2$ are described by $\boldsymbol{y}_i = \boldsymbol{I}(\boldsymbol{x}_i)$. To project a 3D model point into an undistorted image, we use the pinhole camera model

$$\boldsymbol{x}_i = \boldsymbol{\pi}(\boldsymbol{X}_i) = \begin{bmatrix} \frac{X_i}{Z_i} f_x + p_x \\ \frac{Y_i}{Z_i} f_y + p_y \end{bmatrix}, \tag{1}$$

with f_x and f_y the focal lengths and p_x and p_y the principal point coordinates for the directions x and y given in pixels.

To describe the relative pose between the model reference frame M and the camera reference frame C, we use the homogeneous matrix ${}_{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}$ and calculate

$${}_{\mathrm{C}}\widetilde{\boldsymbol{X}}_i = {}_{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}} {}_{\mathrm{M}}\widetilde{\boldsymbol{X}}_i = \begin{bmatrix} {}_{\mathrm{C}}\boldsymbol{R}_{\mathrm{M}} & {}_{\mathrm{C}}\boldsymbol{t}_{\mathrm{M}} \\ \boldsymbol{0} & 1 \end{bmatrix} {}_{\mathrm{M}}\widetilde{\boldsymbol{X}}_i, \tag{2}$$

with ${}_{\mathrm{C}}\widetilde{\boldsymbol{X}}_i$ a 3D model point represented in the camera reference frame C, ${}_{\mathrm{M}}\widetilde{\boldsymbol{X}}_i$ a 3D model point represented in the model reference frame M, ${}_{\mathrm{C}}\boldsymbol{R}_{\mathrm{M}} \in \mathbb{SO}(3)$

the rotation matrix for the transformation from M to C, and ${}_{\mathrm{C}}\boldsymbol{t}_{\mathrm{M}} \in \mathbb{R}^3$ the translation vector for the transformation from M to C.

For small rotations, the angle-axis representation, which is a minimal representation, is used. With the exponential map, the rotation matrix writes as

$$\boldsymbol{R} = \exp([\boldsymbol{r}]_\times) = \boldsymbol{I} + [\boldsymbol{r}]_\times + \frac{1}{2!}[\boldsymbol{r}]_\times^2 + \frac{1}{3!}[\boldsymbol{r}]_\times^3 + ..., \tag{3}$$

where $[\boldsymbol{r}]_\times$ represents the skew-symmetric matrix of $\boldsymbol{r} \in \mathbb{R}^3$. Linearizing Eq. (3), by neglecting higher-order terms of the series expansion, the linear variation of a 3D model point represented in the camera reference frame C is described by

$${}_{\mathrm{C}}\widetilde{\boldsymbol{X}}_i^+ = \begin{bmatrix} {}_{\mathrm{C}}\boldsymbol{R}_{\mathrm{M}} & {}_{\mathrm{C}}\boldsymbol{t}_{\mathrm{M}} \\ \boldsymbol{0} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{I} + [\boldsymbol{\theta}_{\mathrm{r}}]_\times & \boldsymbol{\theta}_{\mathrm{t}} \\ \boldsymbol{0} & 1 \end{bmatrix} {}_{\mathrm{M}}\widetilde{\boldsymbol{X}}_i, \tag{4}$$

with the variated model point ${}_{\mathrm{C}}\widetilde{\boldsymbol{X}}_i^+$, the rotational variation $\boldsymbol{\theta}_{\mathrm{r}} \in \mathbb{R}^3$, and the translational variation $\boldsymbol{\theta}_{\mathrm{t}} \in \mathbb{R}^3$. In general, variated variables that depend on the full variation vector $\boldsymbol{\theta}^\top = \begin{bmatrix} \boldsymbol{\theta}_{\mathrm{r}}^\top & \boldsymbol{\theta}_{\mathrm{t}}^\top \end{bmatrix}$ are indicated by a plus operator. Notice that it is more natural to variate a model point in the reference frame M instead of C since the object is typically moved significantly more than the camera.

3.2 General Formulation

In contrast to most state-of-the-art algorithms, we do not compute a joint probability over the entire image. Instead, inspired by *RAPID* [7], we introduce a sparse model where the probability is only calculated along a small set of entities that we call correspondence lines. Note that the name is motivated by the term *correspondence points* used in *ICP*, since we also first define correspondences and then optimize with respect to them. A correspondence line is described by a center $\boldsymbol{c}_i = \begin{bmatrix} c_{xi} & c_{yi} \end{bmatrix}^\top \in \mathbb{R}^2$ and a normal vector $\boldsymbol{n}_i = \begin{bmatrix} n_{xi} & n_{yi} \end{bmatrix}^\top \in \mathbb{R}^2$, with $\|\boldsymbol{n}_i\|_2 = 1$. Both values are defined by projecting a 3D contour point $\boldsymbol{X}_i$ and an associated vector normal to the contour $\boldsymbol{N}_i$ into the image. With the distance $r \in \mathbb{R}$ from the center, pixels on the line are described by rounding as follows

$$\boldsymbol{x}_{\mathrm{cl}i}(r) = \lfloor \boldsymbol{c}_i + r\ \boldsymbol{n}_i + \boldsymbol{0.5} \rfloor. \tag{5}$$

Once a correspondence line is established, and pixels have been defined, it remains fixed. During the pose variation in 6DoF, the projected difference Δc_i^+ from the unmoved center $\boldsymbol{c}_i$ to the variated model point ${}_{\mathrm{C}}\boldsymbol{X}_i^+$ is calculated as

$$\Delta c_i^+ = \boldsymbol{n}_i^\top \big(\boldsymbol{\pi}({}_{\mathrm{C}}\boldsymbol{X}_i^+) - \boldsymbol{c}_i \big). \tag{6}$$

A visualization of a correspondence line with Δc_i^+ is shown in Fig. 2.

