

# Iterative Corresponding Geometry: Fusing Region and Depth for Highly Efficient 3D Tracking of Textureless Objects

Manuel Stoiber<sup>1,2</sup>Martin Sundermeyer<sup>1,2</sup>Rudolph Triebel<sup>1,2</sup><sup>1</sup> German Aerospace Center (DLR)<sup>2</sup> Technical University of Munich (TUM)

{firstname.lastname}@dlr.de

## Abstract

Tracking objects in 3D space and predicting their 6DoF pose is an essential task in computer vision. State-of-the-art approaches often rely on object texture to tackle this problem. However, while they achieve impressive results, many objects do not contain sufficient texture, violating the main underlying assumption. In the following, we thus propose ICG, a novel probabilistic tracker that fuses region and depth information and only requires the object geometry. Our method deploys correspondence lines and points to iteratively refine the pose. We also implement robust occlusion handling to improve performance in real-world settings. Experiments on the YCB-Video, OPT, and Choi datasets demonstrate that, even for textured objects, our approach outperforms the current state of the art with respect to accuracy and robustness. At the same time, ICG shows fast convergence and outstanding efficiency, requiring only 1.3 ms per frame on a single CPU core. Finally, we analyze the influence of individual components and discuss our performance compared to deep learning-based methods. The source code of our tracker is publicly available<sup>1</sup>.

## 1. Introduction

For many applications in robotic manipulation and augmented reality, it is essential to know the six degrees of freedom (6DoF) pose of relevant objects. To provide this information at high frequency, 3D object tracking is used. The goal is to estimate an object’s position and orientation from consecutive image frames given its 3D model. In real-world applications, occlusions, motion blur, background clutter, textureless surfaces, object symmetries, and real-time requirements remain difficult problems. Over the years many approaches have been developed [29, 69]. They can be differentiated by the use of keypoints, edges, direct optimization, deep learning, object regions, and depth images.

While methods based on keypoints [38, 47, 48, 54, 61],

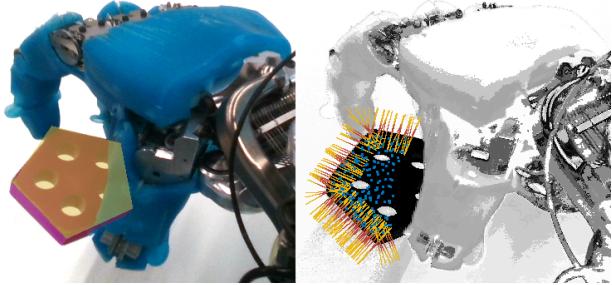


Figure 1. Tracking of a pentagon object for robotic manipulation. The image on the left shows an overlay of the object model for the predicted pose. On the right, probabilities that a pixel belongs to the background are encoded in a grayscale image. Correspondence lines are shown in yellow, with high probabilities indicated in red. Projected correspondence points are illustrated in blue.

edges [12, 17, 20, 52], and direct optimization [1, 13, 34, 53] were very popular in the past, multiple drawbacks exist. Both keypoints and direct optimization are not suitable for textureless objects. Edge-based methods, on the other hand, typically struggle with background clutter and object texture. Further problems emerge from reflections and motion blur, which change the appearance of both texture and edges. To overcome those issues, data-driven techniques that use convolutional neural networks (CNNs) have been proposed [15, 31, 62, 64]. While most of those methods require significant computational resources and a detailed 3D model, they achieve promising results. For the tracking of textureless objects in cluttered environments, region-based techniques have also become very popular [42, 56, 60, 73]. Furthermore, the emergence of consumer depth sensors has enabled additional trackers that do not rely on texture [11, 24, 37, 51, 66]. Finally, while all those methods can be used independently, many approaches demonstrated the benefits of combining different techniques [25, 26, 44, 59].

In the past, it was shown that a combination of region and depth has great potential for the tracking of textureless objects [25, 44]. However, while region-based tech-

<sup>1</sup><https://github.com/DLR-RM/3DObjectTracking>

niques improved greatly with respect to efficiency and quality [55, 56], no recent combined approach exists. In the following, we thus build on current developments and propose *ICG*, a highly efficient method that fuses geometric information from region-based correspondence lines and depth-based correspondence points. An illustration of the used correspondences is shown in Fig. 1. In a detailed evaluation on three different datasets, our method demonstrates state-of-the-art performance compared to both classical and deep learning-based techniques. In addition, given that only few such comparisons were conducted in the past, we are able to gain new insights into the current state of deep learning-based object tracking and pose estimation.

## 2. Related Work

In the following, we provide an overview on region-, depth-, and deep learning-based techniques. Region-based methods typically use color statistics to model the probability that a pixel belongs to the object or to the background. The object pose is then optimized to best explain the segmentation of the image. While early approaches treated segmentation and optimization separately [6, 46, 50], subsequent work [14] combined the two steps. Later, the pixel-wise posterior membership of [3] was used to develop *PWP3D* [42]. Based on this method, multiple combined approaches that incorporate depth information [25, 44], edges [30, 57], inertial measurements [41], or use direct optimization [32, 33, 72] were developed. In addition, it was suggested to localize the probabilistic segmentation model [22, 60, 73]. Different optimization techniques such as particle filters [71], Levenberg Marquardt [41], Gauss Newton [60], or Newton with Tikhonov regularization [55, 56] were also proposed. Finally, starting from the ideas of [25], the efficiency problem of region-based methods was addressed with the development of the sparse tracker *SRT3D* [55, 56].

Depth-based methods try to minimize the distance between the surface of a 3D model and measurements from a depth camera. Often, approaches based on the *Iterative Closest Point (ICP)* framework [2, 9] are used. While many variants exist [40, 49], all algorithms iteratively establish correspondences and minimize a respective error function. For tracking, projective data association [4] and the point-to-plane error metric [9] are very common [25, 37, 39, 59]. Apart from correspondence points and *ICP*, methods that utilize signed distance functions are often used [18, 44, 45, 51]. In addition, approaches that employ particle filters [10, 11, 27, 66] or robust Gaussian filters [24] instead of gradient-based optimization are also very popular.

While deep learning has proven highly successful for 6DoF pose estimation [21, 28, 58, 63, 68], pure tracking methods were only recently proposed. Many approaches are inspired by pose refinement and predict the relative pose between object renderings and subsequent images [31, 35,

64]. In addition, *PoseRBPF* [15] uses a Rao-Blackwellized particle filter on pose-representative latent codes [58] while *6-Pack* [62] tracks anchor-based keypoints.

## 3. Probabilistic Model

In this section, mathematical concepts and the used notation are introduced. This is followed by an explanation of the sparse viewpoint model. Finally, the probability density functions (PDFs) for region and depth are derived.

### 3.1. Preliminaries

In this work, we use  $\mathbf{X} = [X \ Y \ Z]^\top \in \mathbb{R}^3$  and the homogeneous form  $\tilde{\mathbf{X}} = [X \ Y \ Z \ 1]^\top$  to describe 3D model points. Image coordinates  $\mathbf{x} = [x \ y]^\top \in \mathbb{R}^2$  are employed to access color values  $\mathbf{y} = \mathbf{I}_c(\mathbf{x})$  and depth values  $d_Z = I_d(\mathbf{x})$  from the respective color and depth images. With the pinhole camera model, a 3D model point is projected into an undistorted image as follows

$$\mathbf{x} = \pi(\mathbf{X}) = \begin{bmatrix} \frac{X}{Z} f_x + p_x \\ \frac{Y}{Z} f_y + p_y \end{bmatrix}, \quad (1)$$

where  $f_x$  and  $f_y$  are the focal lengths and  $p_x$  and  $p_y$  are the coordinates of the principal point.