We now adopt the segmentation model of [25] and use color histograms to estimate the probability distributions for the foreground and background in the RGB color space. For each pixel on the correspondence line and associated color $\boldsymbol{y}_i(r) = \boldsymbol{I}(\boldsymbol{x}_{\mathrm{cl}i}(r))$, we use those distributions to calculate pixel-wise posteriors

$$p_{ji}(r) = \frac{p(\boldsymbol{y}_i(r)|m_j)}{p(\boldsymbol{y}_i(r)|m_{\mathrm{f}}) + p(\boldsymbol{y}_i(r)|m_{\mathrm{b}})}, \quad j \in \{\mathrm{f}, \mathrm{b}\}, \tag{7}$$

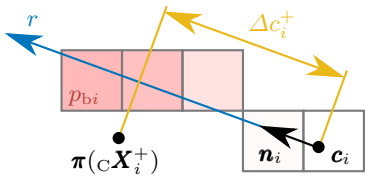

Fig. 2. Correspondence line defined by a center $\boldsymbol{c}_i$ and a normal vector $\boldsymbol{n}_i$, with selected pixels and the projected difference Δc_i^+ from $\boldsymbol{c}_i$ to the variated point ${}_{\mathrm{C}}\boldsymbol{X}_i^+$. The color intensity in red indicates the magnitude of the pixel-wise posterior $p_{\mathrm{b}i}$ for each pixel.

with m_{f} and m_{b} the foreground and background model, respectively. Based on *PWP3D* [17], we finally develop a probabilistic formulation. It describes how well a pose dependent contour model, which uses smoothed step functions to model uncertainty in the contour location, explains the calculated pixel-wise posteriors

$$p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}) \propto \prod_{r\in\mathcal{R}_i} \Big(h_{\mathrm{f}}(r-\Delta c_i^+)p_{\mathrm{f}i}(r) + h_{\mathrm{b}}(r-\Delta c_i^+)p_{\mathrm{b}i}(r)\Big), \tag{8}$$

with $\boldsymbol{\mathcal{D}}_i$ the data specific to a single correspondence line, $\mathcal{R}_i$ a set of distances r from the line center to pixel centers that ensures that every pixel along the line appears exactly once, and h_{f} and h_{b} the smoothed step functions for foreground and background, which will be specified in Section 4.1. Finally, assuming n_{cl} independent correspondence lines, the full likelihood can be calculated as

$$p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) \propto \prod_{i=1}^{n_{\mathrm{cl}}} p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}). \tag{9}$$

3.3 Discrete Scale-Space Formulation

In order to improve computational efficiency, we develop a discrete scale-space formulation that allows the combination of multiple pixels into segments and the precomputation of h_{f} and h_{b} (see Fig. 3). Real-numbered values, like the distances in $\mathcal{R}_i$, which depend on both the angle and the center of the correspondence line, are projected into a discrete space that is scaled according to

$$r_{\mathrm{s}} = (r-\Delta r_i)\frac{\bar{n}_i}{s}, \quad \Delta c_{\mathrm{s}i}^+ = (\Delta c_i^+ - \Delta r_i)\frac{\bar{n}_i}{s}, \tag{10}$$

with $s \in \mathbb{N}^+$ the scale that describes the number of pixels combined into a segment, $\bar{n}_i = \max(|n_{xi}|, |n_{yi}|)$ the normal component that projects a correspondence line to the closest horizontal or vertical coordinate, and $\Delta r_i \in \mathbb{R}$ a distance from the correspondence line center $\boldsymbol{c}_i$ to a defined segment location. Notice that while, in theory, arbitrary values can be chosen for Δr_i, it is typically chosen to point either to the closest center or border of a segment.

Based on Eq. (8), we write the likelihood function in scale-space as follows

$$p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}) \propto \prod_{r_{\mathrm{s}}\in\mathcal{R}_{\mathrm{s}}} \Big(h_{\mathrm{f}}(r_{\mathrm{s}}-\Delta\tilde{c}_{\mathrm{s}i})p_{\mathrm{sf}i}(r_{\mathrm{s}}) + h_{\mathrm{b}}(r_{\mathrm{s}}-\Delta\tilde{c}_{\mathrm{s}i})p_{\mathrm{sb}i}(r_{\mathrm{s}})\Big), \tag{11}$$

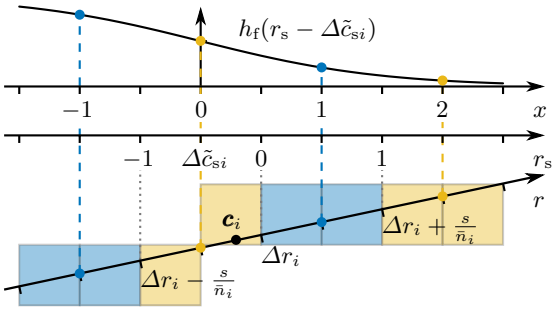

Fig. 3. Example of the relation between the unscaled space r along the correspondence line, the discretized scale-space r_s, and the space x of the smoothed step functions h_f and h_b. Neighboring pixels combined into segments are visualized by the same color. Δr_i was chosen such that the center $r_\mathrm{s} = 0$ lies on a defined location on the border between two segments. Blue and yellow dots indicate precalculated values of h_f and segment centers. Dashed vertical lines connecting those dots highlight that $\Delta\tilde{c}_{\mathrm{s}i}$ has to be chosen such that precalculated values are aligned with segment centers.

with $\mathcal{R}_\mathrm{s}$ a set of distances to segment centers that ensures that every segment along the correspondence line appears exactly once, $\Delta\tilde{c}_{\mathrm{s}i}$ a discretized projected difference value that ensures the alignment with precomputed values of h_f and h_b, and $p_{\mathrm{sf}i}$ and $p_{\mathrm{sb}i}$ segment-wise posteriors. Assuming pixel-wise independence, we define segment-wise posteriors similar to pixel-wise posteriors as

$$p_{\mathrm{s}ji}(r_\mathrm{s}) = \frac{\prod\limits_{r \in \mathcal{S}(r_\mathrm{s})} p(\boldsymbol{y}_i(r)|m_j)}{\prod\limits_{r \in \mathcal{S}(r_\mathrm{s})} p(\boldsymbol{y}_i(r)|m_\mathrm{f}) + \prod\limits_{r \in \mathcal{S}(r_\mathrm{s})} p(\boldsymbol{y}_i(r)|m_\mathrm{b})}, \quad j \in \{\mathrm{f}, \mathrm{b}\}, \tag{12}$$

where $\mathcal{S}$ is a set-valued function that maps r_s to a set of values r that describe the s closest pixel centers of a segment. Note that pixel-wise independence is a well-established approximation that avoids ill-defined assumptions about spatial distribution regularities and is close enough to reality to ensure good results.