To describe the relative pose between two reference frames A and B, the homogeneous matrix  ${}_A\mathbf{T}_B \in \mathbb{SE}(3)$  is used. It transforms 3D model points as follows

$${}_A\tilde{\mathbf{X}} = {}_A\mathbf{T}_B {}_B\tilde{\mathbf{X}} = \begin{bmatrix} {}_A\mathbf{R}_B & {}_A\mathbf{t}_B \\ \mathbf{0} & 1 \end{bmatrix} {}_B\tilde{\mathbf{X}}, \quad (2)$$

with  ${}_A\tilde{\mathbf{X}}$  and  ${}_B\tilde{\mathbf{X}}$  a point written in the coordinate frames A and B. The rotation matrix  ${}_A\mathbf{R}_B \in \mathbb{SO}(3)$  and the translation vector  ${}_A\mathbf{t}_B \in \mathbb{R}^3$  define the transformation from B to A. In this work, M, C, and D will be used to denote the model, the color camera, and the depth camera frames, respectively.

For small variations of the pose in the model reference frame M, we use the following minimal representation

$${}_M\tilde{\mathbf{X}}(\boldsymbol{\theta}) = {}_M\mathbf{T}(\boldsymbol{\theta}) {}_M\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{I} + [\boldsymbol{\theta}_r]_\times & \boldsymbol{\theta}_t \\ \mathbf{0} & 1 \end{bmatrix} {}_M\tilde{\mathbf{X}}, \quad (3)$$

where  $[\boldsymbol{\theta}_r]_\times$  is the skew-symmetric matrix of  $\boldsymbol{\theta}_r$ . The vectors  $\boldsymbol{\theta}_r \in \mathbb{R}^3$  and  $\boldsymbol{\theta}_t \in \mathbb{R}^3$  are the rotational and translational components of the full variation vector  $\boldsymbol{\theta}^\top = [\boldsymbol{\theta}_r^\top \ \boldsymbol{\theta}_t^\top]$ .

### 3.2. Sparse Viewpoint Model

To ensure efficiency and avoid the rendering of the 3D model during tracking, we represent the geometry using a sparse viewpoint model [56]. In the generation process, the object is rendered from a large number of virtual cameras that are placed on the vertices of a geodesic grid all around

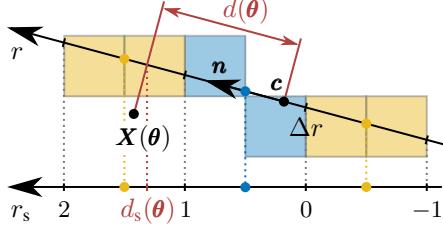


Figure 2. Projection of a correspondence line from image space into scale space. The correspondence line is defined by a center  $\mathbf{c}$  and a normal vector  $\mathbf{n}$ . Pixels along the correspondence line are combined into segments, which are illustrated in blue and yellow. The number of pixels per segment is specified by the scale  $s = 2$ . In the visualization, the pose-dependent 3D contour point  $\mathbf{X}(\theta)$ , which is associated with the correspondence line, is also shown. It is used to compute the distance  $d(\theta)$  from the estimated contour to the center  $\mathbf{c}$ . The transformation from  $r$  into the scale space  $r_s$  is indicated by dotted vertical lines. Note that the space is shifted by  $\Delta r$  and scaled according to  $s$ , as well as the line angle.

the object. Similar to [59], image coordinates are then randomly sampled on the contour and surface of the rendered silhouette. For each coordinate, both the 3D point  ${}_M\mathbf{X}$  and the 3D normal vector  ${}_M\mathbf{N}$  are reconstructed. Together with the orientation that points from the camera to the model center, those vectors are stored for each viewpoint. Given a pose estimate, obtaining the contour and surface representation reduces to a search for the closest orientation vector.

### 3.3. Region Modality

In the following, we adopt the region-based approach of *SRT3D* [55, 56] and modify it to incorporate a user-defined uncertainty. In general, *SRT3D* considers region information sparsely along so-called correspondence lines  $\mathbf{l}$ , which cross the estimated object contour. Similar to the image function  $\mathbf{I}_c$ , correspondence lines map coordinates  $r \in \mathbb{R}$  to color values  $\mathbf{y} = \mathbf{l}(r)$ . Each line is thereby defined by a center  $\mathbf{c} \in \mathbb{R}^2$  and a normal vector  $\mathbf{n} \in \mathbb{R}^2$  in image space. Both vectors are calculated by projecting a 3D contour point  $\mathbf{X}$  and an associated 3D normal vector  $\mathbf{N}$  from the sparse viewpoint model into the image to establish a correspondence. In order to make correspondence lines more efficient, *SRT3D* introduces a scale-space formulation that combines multiple pixel values  $\mathbf{y}$  into segments  $\mathbf{s}$ . The number of pixels is thereby defined by the scale  $s \in \mathbb{N}^+$ . In addition, line coordinates  $r$  are scaled and shifted to make correspondence lines independent of their orientation and sub-pixel location. An illustration is shown in Fig. 2.

Like in most region-based methods, color statistics are used to differentiate between foreground and background. The probabilities  $p(\mathbf{y} \mid m_f)$  and  $p(\mathbf{y} \mid m_b)$  are approximated using normalized color histograms. They describe the likelihood that pixel colors  $\mathbf{y}$  are part of the foreground

or background model  $m_f$  or  $m_b$ . Based on those probabilities, segment-wise posteriors are calculated as

$$p_{si}(r_s) = \frac{\prod_{\mathbf{y} \in \mathbf{s}} p(\mathbf{y} \mid m_i)}{\prod_{\mathbf{y} \in \mathbf{s}} p(\mathbf{y} \mid m_f) + \prod_{\mathbf{y} \in \mathbf{s}} p(\mathbf{y} \mid m_b)}, \quad i \in \{f, b\}, \quad (4)$$

where the segment  $\mathbf{s}$  is defined by the coordinate  $r_s$ . The value describes the probability that a specific segment belongs to the foreground or background.

In addition to those measurements, theoretical probabilities that depend on the location of the object contour are developed. They are modeled by smoothed step functions

$$h_f(x) = \frac{1}{2} - \alpha_h \tanh\left(\frac{x}{2s_h}\right), \quad (5)$$

$$h_b(x) = \frac{1}{2} + \alpha_h \tanh\left(\frac{x}{2s_h}\right), \quad (6)$$

with the amplitude parameter  $\alpha_h \in [0, 0.5]$  and the slope parameter  $s_h \in \mathbb{R}^+$ . Using the variated 3D model point

$${}_C\tilde{\mathbf{X}}(\theta) = {}_C\mathbf{T}_{MM}\mathbf{T}(\theta)_M\tilde{\mathbf{X}}, \quad (7)$$

the distance from the estimated contour to the correspondence line center  $\mathbf{c}$  is approximated in scale space as follows

$$d_s(\theta) = \left( \mathbf{n}^\top (\pi({}_C\mathbf{X}(\theta)) - \mathbf{c}) - \Delta r \right) \frac{\bar{n}}{s}, \quad (8)$$

where  $\bar{n} = \|\mathbf{n}\|_{\max}$  projects to the closest horizontal or vertical image coordinate, and  $\Delta r \in \mathbb{R}$  is an offset to a defined pixel location. An illustration of the transformation is shown in Fig. 2. Finally, based on those functions, *SRT3D* estimates the PDF for the scaled contour distance as

$$p(d_s(\theta) \mid \omega_s, \mathbf{l}) \propto \prod_{r_s \in \omega_s} \sum_{i \in \{f, b\}} h_i(r_s - d_s(\theta)) p_{si}(r_s), \quad (9)$$

with  $\omega_s$  the considered correspondence line domain.

Note that this PDF is defined in scale space. Thanks to the proof in [55], we know that, under certain conditions, the variance of the PDF is equal to the slope parameter  $s_h$  defined for the smoothed step functions  $h_f$  and  $h_b$ . Given this variance, the expected unscaled variance in units of pixels is  $\sigma^2 = s_h s^2 / \bar{n}^2$ . In contrast to previous work, we want correspondence lines to be independent of scale and slope parameters. This has the advantage that for all correspondence lines, the variance is defined in the same unit of pixels. In addition, we want to define our confidence in the region modality. Introducing the user-defined standard deviation  $\sigma_r$ , the PDF in Eq. (9) is thus scaled as follows

$$p(\theta \mid \omega_s, \mathbf{l}) \propto p(d_s(\theta) \mid \omega_s, \mathbf{l}) \frac{s_h s^2}{\sigma_r^2 \bar{n}^2}. \quad (10)$$

The formulation helps to fuse the region modality with other information, given a defined uncertainty. Also, as shown in the supplementary, it improves results compared to *SRT3D*.

### 3.4. Depth Modality

Based on *ICP* [2, 9], the depth modality starts with a search for correspondence points. Similar to projective data association [4], a 3D surface point  $\mathbf{X}$  from the sparse viewpoint model is first projected into the depth image. Given a user-defined radius and stride, multiple 3D points within a quadratic area are then reconstructed. Finally, a correspondence point  $\mathbf{P} \in \mathbb{R}^3$  is selected as the point closest to the point  $\mathbf{X}$ . Correspondences with a distance bigger than a threshold are rejected. Note that techniques such as normal shooting [9, 19], and rejection strategies based on the median distance [16], the best percentage [43], and the compatibility of normal vectors [70] were also tested. However, in the end, this simple procedure worked best.

For the probabilistic model, we formulate a normal distribution that uses the point-to-plane error metric [9]. The distance between the 3D surface point  $\mathbf{X}$  and correspondence point  $\mathbf{P}$  is thereby calculated along the associated normal vector  $\mathbf{N}$ . Given the correspondence point  $\mathbf{P}$ , the probability for the pose variation vector  $\boldsymbol{\theta}$  is written as

$$p(\boldsymbol{\theta} | \mathbf{P}) \propto \exp \left( -\frac{1}{2d_Z^2 \sigma_d^2} \left( \mathbf{M} \mathbf{N}^\top (\mathbf{M} \mathbf{X} - \mathbf{M} \mathbf{P}(\boldsymbol{\theta})) \right)^2 \right) \quad (11)$$

with

$$\mathbf{M} \tilde{\mathbf{P}}(\boldsymbol{\theta}) = \mathbf{M} \mathbf{T}(-\boldsymbol{\theta}) \mathbf{M} \mathbf{T}_{DD} \tilde{\mathbf{P}}. \quad (12)$$

Note that the user-defined standard deviation  $\sigma_d$  is scaled by the depth value  $d_Z$  of the correspondence point  $\mathbf{P}$ . The scaling takes into account that the number and quality of depth measurements decreases with the distance to the camera. In addition, it also ensures compatibility with the uncertainty of the region modality, which increases with the camera distance. In Eq. (11), we variate the correspondence point  $\mathbf{P}$  instead of the model. This has the advantage that the normal vector remains fixed, and only one vector has to be variated. Based on the derived PDFs for region and depth, we can now optimize for the pose that best explains the data.

## 4. Optimization

In the following, we first introduce the Newton method with Tikhonov regularization that is used to maximize the probability. Subsequently, we define the gradient vector and the Hessian matrix that are required in this optimization.

### 4.1. Regularized Newton Method

Assuming that measurements from both modalities are independent, the joint probability function is written as

$$p(\boldsymbol{\theta} | \mathcal{D}) = \prod_{i=0}^{n_r} p(\boldsymbol{\theta} | \omega_{si}, \mathbf{l}_i) \prod_{i=0}^{n_d} p(\boldsymbol{\theta} | \mathbf{P}_i), \quad (13)$$

with  $\mathcal{D}$  the considered data and  $n_r$  and  $n_d$  the number of used correspondence lines and correspondence points, re-

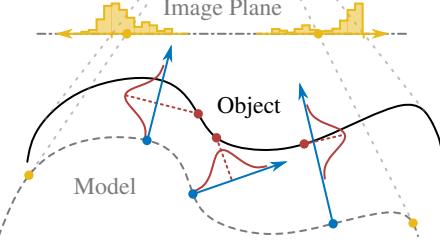


Figure 3. Optimization of the corresponding geometry. For blue surface points and normal vectors, the probability given a red correspondence point is illustrated by a normal distribution. For yellow correspondence lines, the discrete distribution is illustrated. The location of 3D contour points is thereby projected to the image plane. During optimization, the joint probability is maximized.

spectively. The joint optimization of correspondence lines and correspondence points is visualized in Fig. 3.

To maximize the probability, multiple iterations, where we calculate the variation vector  $\hat{\boldsymbol{\theta}}$  and update the object pose, are performed. In each iteration, the Newton method with Tikhonov regularization is used as follows

$$\hat{\boldsymbol{\theta}} = \left( -\mathbf{H} + \begin{bmatrix} \lambda_r \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \lambda_t \mathbf{I}_3 \end{bmatrix} \right)^{-1} \mathbf{g}, \quad (14)$$

where  $\mathbf{g}$  is the gradient vector,  $\mathbf{H}$  the Hessian matrix, and  $\lambda_r$  and  $\lambda_t$  are the rotational and translational regularization parameters. The gradient vector and the Hessian matrix are thereby defined as the first- and second-order derivatives of the logarithm of the joint probability function in Eq. (13).

The big advantage of the Newton formulation is that, with the Hessian matrix, uncertainty is considered in all dimensions. This means that, in addition to weighting the two modalities with  $\sigma_r$  and  $\sigma_d$ , it also takes into account how well each correspondence constrains the different directions. In addition, Tikhonov regularization acts as a prior probability that controls how much we believe in the previous pose. For directions with little information, this regularization helps to keep the optimization stable.

Finally, given the knowledge that  $\hat{\boldsymbol{\theta}}_r$  corresponds to the rotation vector of the axis-angle representation, we are able to update the pose using the exponential map as follows

$${}^A \mathbf{T}_M^+ = {}^A \mathbf{T}_M \begin{bmatrix} \exp([\hat{\boldsymbol{\theta}}_r] \times) & \hat{\boldsymbol{\theta}}_t \\ \mathbf{0} & 1 \end{bmatrix}, \quad A \in \{C, D\}. \quad (15)$$

Note that, typically, either the pose with respect to the color or the depth camera is calculated using Eq. (15). The other is then updated using the known relative transformation.