Due to a limited number of precalculated values for h_f and h_b, the likelihood function in Eq. (11) can only be evaluated at discrete values $\Delta\tilde{c}_{\mathrm{s}i}$. To approximate the likelihood for arbitrary $\boldsymbol{\theta}$ and corresponding $\Delta c^+_{\mathrm{s}i}$, the upper and lower neighboring discretized values $\Delta\tilde{c}^+_{\mathrm{s}i}$ and $\Delta\tilde{c}^-_{\mathrm{s}i}$ are used to linearly interpolate

$$p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}) \propto (\Delta\tilde{c}^+_{\mathrm{s}i} - \Delta c^+_{\mathrm{s}i}) p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}^-_{\mathrm{s}i}) + (\Delta c^+_{\mathrm{s}i} - \Delta\tilde{c}^-_{\mathrm{s}i}) p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}^+_{\mathrm{s}i}). \tag{13}$$

4 Optimization

In the following section, a novel mathematical proof is derived to find functions h_f and h_b that ensure that the likelihood follows a Gaussian distribution. This is followed by a description of the regularized Newton method that is used to optimize the likelihood. Finally, we define the required gradient and Hessian and discuss how to find robust approximations in the presence of noise.

4.1 Gaussian Equivalence

In general, Newton optimization yields particularly good results for Gaussian distributions. In the case of a perfect normal distribution, where the application of the logarithm leads to a quadratic function, the algorithm converges in a single step. To improve convergence, we thus try to find smoothed step functions h_f and h_b that ensure that the developed likelihood function follows a normal distribution. Please note that only the main concepts of the proof are covered in this section. A detailed version is provided in the supplementary material.

To make the problem tractable, a contour at the correspondence line center and perfect segmentation are assumed. This results in simple unit step functions for pixel-wise posteriors. Also, the smoothed step functions h_f and h_b are restricted to sum to one and to be symmetric. Consequently, we define

$$h_\mathrm{f}(x) = 0.5 - f(x), \quad h_\mathrm{b}(x) = 0.5 + f(x), \tag{14}$$

where $f(x)$ is an odd function that lies within the interval $[-0.5, 0.5]$ and that fulfills $\lim_{x\to\infty} f(x) = 0.5$ and $\lim_{x\to-\infty} f(x) = -0.5$. Finally, we assume infinitesimally small pixels to write the likelihood from Eq. (8) in continuous form

$$p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}) \propto \exp\left(\int_{r=-\infty}^{\infty} \ln\Big(h_\mathrm{f}(r-\Delta c_i^+)p_{\mathrm{f}i}(r) + h_\mathrm{b}(r-\Delta c_i^+)p_{\mathrm{b}i}(r)\Big)dr\right). \tag{15}$$

Using the assumption of perfect pixel-wise posteriors, one is able to simplify

$$p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}) \propto \exp\left(\int_{x=-\infty}^{-\Delta c_i^+} \ln\big(h_\mathrm{f}(x)\big)dx + \int_{x=-\Delta c_i^+}^{\infty} \ln\big(h_\mathrm{b}(x)\big)dx\right). \tag{16}$$

To eliminate constant scaling terms and the integral, we apply the logarithm and use Leibniz's rule for differentiation under the integral to calculate the first-order derivative with respect to Δc_i^+. This is then factorized using Eq. (14)

$$\frac{\partial \ln\big(p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta})\big)}{\partial \Delta c_i^+} = -\ln\big(h_\mathrm{f}(-\Delta c_i^+)\big) + \ln\big(h_\mathrm{b}(-\Delta c_i^+)\big) \tag{17}$$

$$= 2\tanh^{-1}\big(2f(-\Delta c_i^+)\big). \tag{18}$$

Finally, equality is enforced for the first-order derivative of the logarithm of both the likelihood in Eq. (18) and the normal distribution $\mathcal{N}(\Delta c_i^+|0, s_\mathrm{h})$. Knowing that for the normal distribution one obtains $-s_\mathrm{h}^{-1}\Delta c_i^+$, we solve for f and use Eq. (14) to find the following expressions for the smoothed step functions

$$h_\mathrm{f}(x) = \frac{1}{2} - \frac{1}{2}\tanh\Big(\frac{x}{2s_\mathrm{h}}\Big), \quad h_\mathrm{b}(x) = \frac{1}{2} + \frac{1}{2}\tanh\Big(\frac{x}{2s_\mathrm{h}}\Big), \tag{19}$$

where s_h is at the same time the slope parameter for the smoothed step functions and the variance of the designed likelihood function.

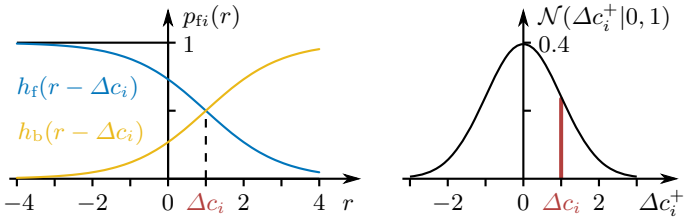

Fig. 4. Example of the relation between our smoothed step functions h_{f} and h_{b} and the normal distribution of Δc_i^+ for $s_h = 1$. The graph on the left shows perfect pixel-wise posterior probabilities $p_{\mathrm{f}i}$ for the foreground and smoothed step functions for a specific projected difference Δc_i. The graph on the right displays the corresponding normal distribution for all values of Δc_i^+. The probability value for Δc_i is depicted in red.

Since equality is enforced for the first-order derivatives of the logarithms, the original functions can only differ by a constant scaling factor, and we can write

$$p(\boldsymbol{\mathcal{D}}_i|\boldsymbol{\theta}) \propto \mathcal{N}(\Delta c_i^+|0, s_{\mathrm{h}}). \tag{20}$$

An example of the relation between the smoothed step functions and the resulting normal distribution is shown in Fig. 4. In summary, the proof shows that with the derived smoothed step functions, we attain a probabilistic model that follows a Gaussian distribution. Although assumptions, such as perfect pixel-wise posteriors and infinitesimally small pixels, are not an exact description of reality, experiments demonstrate that, with the constructed probabilistic model, we achieve excellent convergence for the used regularized Newton optimization.

4.2 Regularized Newton Method

To maximize the likelihood, we estimate the variation vector and iteratively update the pose. For a single iteration, the variation vector is calculated using Newton optimization with Tikhonov regularization as follows

$$\hat{\boldsymbol{\theta}} = \left(-\boldsymbol{H} + \begin{bmatrix} \lambda_{\mathrm{r}} \boldsymbol{I}_3 & \boldsymbol{0} \\ \boldsymbol{0} & \lambda_{\mathrm{t}} \boldsymbol{I}_3 \end{bmatrix} \right)^{-1} \boldsymbol{g}, \tag{21}$$

where $\boldsymbol{g}$ and $\boldsymbol{H}$ are the gradient vector and the Hessian matrix defined as

$$\boldsymbol{g}^\top = \frac{\partial}{\partial \boldsymbol{\theta}} \ln\left(p(\boldsymbol{\mathcal{D}} \mid \boldsymbol{\theta})\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{0}}, \quad \boldsymbol{H} = \frac{\partial^2}{\partial \boldsymbol{\theta}^2} \ln\left(p(\boldsymbol{\mathcal{D}} \mid \boldsymbol{\theta})\right)\Big|_{\boldsymbol{\theta}=\boldsymbol{0}}, \tag{22}$$

and λ_{r} and λ_{t} the regularization parameters for rotation and translation, respectively. Using the log-likelihood has the advantage that scaling terms vanish and products turn into summations. Finally, the pose can be updated according to

$${}_{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}} = {}_{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}} \begin{bmatrix} \exp([\hat{\boldsymbol{\theta}}_{\mathrm{r}}]_\times) & \hat{\boldsymbol{\theta}}_{\mathrm{t}} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{23}$$

Notice that due to the exponential map, no orthonormalization is necessary. By iteratively repeating the process, we are now able to estimate the optimal pose.

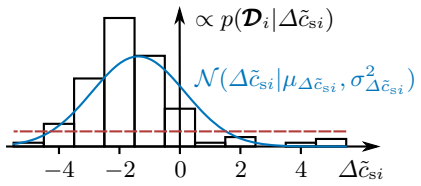

Fig. 5. Example showing normalized values of a noisy discrete likelihood $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i})$ and the normal distribution $\mathcal{N}(\Delta\tilde{c}_{\mathrm{s}i}|\mu_{\Delta\tilde{c}_{\mathrm{s}i}}, \sigma^2_{\Delta\tilde{c}_{\mathrm{s}i}})$ that approximates that likelihood. The red line indicates a threshold for the probability values. To avoid errors from image noise and invalid pixel-wise posteriors, for values below this threshold, the normal distribution is used in the calculation of partial derivatives of the log-likelihood.

4.3 Gradient and Hessian Approximation

Using the chain rule, the gradient vector and Hessian matrix can be defined as

$$\boldsymbol{g}^\top = \sum_{i=1}^{n_{\mathrm{cl}}} \frac{\partial \ln\left(p(\boldsymbol{\mathcal{D}}_i \mid \boldsymbol{\theta})\right)}{\partial \Delta c_{\mathrm{s}i}^+} \frac{\partial \Delta c_{\mathrm{s}i}^+}{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+} \frac{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\mathbf{0}}, \tag{24}$$

$$\boldsymbol{H} \approx \sum_{i=1}^{n_{\mathrm{cl}}} \frac{\partial^2 \ln\left(p(\boldsymbol{\mathcal{D}}_i \mid \boldsymbol{\theta})\right)}{\partial {\Delta c_{\mathrm{s}i}^+}^2} \left(\frac{\partial \Delta c_{\mathrm{s}i}^+}{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+} \frac{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+}{\partial \boldsymbol{\theta}}\right)^\top \left(\frac{\partial \Delta c_{\mathrm{s}i}^+}{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+} \frac{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+}{\partial \boldsymbol{\theta}}\right)\bigg|_{\boldsymbol{\theta}=\mathbf{0}}. \tag{25}$$

Notice that since the first-order partial derivative of the log-likelihood becomes zero when the optimization reaches the maximum, second-order partial derivatives for $\Delta c_{\mathrm{s}i}^+$ and ${}_{\mathrm{C}}\boldsymbol{X}_i^+$ are neglected. Omitting the plus operator for variables evaluated at $\boldsymbol{\theta} = \mathbf{0}$ and using Eq. (4), (6), and (10), we calculate

$$\frac{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\mathbf{0}} = {}_{\mathrm{C}}\boldsymbol{R}_{\mathrm{M}} \begin{bmatrix} -[{}_{\mathrm{M}}\boldsymbol{X}_i]_\times & \boldsymbol{I} \end{bmatrix}, \tag{26}$$

$$\frac{\partial \Delta c_{\mathrm{s}i}^+}{\partial {}_{\mathrm{C}}\boldsymbol{X}_i^+}\bigg|_{\boldsymbol{\theta}=\mathbf{0}} = \frac{\bar{n}_i}{s} \frac{1}{{}_{\mathrm{C}}Z_i^2} \begin{bmatrix} n_{xi} f_{x\mathrm{C}} Z_i & n_{yi} f_{y\mathrm{C}} Z_i & -n_{xi} f_{x\mathrm{C}} X_i - n_{yi} f_{y\mathrm{C}} Y_i \end{bmatrix}. \tag{27}$$

For the first-order partial derivative of the log-likelihood, two cases are distinguished. If the normalized values of $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^+)$ and $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^-)$ are above a defined threshold, we assume that they are reliable and, based on Eq. (13), write

$$\frac{\partial \ln\left(p(\boldsymbol{\mathcal{D}}_i \mid \boldsymbol{\theta})\right)}{\partial \Delta c_{\mathrm{s}i}^+}\bigg|_{\boldsymbol{\theta}=\mathbf{0}} \approx \frac{p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^+) - p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^-)}{(\Delta\tilde{c}_{\mathrm{s}i}^+ - \Delta c_{\mathrm{s}i}) p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^-) + (\Delta c_{\mathrm{s}i} - \Delta\tilde{c}_{\mathrm{s}i}^-) p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^+)}. \tag{28}$$

If one value is below the threshold, the knowledge that $p(\boldsymbol{\mathcal{D}}_i \mid \Delta\tilde{c}_{\mathrm{s}i})$ follows a normal distribution is used. The distribution can be estimated using the mean $\mu_{\Delta\tilde{c}_{\mathrm{s}i}}$ and the standard deviation $\sigma_{\Delta\tilde{c}_{\mathrm{s}i}}$ calculated from $p(\boldsymbol{\mathcal{D}}_i \mid \Delta\tilde{c}_{\mathrm{s}i})$ (see Fig. 5). Consequently, the first-order partial derivative can be approximated by

$$\frac{\partial \ln\left(p(\boldsymbol{\mathcal{D}}_i \mid \boldsymbol{\theta})\right)}{\partial \Delta c_{\mathrm{s}i}^+}\bigg|_{\boldsymbol{\theta}=\mathbf{0}} \approx -\frac{1}{{\sigma_{\Delta\tilde{c}_{\mathrm{s}i}}}^2} \left(\Delta c_{\mathrm{s}i} - \mu_{\Delta\tilde{c}_{\mathrm{s}i}}\right). \tag{29}$$