### 4.2. Gradient and Hessian

Because the logarithm is applied in the calculation of the gradient vector and the Hessian matrix, products turn into

summations and, based on Eq. (13), we can write

$$\mathbf{g} = \sum_{i=1}^{n_r} \mathbf{g}_{ri} + \sum_{i=1}^{n_d} \mathbf{g}_{di}, \quad (16)$$

$$\mathbf{H} = \sum_{i=1}^{n_r} \mathbf{H}_{ri} + \sum_{i=1}^{n_d} \mathbf{H}_{di}, \quad (17)$$

where  $\mathbf{g}_{ri}$  and  $\mathbf{H}_{ri}$  are calculated from individual correspondence lines of the region modality, and  $\mathbf{g}_{di}$  and  $\mathbf{H}_{di}$  are based on correspondence points from the depth modality.

For the region modality, we apply the chain rule to calculate gradient vectors and Hessian matrices as follows

$$\mathbf{g}_{ri} = \frac{s_h s^2}{\sigma_r^2 \bar{n}_i^2} \frac{\partial \ln(p(d_{si} | \omega_{si}, \mathbf{l}_i))}{\partial d_{si}} \frac{\partial d_{si}}{\partial \mathbf{C} \mathbf{X}_i} \bigg|_{\boldsymbol{\theta}=\mathbf{0}}, \quad (18)$$

$$\begin{aligned} \mathbf{H}_{ri} \approx & \frac{s_h s^2}{\sigma_r^2 \bar{n}_i^2} \frac{\partial^2 \ln(p(d_{si} | \omega_{si}, \mathbf{l}_i))}{\partial d_{si}^2} \left( \frac{\partial d_{si}}{\partial \mathbf{C} \mathbf{X}_i} \frac{\partial \mathbf{C} \mathbf{X}_i}{\partial \boldsymbol{\theta}} \right)^\top \\ & \left( \frac{\partial d_{si}}{\partial \mathbf{C} \mathbf{X}_i} \frac{\partial \mathbf{C} \mathbf{X}_i}{\partial \boldsymbol{\theta}} \right) \bigg|_{\boldsymbol{\theta}=\mathbf{0}}. \end{aligned} \quad (19)$$

Note that, similar to [55], second-order partial derivatives for  $d_{si}$  and  $\mathbf{C} \mathbf{X}_i$  are neglected. Using Eqs. (7) and (8), the following first-order partial derivatives can be calculated

$$\begin{aligned} \frac{\partial d_{si}}{\partial \mathbf{C} \mathbf{X}_i} = & \frac{\bar{n}_i}{s} \frac{1}{\mathbf{C} Z_i^2} [n_{xi} f_{xc} Z_i & n_{yi} f_{yc} Z_i \\ & - n_{xi} f_{xc} X_i - n_{yi} f_{yc} Y_i], \end{aligned} \quad (20)$$

$$\frac{\partial \mathbf{C} \mathbf{X}_i}{\partial \boldsymbol{\theta}} = \mathbf{C} \mathbf{R}_M \left[ -[\mathbf{M} \mathbf{X}_i] \times \mathbf{I}_3 \right]. \quad (21)$$

To estimate the first- and second-order partial derivatives for the posterior probability distribution  $p(d_{si} | \omega_{si}, \mathbf{l}_i)$ , we use the same techniques as in [56] and differentiate between global and local optimization. For global optimization, the PDF is sampled over  $d_{si}$ , and the mean  $\mu_i$  and variance  $\sigma_i^2$  are calculated to approximate a normal distribution. Based on this normal distribution, derivatives are calculated as

$$\frac{\partial \ln(p(d_{si} | \omega_{si}, \mathbf{l}_i))}{\partial d_{si}} \approx -\frac{1}{\sigma_i^2} (d_{si} - \mu_i), \quad (22)$$

$$\frac{\partial^2 \ln(p(d_{si} | \omega_{si}, \mathbf{l}_i))}{\partial d_{si}^2} \approx -\frac{1}{\sigma_i^2}. \quad (23)$$

For local optimization, the two probability values corresponding to the discrete distances  $d_{si}^-$  and  $d_{si}^+$  closest to  $d_{si}$  are used to approximate first-order partial derivatives as

$$\frac{\partial \ln(p(d_{si} | \omega_{si}, \mathbf{l}_i))}{\partial d_{si}} \approx \frac{\alpha_s}{\sigma_i^2} \ln \left( \frac{p(d_{si}^+ | \omega_{si}, \mathbf{l}_i)}{p(d_{si}^- | \omega_{si}, \mathbf{l}_i)} \right), \quad (24)$$

where  $\alpha_s$  is a user-defined learning rate. Second-order partial derivatives are again calculated according to Eq. (23).

Finally, for the depth modality, gradient vectors and Hessian matrices can be calculated using Eqs. (11) and (12)

$$\mathbf{g}_{di} = -\frac{1}{d_Z^2 \sigma_d^2} \mathbf{M} \mathbf{N}_i^\top (\mathbf{M} \mathbf{X}_i - \mathbf{M} \mathbf{P}_i) \begin{bmatrix} \mathbf{M} \mathbf{P}_i \times \mathbf{M} \mathbf{N}_i \\ \mathbf{M} \mathbf{N}_i \end{bmatrix}, \quad (25)$$

$$\mathbf{H}_{di} = -\frac{1}{d_Z^2 \sigma_d^2} \begin{bmatrix} \mathbf{M} \mathbf{P}_i \times \mathbf{M} \mathbf{N}_i \\ \mathbf{M} \mathbf{N}_i \end{bmatrix} \begin{bmatrix} \mathbf{M} \mathbf{P}_i \times \mathbf{M} \mathbf{N}_i \\ \mathbf{M} \mathbf{N}_i \end{bmatrix}^\top. \quad (26)$$

## 5. Implementation

The following section provides implementation details, discusses how *ICG* can be used for pose refinement, and explains how occlusions are handled. For our implementation, we built on the code of *SRT3D* [56]. To generate the sparse viewpoint model, the object is rendered from 2562 virtual cameras that are placed on a geodesic grid with a distance of 0.8 m to the object center. For each view, contour and surface points are sampled and normal vectors are approximated. For contour points, distances along the normal vector for which the foreground and background are not interrupted are also computed. To ensure only one transition exists on a correspondence line, lines with at least one of the two distances below 3 segments are rejected.

The two color histograms for foreground and background are discretized by 4096 equidistant bins. In their calculation, the first 20 pixels from the line center are used. During tracking, we update histograms once the final pose was computed using the online adaptation of [3] with a learning rate of  $\alpha = 0.2$ . In addition to tracking, our algorithm can also be used for pose refinement. In such cases, we initialize histograms at each iteration before correspondences are established. Since we do not perform a continuous update, histograms do not show the same quality as for tracking. Nevertheless, they still include useful information that helps the algorithm to converge. Experiments that demonstrate the performance for pose refinement are provided in the supplementary material.