For the second-order partial derivative, we always use the approximation

$$\left.\frac{\partial^2 \ln\left(p(\boldsymbol{\mathcal{D}}_i \mid \boldsymbol{\theta})\right)}{\partial {\Delta c_{\mathrm{s}i}^{+}}^2}\right|_{\boldsymbol{\theta}=\mathbf{0}} \approx -\frac{1}{{\sigma_{\Delta \tilde{c}_{\mathrm{s}i}}}^2}. \tag{30}$$

The approximations ensure that our partial derivatives maintain a global view of the distribution and that they are robust in the presence of image noise and invalid pixel-wise posteriors. Also, notice that distinguishing between the two cases allows the first-order derivative to direct the optimization toward the actual maximum while remaining stable for small, inaccurate probability values.

5 Implementation

For the proposed method, 3D points and normal vectors from the contour of the model are required. To ensure computational efficiency, we take an approach similar to [16] and precompute template views that store the required information. The 3D model of an object is thereby rendered from 2562 different viewpoints that are placed on the vertices of a geodesic grid with a distance of 0.8 m to the object center. For each rendering, we randomly sample $n_{\mathrm{cl}} = 200$ points from the object contour. Based on those coordinates, 3D model points and associated normal vectors are reconstructed. In addition, we compute continuous distances for the foreground and background, where the corresponding regions are not interrupted by the other. Starting from a coordinate, those distances are measured along the normal vector. For each view, the data is then stored together with a direction vector ${}_{\mathrm{M}}\boldsymbol{v}_i \in \mathbb{R}^3$ that points from the camera to the model center.

Each tracking step starts either from the previous pose or an initial pose provided by a 3D object detection pipeline. Based on this pose, we first retrieve the template view i_{t} that is closest to the direction in which the object is located

$$i_{\mathrm{t}} = \underset{i \in \{1,\dots,2562\}}{\arg\max} \left({}_{\mathrm{M}}\boldsymbol{v}_i^{\top}\, {}_{\mathrm{M}}\boldsymbol{R}_{\mathrm{CC}} \boldsymbol{t}_{\mathrm{M}}\right). \tag{31}$$

Model points and normal vectors from the template view are then projected into the image plane to define correspondence lines. For the probabilistic model, a continuous foreground and background on each side of the contour are required. We thus scale precomputed continuous distances according to the current camera distance and scale s and reject correspondence lines with a continuous distance below 8 segments. After that, each distribution $p(\boldsymbol{\mathcal{D}}_i \mid \Delta\tilde{c}_{\mathrm{s}i})$ is calculated for 11 discrete values $\Delta\tilde{c}_{\mathrm{s}i} \in \{-5, -4, \dots, 5\}$. The distance Δr_i is thereby chosen to be minimal such that the distribution center at $\Delta c_{\mathrm{s}i}^{+} = 0$ is closest to the correspondence line center $\boldsymbol{c}_i$. For the smoothed step functions, 10 precomputed values corresponding to $x \in \{-4.5, -3.5, \dots, 4.5\}$ are used. Both h_{f} and h_{b} are defined using a slope of $s_h = 1.3$. Notice that with this slope, the smallest considered value is $h_{\mathrm{f}}(4.5) = 0.03$. Values smaller than that are neglected since, in general, they do not contribute significantly to the overall distribution.

Following the calculation of the distributions, 2 iterations of the regularized Newton method are performed. Both the gradient and Hessian are recomputed

for each iteration. As threshold for the normalized values of $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^{+})$ and $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{\mathrm{s}i}^{-})$, we use 0.01. Also, we check if the standard deviation $\sigma_{\Delta\tilde{c}_{\mathrm{s}i}}$ is above the theoretical minimum $\sqrt{s}_{\mathrm{h}}$. If it is smaller, due to the limited number of values considered in the distribution, it is set to $\sqrt{s}_{\mathrm{h}}$. Similarly, the magnitude of the first-order derivative of the log-likelihood in Eq. (28) is limited by the theoretical maximum $10/s_{\mathrm{h}}$. For the regularization, we use $\lambda_{\mathrm{r}} = 5000$ and $\lambda_{\mathrm{t}} = 500000$.

To find the final pose, the process is repeated 7 times, starting each time with the retrieval of a new template view. The first and second iterations use a scale of $s = 5$ and $s = 2$, respectively. For all other iterations, the scale is set to $s = 1$. Examples of different scales are shown in Fig. 1. The specified scales have the effect that in the first iteration a large area with low resolution is considered while short lines with high resolution are used in later iterations.

Once the final pose is estimated, the color histograms, that correspond to the probability distributions $p(\boldsymbol{y}|m_{\mathrm{f}})$ and $p(\boldsymbol{y}|m_{\mathrm{b}})$, are updated. The RGB color space is thereby discretized with 32 equidistant bins in each dimension, leading to a total of 32768 values. For the calculation, correspondence lines are established. Pixels along the line are then assigned to either the foreground or background, depending on which side of the center they are. Starting with an offset of 2 pixels, the first 10 pixels are used in both directions. Based on [25], we allow for online adaptation of the foreground and background statistical model as follows

$$p_t(\boldsymbol{y}|m_i) = \alpha_i p_t(\boldsymbol{y}|m_i) + (1 - \alpha_i) p_{t-1}(\boldsymbol{y}|m_i), \quad i \in \{\mathrm{f}, \mathrm{b}\}, \tag{32}$$

where $\alpha_{\mathrm{f}} = 0.1$ and $\alpha_{\mathrm{b}} = 0.2$ are the learning rates for the foreground and background, respectively. The update ensures that effects such as changes in illumination or in the background are considered. For the initialization, we directly use the computed histograms instead of blending them with previous values.

To take into account known occlusions, a renderer that generates occlusion masks for the rejection of correspondence lines is implemented. It only uses a fourth of the camera resolution to ensure computational efficiency. Also, the silhouette of each object is dilated by 1 pixel in the mask domain to consider uncertainty in the object pose. Finally, to reject occluded correspondence lines, the algorithm checks if the center $\boldsymbol{c}_i$ falls on the mask of another object.

6 Evaluation

In the following, we first introduce the *Region-Based Object Tracking (RBOT)* dataset [19] and describe the conducted experiments. Our results are then compared to the state of the art in region-based tracking [33, 19, 27, 20]. Finally, based on an ablation study in the supplementary material, we discuss the essential aspects of our method. To ensure reproducibility, we provide the source code on *GitHub*[1]. Also, for visualization, we added a video in the supplementary material that illustrates the *RBOT* dataset, our approach, and the experiments. In addition, multiple real-world sequences are available on our project site[2].

[1] https://github.com/DLR-RM/RBGT
[2] https://rmc.dlr.de/rm/staff/manuel.stoiber/accv2020

Fig. 6. Overview of all objects in the *RBOT* dataset [19]. Objects from the *LINEMOD* dataset [34] and *Rigid Pose* dataset [35] are marked with ⋆ and ◇, respectively.

6.1 Experiments

The *RBOT* dataset consists of a collection of eighteen objects (see Fig. 6): twelve from the *LINEMOD* dataset [34], five from the *Rigid Pose* dataset [35], and one from its creators. For each object, four sequences exist. Each sequence consists of 1001 semi-synthetic monocular images with a resolution of 640×512 pixels. Objects were rendered into real-world images, recorded from a hand-held camera that moves around a cluttered desk scene (see Fig. 1). To simulate motion blur, a 3×3 Gaussian kernel was applied. The first sequence features a *regular* version with a static point light source. The second, *dynamic light*, variant simulates simultaneous motion of the camera and object. For the third sequence, Gaussian *noise* and dynamic lighting were added. Finally, the fourth sequence features an additional squirrel object that leads to *occlusion* by orbiting around the first object. It also includes dynamic lighting. In all sequences, objects move along the same trajectories. The ground-truth pose is given by the rotation matrix ${}_\mathrm{C}\boldsymbol{R}_{\mathrm{M_{gt}}}(t_k)$ and the translation vector ${}_\mathrm{C}\boldsymbol{t}_{\mathrm{M_{gt}}}(t_k)$ for $k \in \{0, \dots, 1000\}$.

The evaluation is conducted on a computer with an *Intel Xeon E5-1630 v4* CPU and a *Nvidia Quadro P600* GPU. All experiments are performed exactly as in [19], with the translational and rotational error calculated as

$$e_\mathrm{t}(t_k) = \| {}_\mathrm{C}\boldsymbol{t}_\mathrm{M}(t_k) - {}_\mathrm{C}\boldsymbol{t}_{\mathrm{M_{gt}}}(t_k) \|_2, \tag{33}$$

$$e_\mathrm{r}(t_k) = \cos^{-1}\Big(0.5\big(\operatorname{trace}({}_\mathrm{C}\boldsymbol{R}_\mathrm{M}(t_k)^\top {}_\mathrm{C}\boldsymbol{R}_{\mathrm{M_{gt}}}(t_k)) - 1\big)\Big). \tag{34}$$

Starting from an initialization with the ground-truth pose at t_0, the tracker runs automatically until either the recorded sequence ends or tracking was unsuccessful. A pose is considered successful if both $e_\mathrm{t}(t_k) < 5\,\mathrm{cm}$ and $e_\mathrm{r}(t_k) < 5°$. Otherwise, the tracker is re-initialized with the ground-truth pose at t_k. For the *occlusion* sequence, we differentiate between two variants. In the first, only the main object is tracked, and occlusions are considered as *unmodeled*. In the second, both objects are tracked, and occlusions are *modeled* using occlusion masks. Notice that, while the occluding object is re-initialized if tracking is unsuccessful, it is not considered in the evaluation of the tracking success.

6.2 Results

In Table 1, we compare the success rates of state-of-the-art methods to our proposed approach. Note that [27] and [20] do not support the *modeled occlusion*

Table 1. Tracking success rates of [33], [19], [27], [20], and our method, featuring all variants of the *RBOT* dataset. The best values are highlighted in bold numbers.