For the region modality, the probability distributions  $p(d_{si} | \omega_{si}, \mathbf{l}_i)$  are evaluated at 12 discrete distance values  $d_{si} \in \{-5.5, -4.5, \dots, 5.5\}$ . In the calculation of each probability value, we use 8 precomputed values for the smoothed step functions  $h_f$  and  $h_b$  that correspond to  $x \in \{-3.5, -2.5, \dots, 3.5\}$ . Also, we define the slope parameter  $s_h = 0.5$ , the amplitude parameter  $\alpha_h = 0.43$ , and the learning rate  $\alpha_s = 1.3$ . To find correspondence points for the depth modality, image values on a quadratic grid with a stride of 5 mm and a radius equal to the correspondence threshold  $r_t$  are considered. Both parameter values are projected from meters into pixels based on the depth of the 3D surface point. Correspondence points with a distance that is bigger than the threshold  $r_t$  are rejected. Valid correspondences are then used in the optimization with the regularization parameters  $\lambda_r = 1000$  and  $\lambda_t = 30000$ .

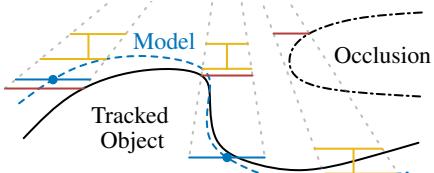


Figure 4. Visualization of the occlusion handling strategy. For each blue model point, the considered region is defined by a blue line and dotted gray cone. The lower yellow line in each cone visualizes the depth value that is calculated from the model offset, while the upper yellow line adds the user-defined threshold. Red lines indicate the minimum depth measurements from the camera. For the right point, an occlusion is detected since the red depth measurement is smaller than the expected value in yellow.

To find the final pose, we conduct 4 iterations in which correspondences are established. For each iteration, the standard deviations  $\sigma_r$  and  $\sigma_d$ , the scale  $s$ , and the threshold  $r_t$  can be adjusted. This allows to define our confidence in the data and the range in which region and depth information is considered. Many characteristics such as the resolution, depth image quality, or frame-to-frame pose difference depend on the sequence. We thus adjust the parameters for every dataset and provide them in the evaluation section. Finally, in each iteration, two optimization steps are conducted. For the region modality, global optimization is used in the first and local optimization in the second.

In many real-world cases, correspondence lines and correspondence points are subject to occlusion. Based on measurements from the depth camera and estimated positions of 3D model points, occlusions can be detected. First, the minimum depth based on 25 depth image values within a quadratic region of  $20 \times 20$  mm is computed. Similarly, during model generation, an offset between the depth of the sampled model point and the minimum depth within a quadratic region of  $20 \times 20$  mm is calculated. Finally, we are able to reject correspondences for which the depth of the model point minus the precomputed offset and a user-defined threshold of 30 mm is smaller than the measured minimum depth. Considering a region of values makes the technique robust to missing depth measurements and large local depth differences in the object surface. In cases where depth images are not available, depth renderings can be used. An illustration of the strategy is shown in Fig. 4.

## 6. Evaluation

In this section, we present an extensive evaluation of our approach on the *YCB-Video* dataset [68], the *OPT* dataset [67], and the dataset of Choi [10]. We thereby evaluate the robustness, accuracy, and efficiency of *ICG* in comparison to the state of the art. In addition, we conduct an ablation study that demonstrates the importance of individual com-

ponents. Finally, we explain limitations of our approach. Note that in the supplementary material, we present further results for the region modality and the Choi dataset. Also, we discuss how *ICG* performs for pose refinement and how our tracker compares to modern 6DoF pose estimation algorithms. Qualitative results on the *YCB-Video* dataset and in the real world are shown in the provided videos<sup>1</sup>.

### 6.1. YCB-Video Dataset

The *YCB-Video* dataset [68] contains 21 *YCB* objects [8] and evaluates on 12 sequences with a total of 2949 key frames. Because it includes additional training sequences, it is very popular with deep learning-based methods. In the evaluation, the conventional and symmetric average distance errors  $e_{ADD}$  and  $e_{ADD-S}$  [23] are calculated as

$$e_{ADD} = \frac{1}{n} \sum_{i=1}^n \|(\mathbf{M}\tilde{\mathbf{X}}_i - \mathbf{M}\mathbf{T}_{\mathbf{M}_{gt}\mathbf{M}}\tilde{\mathbf{X}}_i)_{3 \times 1}\|_2, \quad (27)$$

$$e_{ADD-S} = \frac{1}{n} \sum_{i=1}^n \min_{j \in [n]} \|(\mathbf{M}\tilde{\mathbf{X}}_i - \mathbf{M}\mathbf{T}_{\mathbf{M}_{gt}\mathbf{M}}\tilde{\mathbf{X}}_j)_{3 \times 1}\|_2, \quad (28)$$

where  $\mathbf{M}\mathbf{T}_{\mathbf{M}_{gt}}$  is the difference between the ground-truth and the estimated model pose,  $\tilde{\mathbf{X}}_i$  is a vertex from the object mesh, and  $n$  is the number of vertices. Based on those error metrics for a single frame, [68] reports *ADD* and *ADD-S* area under curve scores. They can be calculated as

$$s_i = \frac{1}{m} \sum_{j=1}^m \max \left( 1 - \frac{e_{ij}}{e_t}, 0 \right), \quad (29)$$

with  $i \in \{\text{ADD, ADD-S}\}$ , the respective frame error  $e_{ij}$ , the number of frames  $m$ , and the threshold  $e_t = 0.1$  m.

Results of the evaluation on the *YCB-Video* dataset are shown in Tab. 1. For our algorithm, we use the parameters  $\sigma_r = \{25, 15, 10\}$ ,  $\sigma_d = \{50, 30, 20\}$ ,  $s = \{7, 4, 2\}$ , and  $r_t = \{70, 50, 40\}$ , where values are given in units of pixels and millimeters. Our method is compared to *PoseCNN* [68], the particle-filter-based approaches of [66] and [24], and the current state of the art in deep learning-based 3D object tracking [31], [64], and [15]. The evaluation shows that *ICG* achieves state-of-the-art results with respect to the *ADD-S* metric, outperforming all other algorithms. For the *ADD* score, textureless methods have a significant disadvantage since, for some objects, the geometry is not conclusive. It is, for example, not possible to determine the rotation of a rotationally symmetric object without using texture. However, even with that handicap, *ICG* surpasses the texture-based approaches of *PoseCNN* and *DeepIM*. Also, it comes very close to the results of *PoseRBPF*. In the end, only *se(3)-TrackNet* is able to perform significantly better.

To utilize the full frequency of modern cameras, track multiple objects simultaneously, and save resources, efficiency is essential in real-world applications. We thus report the speed and the required hardware for all algorithms

Table 1. Results on the *YCB-Video* dataset [68] with *ADD* and *ADD-S* area under curve scores in percent. Except for *PoseRBPF* [15], results are taken from [64]. For *DeepIM* [31], the score over all frames was adjusted to be consistent with the evaluation of other methods. Objects with no conclusive geometry are indicated by a \* while objects with no or very little texture are marked by a  $\diamond$ .