Method	Ape	Soda	Vise	Soup	Camera	Can	Cat	Clown	Cube	Driller	Duck	Egg Box	Glue	Iron	Candy	Lamp	Phone	Squirrel	**Average**
Regular																			
[33]	62.1	30.5	95.8	66.2	61.6	81.7	96.7	89.1	44.1	87.7	74.9	50.9	20.2	68.4	20.0	92.3	64.9	98.5	67.0
[19]	85.0	39.0	**98.9**	82.4	79.7	87.6	95.9	93.3	78.1	**93.0**	86.8	74.6	38.9	81.0	46.8	**97.5**	80.7	99.4	79.9
[27]	82.6	40.1	92.6	85.0	82.8	87.2	98.0	92.9	81.3	84.5	83.3	76.2	56.1	84.6	57.6	90.5	82.6	95.6	80.8
[20]	88.8	41.3	94.0	85.9	86.9	89.0	98.5	93.7	83.1	87.3	86.2	78.5	58.6	86.3	57.9	91.7	85.0	96.2	82.7
Us	**96.4**	**53.2**	98.8	**93.9**	**93.0**	**92.7**	**99.7**	**97.1**	**92.5**	92.5	**93.7**	**88.5**	**70.0**	**92.1**	**78.8**	95.5	**92.5**	**99.6**	**90.0**
Dynamic Light																			
[33]	61.7	32.0	94.2	66.3	68.0	84.1	96.6	85.8	45.7	88.7	74.1	56.9	29.9	49.1	20.7	91.5	63.0	98.5	67.0
[19]	84.9	42.0	99.0	81.3	84.3	88.9	95.6	92.5	77.5	**94.6**	86.4	77.3	52.9	77.9	47.9	**96.9**	81.7	99.3	81.2
[27]	81.8	39.7	91.5	85.1	82.6	87.1	98.1	90.7	79.7	87.4	81.6	73.1	51.7	75.9	53.4	88.8	78.6	95.6	79.0
[20]	89.7	40.2	92.7	86.5	86.6	89.2	98.3	93.9	81.8	88.4	83.9	76.8	55.3	79.3	54.7	88.7	81.0	95.8	81.3
Us	**96.5**	**54.6**	**99.1**	**93.9**	**93.1**	**94.7**	**99.5**	**97.0**	**93.0**	93.4	**93.3**	**92.6**	**74.9**	**91.0**	**79.2**	95.6	**89.8**	**99.5**	**90.6**
Noise																			
[33]	55.9	35.3	75.4	67.4	27.8	10.2	94.3	33.4	8.6	50.9	76.3	2.3	2.2	18.2	11.4	36.6	31.3	93.5	40.6
[19]	77.5	44.5	**91.5**	82.9	51.7	38.4	95.1	69.2	24.4	64.3	88.5	11.2	2.9	46.7	32.7	57.3	44.1	96.6	56.6
[27]	80.5	35.0	80.9	85.5	58.4	53.5	96.7	65.9	38.2	71.8	85.8	29.7	17.0	59.3	34.8	61.1	60.8	93.6	61.6
[20]	79.3	35.2	82.6	86.2	65.1	56.9	96.9	67.0	37.5	75.2	85.4	35.2	18.9	63.7	35.4	**64.6**	66.3	93.2	63.6
Us	**91.9**	**53.3**	90.2	**92.6**	**67.9**	**59.3**	**98.4**	**80.6**	**43.5**	**78.1**	**92.5**	**44.0**	**31.3**	**72.3**	**62.0**	59.9	**71.7**	**98.3**	**71.5**
Unmodeled Occlusion																			
[33]	55.2	29.9	82.4	56.9	55.7	72.2	87.9	75.7	39.6	78.7	68.1	47.1	26.2	35.6	16.6	78.6	50.3	77.6	57.5
[19]	80.0	42.7	91.8	73.5	76.1	81.7	89.8	82.6	68.7	86.7	80.5	67.0	46.6	64.0	43.6	88.8	68.6	86.2	73.3
[27]	77.7	37.3	87.1	78.7	74.6	81.0	93.8	84.3	73.2	83.7	77.0	66.4	48.6	70.8	49.6	85.0	73.8	90.6	74.1
[20]	83.9	38.1	92.4	81.5	81.3	85.5	**97.5**	88.9	76.1	87.5	81.7	72.7	52.5	77.2	53.9	88.5	79.3	92.5	78.4
Us	**90.8**	**51.7**	**95.9**	**88.5**	**88.0**	**90.5**	96.9	**91.6**	**87.1**	**90.3**	**86.4**	**85.6**	**65.8**	**87.0**	**72.7**	**91.2**	**84.0**	**97.0**	**85.6**
Modeled Occlusion																			
[33]	60.3	31.0	94.3	64.5	67.0	81.6	92.5	81.4	43.2	89.3	72.7	51.6	28.8	53.5	19.1	89.3	62.2	96.7	65.5
[19]	82.0	42.0	95.7	81.1	78.7	83.4	92.8	87.9	74.3	91.7	84.8	71.0	49.1	73.0	46.3	90.9	76.2	96.9	77.7
Us	**95.0**	**53.8**	**97.8**	**92.4**	**90.6**	**93.5**	**99.1**	**96.3**	**91.5**	**92.6**	**90.9**	**91.3**	**70.5**	**91.8**	**77.2**	**93.7**	**87.0**	**99.0**	**89.1**

scenario. The results show that our approach outperforms the competition in almost all cases by a large margin. Only for very few cases, [19] or [20] produce better results. With respect to the average success rates, our method consistently achieves about seven percentage points more than [20], which previously performed best on the *RBOT* dataset. The superior tracking success is especially interesting since, in contrast to the competition, we do not use advanced segmentation models. Assuming that localized appearance models provide better segmentation, this represents a major disadvantage.

With respect to the remaining failure cases, we notice that there are mainly two causes. One challenge are local ambiguities, where the object silhouette is very similar in the vicinity of a particular pose. Naturally, in such cases, there is not enough information to converge to the correct pose. Another problem arises if large regions in the background contain colors that are also present in the object. Depending on how well the pose is constrained, this might perturb

the final result. Together with the requirements that the 3D model has to be known and that frame-to-frame motion should not exceed the area covered by correspondence lines, these are the natural limitations of our algorithm.

For the average runtime, we measure 1.04 ms. The only exception is the *modeled occlusion* scenario, which features two objects and requires to render occlusion masks. For this scenario, we obtain an average execution time of 7.41 ms. In comparison, [33], [19], [27], and [20] report average runtimes of $12.5 \sim 33.3$ ms, $15.5 \sim 21.8$ ms, 47.0 ms, and 41.2 ms, respectively. While experiments were performed on different computers, the significantly lower execution time demonstrates our method's computational efficiency. Also, our algorithm only utilizes a single CPU core and, except for the *modeled occlusion* variant, does not rely on a GPU. In contrast, the competition always requires a GPU. We thus conclude that, with less computational resources, our method is approximately one order of magnitude faster while achieving significantly better tracking success.

6.3 Essential Aspects

Based on an ablation study that is presented in the supplementary material, in the following, we want to highlight essential aspects that contribute to our excellent results. Regarding computational efficiency, the biggest performance gain is associated with our sparse formulation that considers fewer pixels. In addition, we simply measure the projected distance along correspondence lines instead of calculating a two-dimensional signed distance function over the entire image. Also, with our discrete scale-space formulation, we are able to both combine multiple pixels and use precomputed values for the smoothed step functions.

Concerning tracking success, one essential aspect is the adopted Tikhonov regularization, which constrains the optimization relative to the previous pose. Also, due to the one-dimensionality of our correspondence lines and the developed discrete scale-space implementation, multiple probability values along lines can be sampled in reasonable time. This allows the calculation of sound derivatives that maintain a global view of the distribution. Together with the derived smoothed step functions, that ensure Gaussian properties, a realistic gradient and Hessian are provided to the regularized Newton optimization.

7 Conclusion

In this work, we presented a novel, sparse approach to region-based 6DoF object tracking. On the *RBOT* dataset, we showed that our algorithm outperforms state-of-the-art region-based methods by a considerable margin, both with respect to tracking success and computational efficiency. Because of its general formulation, it is easy to conceive future methods that extend our approach with other developments in region-based tracking. Potential directions include the implementation of more advanced segmentation models or the incorporation of additional information such as depth or texture. In addition, by proving that our probabilistic model follows a Gaussian distribution, we provided a solid foundation for sound uncertainty estimation based on the Hessian matrix.

References

1. Lepetit, V., Fua, P.: Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. Volume 1. Foundations and Trends in Computer Graphics and Vision (2005)
2. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys **38** (2006) 13
3. Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: IEEE International Conference on Computer Vision. Volume 2. (2005) 1508–1515
4. Wagner, D., Reitmayr, G., Mulloni, A., Drummond, T., Schmalstieg, D.: Real-time detection and tracking for augmented reality on mobile phones. IEEE Transactions on Visualization and Computer Graphics **16** (2010) 355–368
5. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence. Volume 2. (1981) 674–679
6. Baker, S., Matthews, I.: Lucas-Kanade 20 years on: A unifying framework. International Journal of Computer Vision **56** (2004) 221–255
7. Harris, C., Stennett, C.: RAPID - A video rate object tracker. In: Proceedings of the British Machine Vision Conference. (1990) 15.1–15.6
8. Wuest, H., Stricker, D.: Tracking of industrial objects by using CAD models. Journal of Virtual Reality and Broadcasting **4** (2007)
9. Seo, B., Park, H., Park, J., Hinterstoisser, S., Ilic, S.: Optimal local searching for fast and robust textureless 3D object tracking in highly cluttered backgrounds. IEEE Transactions on Visualization and Computer Graphics **20** (2014) 99–110
10. Garon, M., Lalonde, J.F.: Deep 6-DOF tracking. IEEE Transactions on Visualization and Computer Graphics **23** (2017) 2410–2418
11. Crivellaro, A., Rad, M., Verdie, Y., Yi, K.M., Fua, P., Lepetit, V.: Robust 3D object tracking from monocular images using stable parts. IEEE Transactions on Pattern Analysis and Machine Intelligence **40** (2018) 1465–1479
12. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: DeepIM: Deep iterative matching for 6D pose estimation. In: European Conference on Computer Vision. (2018) 695–711
13. Newcombe, R.A., Davison, A.J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., Fitzgibbon, A.: KinectFusion: Real-time dense surface mapping and tracking. In: IEEE International Symposium on Mixed and Augmented Reality. (2011) 127–136
14. Choi, C., Christensen, H.I.: RGB-D object tracking: A particle filter approach on GPU. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. (2013) 1084–1091
15. Krull, A., Michel, F., Brachmann, E., Gumhold, S., Ihrke, S., Rother, C.: 6-DOF model based tracking via object coordinate regression. In: Asian Conference on Computer Vision. (2015) 384–399
16. Tan, D.J., Navab, N., Tombari, F.: Looking beyond the simple scenarios: Combining learners and optimizers in 3D temporal tracking. IEEE Transactions on Visualization and Computer Graphics **23** (2017) 2399–2409
17. Prisacariu, V.A., Reid, I.D.: PWP3D: Real-time segmentation and tracking of 3D objects. International Journal of Computer Vision **98** (2012) 335–354
18. Ren, C.Y., Prisacariu, V.A., Kähler, O., Reid, I.D., Murray, D.W.: Real-time tracking of single and multiple objects from depth-colour imagery using 3D signed distance functions. International Journal of Computer Vision **124** (2017) 80–95

19. Tjaden, H., Schwanecke, U., Schömer, E., Cremers, D.: A region-based Gauss-Newton approach to real-time monocular multiple object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **41** (2018) 1797–1812
20. Zhong, L., Zhao, X., Zhang, Y., Zhang, S., Zhang, L.: Occlusion-aware region-based 3D pose tracking of objects with temporally consistent polar-based local partitioning. IEEE Transactions on Image Processing **29** (2020) 5065–5078
21. Liu, Y., Sun, P., Namiki, A.: Target tracking of moving and rotating object by high-speed monocular active vision. IEEE Sensors Journal **20** (2020) 6727–6744
22. Rosenhahn, B., Brox, T., Weickert, J.: Three-dimensional shape knowledge for joint image segmentation and pose tracking. International Journal of Computer Vision **73** (2007) 243–262
23. Schmaltz, C., Rosenhahn, B., Brox, T., Weickert, J.: Region-based pose tracking with occlusions using 3D models. Machine Vision and Applications **23** (2012) 557–577
24. Dambreville, S., Sandhu, R., Yezzi, A., Tannenbaum, A.: Robust 3D pose estimation and efficient 2D region-based segmentation from a 3D shape prior. In: European Conference on Computer Vision. (2008) 169–182
25. Bibby, C., Reid, I.: Robust real-time visual tracking using pixel-wise posteriors. In: European Conference on Computer Vision. (2008) 831–844
26. Kehl, W., Tombari, F., Ilic, S., Navab, N.: Real-time 3D model tracking in color and depth on a single CPU core. In: IEEE Conference on Computer Vision and Pattern Recognition. (2017) 465–473
27. Zhong, L., Zhang, L.: A robust monocular 3D object tracking method combining statistical and photometric constraints. International Journal of Computer Vision **127** (2019) 973–992
28. Zhong, L., Zhang, Y., Zhao, H., Chang, A., Xiang, W., Zhang, S., Zhang, L.: Seeing through the occluders: Robust monocular 6-DOF object pose tracking via model-guided video object segmentation. IEEE Robotics and Automation Letters (2020) 1–8
29. Prisacariu, V.A., Kähler, O., Murray, D.W., Reid, I.D.: Real-time 3D tracking and reconstruction on mobile phones. IEEE Transactions on Visualization and Computer Graphics **21** (2015) 557–570
30. Zhao, S., Wang, L., Sui, W., Wu, H., Pan, C.: 3d object tracking via boundary constrained region-based model. In: IEEE International Conference on Image Processing. (2014) 486–490
31. Lankton, S., Tannenbaum, A.: Localizing region-based active contours. IEEE Transactions on Image Processing **17** (2008) 2029–2039
32. Hexner, J., Hagege, R.R.: 2D-3D pose estimation of heterogeneous objects using a region based approach. International Journal of Computer Vision **118** (2016) 95–112
33. Tjaden, H., Schwanecke, U., Schomer, E.: Real-time monocular pose estimation of 3D objects using temporally consistent local color histograms. In: IEEE International Conference on Computer Vision. (2017) 124–132
34. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: Asian Conference on Computer Vision. (2013) 548–562
35. Pauwels, K., Rubio, L., Díaz, J., Ros, E.: Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues. In: IEEE Conference on Computer Vision and Pattern Recognition. (2013) 2347–2354