Approach	PoseCNN + ICP + DeepIM [68]		Wüthrich [66]		RGF [24]		DeepIM [31]		se(3)-TrackNet [64]		PoseRBPF + SDF [15]		ICG (Ours)		
Initial Pose Re-initialization	-		Ground Truth No		Ground Truth No		Ground Truth Yes (290)		Ground Truth No		PoseCNN Yes (2)		Ground Truth No		
Objects		ADD	ADD-S		ADD	ADD-S		ADD	ADD-S		ADD	ADD-S		ADD	ADD-S
002_master_chef_can*	78.0	96.3	55.6	90.7	46.2	90.2	89.0	93.8	93.9	96.3	89.3	96.7	66.4	89.7	
003_cracker_box	91.4	95.3	96.4	97.2	57.0	72.3	88.5	93.0	96.5	97.2	96.0	97.1	82.4	92.1	
004_sugar_box	97.6	98.2	97.1	97.9	50.4	72.7	94.3	96.3	97.6	98.1	94.0	96.4	96.1	98.4	
005_tomato_soup_can*	90.3	94.8	64.7	89.5	72.4	91.6	89.1	93.2	95.0	97.2	87.2	95.2	73.2	97.3	
006_mustard_bottle	97.1	98.0	97.1	98.0	87.7	98.2	92.0	95.1	95.8	97.4	98.3	98.5	96.2	98.4	
007_tuna_fish_can*	92.2	98.0	69.1	93.3	28.7	52.9	92.0	96.4	86.5	91.1	86.8	93.6	73.2	95.8	
008_pudding_box	83.5	90.6	96.8	97.9	12.7	18.0	80.1	88.3	97.9	98.4	60.9	87.1	73.8	88.9	
009_gelatin_box	98.0	98.5	97.5	98.4	49.1	70.7	92.0	94.4	97.8	98.4	98.2	98.6	97.2	98.8	
010_potted_meat_can	82.2	90.3	83.7	86.7	44.1	45.6	78.0	88.9	77.8	84.2	76.4	83.5	93.3	97.3	
011_banana $\diamond$	94.9	97.6	86.3	96.1	93.3	97.7	81.0	90.5	94.9	97.2	92.8	97.7	95.6	98.4	
019_pitcher_base $\diamond$	97.4	97.9	97.3	97.7	97.9	98.2	90.4	94.7	96.8	97.5	97.7	98.1	97.0	98.8	
021_bleach_cleanser	91.6	96.9	95.2	97.2	95.9	97.3	81.7	90.5	95.9	97.2	95.9	97.0	92.6	97.5	
024_bowl $\diamond$ *	8.1	87.0	30.4	97.2	24.2	82.4	38.8	90.6	80.9	94.5	34.0	93.0	74.4	98.4	
025_mug $\diamond$	94.2	97.6	83.2	93.3	60.0	71.2	83.2	92.0	91.5	96.9	86.9	96.7	95.6	98.5	
035_power_drill	97.2	97.9	97.1	97.8	97.9	98.3	85.4	92.3	96.4	97.4	97.8	98.2	96.7	98.5	
036_wood_block	81.1	91.5	95.5	96.9	45.7	62.5	44.3	75.4	95.2	96.7	37.8	93.6	93.5	97.2	
037_scissors $\diamond$	92.7	96.0	4.2	16.2	20.9	38.6	70.3	84.5	95.7	97.5	72.7	85.5	93.5	97.3	
040_large_marker*	88.9	98.2	35.6	53.0	12.2	18.9	80.4	91.2	92.2	96.0	89.2	97.3	88.5	97.8	
051_large_clamp $\diamond$	54.2	77.9	61.2	72.3	62.8	80.1	73.9	84.1	94.7	96.9	90.1	95.5	91.8	96.9	
052_extra_large_clamp $\diamond$	36.5	77.8	93.7	96.6	67.5	69.7	49.3	90.3	91.7	95.8	84.4	94.1	85.9	94.3	
061_foam_brick $\diamond$	48.2	97.6	96.8	98.1	70.0	86.5	91.6	95.5	93.7	96.7	96.1	98.3	96.2	98.5	
All Frames	80.7	94.0	78.0	90.2	59.2	74.3	82.3	91.9	93.0	95.7	87.5	95.2	86.4	96.5	

Table 2. Average speed in frames per second and hardware requirements for the CPU and GPU. Except for *PoseRBPF* [15], results are taken from [64]. *PoseRBPF* was evaluated without *SDF*.

Approach	Single Core	No GPU	FPS
PoseCNN+ICP+DeepIM [68]		$\times$	0.1 Hz
Wüthrich [66]	✓	✓	12.9 Hz
RGF [24]	✓	✓	11.8 Hz
DeepIM [31]		$\times$	12.0 Hz
se(3)-TrackNet [64]		$\times$	90.9 Hz
PoseRBPF [15]		$\times$	7.6 Hz
ICG (Ours)	✓	✓	788.4 Hz

in Tab. 2. The evaluation of *ICG* is conducted on an *Intel Xeon E5-1630 v4* CPU. In comparison, [64] used an *Intel Xeon E5-1660 v3* CPU and a *NVIDIA Tesla K40c* GPU. The results show the outstanding efficiency of our approach. While *ICG* runs only on a single CPU core, it is almost one order of magnitude faster than the second-best algorithm *se(3)-TrackNet*, which requires a high-performance GPU.

## 6.2. OPT Dataset

While the *YCB-Video* dataset features challenging sequences in real-world environments and a large number of objects, ground truth has limited accuracy and with a large threshold  $e_t = 0.1$  m, the dataset mostly evaluates robust-

Table 3. Results on the *OPT* dataset [67]. The reported *AUC* scores are scaled between zero and twenty to match other evaluations. Results are taken from [67] and the respective publications.

Approach	Soda	Chest	Ironman	House	Bike	Jet	Avg.
PWP3D [42]	5.87	5.55	3.92	3.58	5.36	5.81	5.01
ElasticFusion [65]	1.90	1.53	1.69	2.70	1.57	1.86	1.87
UDP [5]	8.49	6.79	5.25	5.97	6.10	2.34	5.82
ORB-SLAM2 [36]	13.44	15.53	11.20	17.28	10.41	9.93	12.97
Bugaev [7]	14.85	14.97	14.71	14.48	12.55	17.17	14.79
Tjaden [60]	8.86	11.76	11.99	10.15	11.90	13.22	11.31
Zhong [73]	9.01	12.24	11.21	13.61	12.83	15.44	12.39
Li [30]	9.00	14.92	13.44	13.60	12.85	10.64	12.41
SRT3D [56]	15.64	16.30	17.41	16.36	13.02	15.64	15.73
ICG (Ours)	15.32	15.85	17.86	17.92	16.36	15.90	16.54

ness. Also, images do not contain motion blur, favoring texture-based methods. The *OPT* dataset [67] nicely complements those properties. It includes 6 objects and consists of 552 real-world sequences with significant motion blur. Ground truth is obtained using a calibration board. For the evaluation, the area under curve of the *ADD* metric is used with a threshold of  $r_t = 0.2d$ , where  $d$  is the largest distance between model vertices. The final value is scaled between 0 and 20. Following [67], we refer to it as *AUC* score.

Evaluation results are reported in Tab. 3. For *ICG*, the parameters  $\sigma_r = \{15, 5, 1.5\}$ ,  $\sigma_d = \{35, 35, 25\}$ ,  $s = \{6, 4, 1\}$ , and  $r_t = \{50, 20, 10\}$  are used. Also, like in [56], we constrain the rotationally symmetric soda object using

Table 4. Mean RMS errors for translation and rotation parameters on the Choi dataset [10]. Results are from the respective papers.

Approach	Choi [10]	Krull [27]	Tan [59]	Kehl [25]	ICG (Ours)
Translation [mm]	1.36	0.82	<u>0.10</u>	0.51	<b>0.04</b>
Rotation [degree]	2.45	1.38	<u>0.07</u>	0.26	<b>0.04</b>

Table 5. Ablation study on critical components of our algorithm.

Dataset	Choi [10]		OPT [67]		YCB-Video [68]
Experiment	Trans.	Rot.	AUC	ADD	ADD-S
Original	0.04	0.04	16.54	86.4	96.5
W/o Region	0.06	0.04	8.94	66.1	84.1
W/o Depth	41.65	23.39	15.88	26.6	42.8
W/o Regularization	0.04	0.04	14.48	72.0	91.5
W/o Occlusion Hand.	-	-	-	77.6	91.9

$\lambda_r = 70000$ . In the evaluation, *ICG* is compared to state-of-the-art classical methods that use different sources of information, including region, edge, texture, and depth. Our approach performs best or second-best for all objects and improves on the state of the art by a significant margin.

### 6.3. Choi Dataset

Finally, we also want to evaluate the accuracy of our method. For this, the simulated dataset of Choi [10], which features four sequences and perfect ground truth, is used. To evaluate the accuracy, root mean square (RMS) errors in the x, y, and z directions and in the roll, pitch, and yaw angles are calculated. The rotational and translational mean over all four sequences is reported in Tab. 4. Detailed results are provided in the supplementary material.

For our algorithm, we use the parameters  $\sigma_r = \{5\}$ ,  $\sigma_d = \{10, 1\}$ ,  $s = \{2, 1\}$ , and  $r_t = \{10\}$ . Note that since the dataset provides perfect depth and uncluttered color images, results have to be considered as an upper bound. Nevertheless, the experiments demonstrate that, with good enough data, the method is highly accurate.

### 6.4. Ablation Study

To demonstrate the importance of the algorithm’s components, we perform an ablation study on all three datasets. Average results of this evaluation are provided in Tab. 5. The experiments show that while the effect of the depth and region modality differs between the datasets, each modality significantly contributes to the final result. Also, we observe that while values on the Choi dataset stay the same without regularization, it is very important to the challenging sequences of the *OPT* and *YCB-Video* dataset. For the case without occlusion handling, similar results are obtained.

Finally, the convergence of our approach is evaluated in Fig. 5. For the *YCB-Video* dataset, which values robustness, the algorithm converges after only two iterations. With

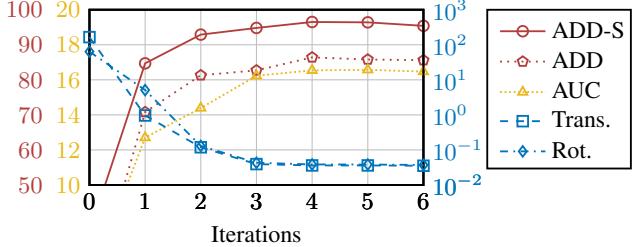


Figure 5. Convergence plot showing the final results for the *YCB-Video*, *OPT*, and Choi dataset in red, yellow, and blue, respectively.

one additional iteration, accurate results are obtained on the Choi and *OPT* datasets. Independent of accuracy and robustness, fast convergence ensures that a maximum of only four iterations is sufficient to obtain excellent results.

### 6.5. Limitations

While *ICG* achieves remarkable results, some limitations remain. First, our method requires the geometry of tracked objects. Also, for the region modality, objects have to be distinguishable from the background. In addition, the depth camera has to provide reasonable measurements for the object’s surface. Another important limitation emerges if the object geometry is very similar in the vicinity of a particular pose. Naturally, using geometric information alone, it is impossible to predict the correct pose in such cases. Finally, like many conventional approaches that use line search, the algorithm has only a local view of the six-dimensional joint probability distribution. As a consequence, it is constrained by local minima with a limit to the maximum pose difference between consecutive frames.

### 7. Conclusion

In this work, we developed *ICG*, a highly efficient, textureless approach to 3D object tracking. The method fuses region and depth in a well-founded probabilistic formulation that is able to handle occlusions in a robust manner. While the overall algorithm is relatively simple and requires little computation, it performs surprisingly well on multiple datasets, outperforming the current state of the art for many cases in robustness, accuracy, and efficiency. This is even more remarkable since *ICG* has an inherent disadvantage in not using texture. Assuming that texture would further improve results, our evaluation suggests that deep learning-based techniques do not yet surpass classical methods. This is especially surprising since, at the expense of high computational cost and limited real-time capability, such algorithms can, in theory, directly consider all available information. As a consequence, we believe that for both classical and learning-based tracking, as well as potential combinations, there remains a large number of ideas that wait to be explored to further improve 3D object tracking.

## References

[1] Selim Benhimane and Ezio Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 943–948, 2004. 1

[2] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. 2, 4

[3] Charles Bibby and Ian Reid. Robust real-time visual tracking using pixel-wise posteriors. In *European Conference on Computer Vision*, pages 831–844, 2008. 2, 5

[4] Gérard Blais and Martin D. Levine. Registering multiview range data to create 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, 1995. 2, 4

[5] Eric Brachmann, Frank Michel, Alexander Krull, Michael Y. Yang, Stefan Gumhold, and Carsten Rother. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3364–3372, 2016. 7

[6] Thomas Brox, Bodo Rosenhahn, Juergen Gall, and Daniel Cremers. Combined region and motion-based 3D tracking of rigid and articulated objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):402–415, 2010. 2

[7] Bogdan Bugaev, Anton Kryshchenko, and Roman Belov. Combining 3D model contour energy and keypoints for object tracking. In *European Conference on Computer Vision*, pages 55–70, 2018. 7

[8] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The YCB object and model set: Towards common benchmarks for manipulation research. In *International Conference on Advanced Robotics*, pages 510–517, 2015. 6

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992. 2, 4

[10] Changhyun Choi and Henrik I. Christensen. RGB-D object tracking: A particle filter approach on GPU. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1084–1091, 2013. 2, 6, 8

[11] Cristina G. Cifuentes, Jan Issac, Manuel Wüthrich, Stefan Schaal, and Jeannette Bohg. Probabilistic articulated real-time tracking for robot manipulation. *IEEE Robotics and Automation Letters*, 2(2):577–584, 2017. 1, 2

[12] Andrew I. Comport, Eric Marchand, Muriel Pressigout, and Francois Chaumette. Real-time markerless tracking for augmented reality: The virtual visual servoing framework. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):615–628, 2006. 1

[13] Alberto Crivellaro and Vincent Lepetit. Robust 3D tracking with descriptor fields. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3414–3421, 2014. 1

[14] Samuel Dambreville, Romeil Sandhu, Anthony Yezzi, and Allen Tannenbaum. Robust 3D pose estimation and efficient 2D region-based segmentation from a 3D shape prior. In *European Conference on Computer Vision*, pages 169–182, 2008. 2

[15] Xinkle Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. PoseRBPF: A Rao–Blackwellized particle filter for 6-D object pose tracking. *IEEE Transactions on Robotics*, 37(5):1328–1342, 2021. 1, 2, 6, 7

[16] James Diebel, Kjell Reutersward, Sebastian Thrun, James Davis, and Rakesh Gupta. Simultaneous localization and mapping with active stereo vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3436–3443, 2004. 4

[17] Tom Drummond and Roberto Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):932–946, 2002. 1

[18] Andrew W. Fitzgibbon. Robust registration of 2D and 3D point sets. *Image and Vision Computing*, 21(13-14):1145–1153, 2003. 2

[19] Hervé Gagnon, Marc Soucy, Robert Bergevin, and Denis Laurendeau. Registration of multiple range views for automatic 3-d model building. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 581–586, 1994. 4

[20] Chris Harris and Carl Stennett. RAPID - A video rate object tracker. In *Proceedings of the British Machine Vision Conference*, pages 15.1–15.6, 1990. 1

[21] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. FFB6D: A full flow bidirectional fusion network for 6D pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3003–3013, 2021. 2

[22] Jonathan Hexner and Rami R. Hagege. 2D-3D pose estimation of heterogeneous objects using a region based approach. *International Journal of Computer Vision*, 118(1):95–112, 2016. 2

[23] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, pages 548–562, 2013. 6

[24] Jan Issac, Manuel Wüthrich, Cristina G. Cifuentes, Jeannette Bohg, Sebastian Trimpe, and Stefan Schaal. Depth-based object tracking using a robust gaussian filter. In *IEEE International Conference on Robotics and Automation*, pages 608–615, 2016. 1, 2, 6, 7

[25] Wadim Kehl, Federico Tombari, Slobodan Ilic, and Nassir Navab. Real-time 3D model tracking in color and depth on a single CPU core. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 465–473, 2017. 1, 2, 8

[26] Michael Krainin, Peter Henry, Xiaofeng Ren, and Dieter Fox. Manipulator and object tracking for in-hand 3D object modeling. *International Journal of Robotics Research*, 30(11):1311–1327, 2011. 1

[27] Alexander Krull, Frank Michel, Eric Brachmann, Stefan Gumhold, Stephan Ihrke, and Carsten Rother. 6-DOF model based tracking via object coordinate regression. In *Asian Conference on Computer Vision*, pages 384–399, 2015. 2, 8

[28] Yann Labb , Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *European Conference on Computer Vision*, pages 574–591, 2020. 2

[29] Vincent Lepetit and Pascal Fua. *Monocular Model-Based 3D Tracking of Rigid Objects: A Survey*, volume 1. Foundations and Trends in Computer Graphics and Vision, 2005. 1

[30] Jia-Chen Li, Fan Zhong, Song-Hua Xu, and Xue-Ying Qin. 3D object tracking with adaptively weighted local bundles. *Journal of Computer Science and Technology*, 36(3):555–571, 2021. 2, 7

[31] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6D pose estimation. In *European Conference on Computer Vision*, pages 695–711, 2018. 1, 2, 6, 7

[32] Fulin Liu, Zhenzhong Wei, and Guangjun Zhang. An off-board vision system for relative attitude measurement of aircraft. *IEEE Transactions on Industrial Electronics*, 69(4):4225–4233, 2021. 2

[33] Yang Liu, Pansiyu Sun, and Akio Namiki. Target tracking of moving and rotating object by high-speed monocular active vision. *IEEE Sensors Journal*, 20(12):6727–6744, 2020. 2

[34] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, volume 2, pages 674–679, 1981. 1

[35] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6D pose refinement in RGB. In *European Conference on Computer Vision*, pages 833–849, 2018. 2

[36] Raul Mur-Artal and Juan D. Tardós. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 7

[37] Richard A. Newcombe, Andrew J. Davison, Shahram Izadi, Pushmeet Kohli, Otmar Hilliges, Jamie Shotton, David Molyneaux, Steve Hodges, David Kim, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, 2011. 1, 2

[38] Mustafa Özuysal, Vincent Lepetit, François Fleuret, and Pascal Fua. Feature harvesting for tracking-by-detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *European Conference on Computer Vision*, pages 592–605, 2006. 1

[39] Karl Pauwels, Vladimir Ivan, Eduardo Ros, and Sethu Vijayakumar. Real-time object pose recognition and tracking with an imprecisely calibrated moving rgbd camera. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2733–2740, 2014. 2

[40] François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics*, 4(1):1–104, 2015. 2

[41] Victor A. Prisacariu, Olaf Kähler, David W. Murray, and Ian D. Reid. Real-time 3D tracking and reconstruction on mobile phones. *IEEE Transactions on Visualization and Computer Graphics*, 21(5):557–570, 2015. 2

[42] Victor A. Prisacariu and Ian D. Reid. PWP3D: Real-time segmentation and tracking of 3D objects. *International Journal of Computer Vision*, 98(3):335–354, 2012. 1, 2, 7

[43] Kari Pulli. Multiview registration for large data sets. In *Second International Conference on 3-D Digital Imaging and Modeling*, pages 160–168, 1999. 4

[44] Carl Y. Ren, Victor A. Prisacariu, Olaf Kähler, Ian D. Reid, and David W. Murray. Real-time tracking of single and multiple objects from depth-colour imagery using 3D signed distance functions. *International Journal of Computer Vision*, 124(1):80–95, 2017. 1, 2

[45] Carl Y. Ren and Ian Reid. A unified energy minimization framework for model fitting in depth. In Andrea Fusiello, Vittorio Murino, and Rita Cucchiara, editors, *Computer Vision – ECCV 2012. Workshops and Demonstrations*, pages 72–82, 2012. 2

[46] Bodo Rosenhahn, Thomas Brox, and Joachim Weickert. Three-dimensional shape knowledge for joint image segmentation and pose tracking. *International Journal of Computer Vision*, 73(3):243–262, 2007. 2

[47] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1515, 2005. 1

[48] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision*, pages 2564–2571, 2011. 1

[49] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001. 2

[50] Christian Schmaltz, Bodo Rosenhahn, Thomas Brox, and Joachim Weickert. Region-based pose tracking with occlusions using 3D models. *Machine Vision and Applications*, 23(3):557–577, 2012. 2

[51] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Dart: dense articulated real-time tracking with consumer depth cameras. *Autonomous Robots*, 39(3):239–258, 2015. 1, 2

[52] Byung-Kuk Seo, Hanhoon Park, Jong-Il Park, Stefan Hinterstoisser, and Slobodan Ilic. Optimal local searching for fast and robust textureless 3D object tracking in highly cluttered backgrounds. *IEEE Transactions on Visualization and Computer Graphics*, 20(1):99–110, 2014. 1

[53] Byung-Kuk Seo and Harald Wuest. A direct method for robust model-based 3D object tracking from a monocular RGB image. In *European Conference on Computer Vision Workshop*, pages 551–562, 2016. 1

[54] Iryna Skrypnyk and David G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 110–119, 2004. 1

[55] Manuel Stoiber, Martin Pfanne, Klaus H. Strobl, Rudolph Triebel, and Alin Albu-Schaeffer. A sparse gaussian approach to region-based 6DoF object tracking. In *Asian Conference on Computer Vision*, pages 666–682, 2020. 2, 3, 5

[56] Manuel Stoiber, Martin Pfanne, Klaus H. Strobl, Rudolph Triebel, and Alin Albu-Schaeffer. SRT3D: A sparse region-based 3D object tracking approach for the real world. *International Journal of Computer Vision*, 2022. 1, 2, 3, 5, 7

[57] Xiaoliang Sun, Jiebin Zhou, Wenlong Zhang, Zi Wang, and Qifeng Yu. Robust monocular pose tracking of less-distinct

objects based on contour-part model. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(11):4409–4421, 2021. 2

[58] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3D orientation learning for 6D object detection from RGB images. In *European Conference on Computer Vision*, pages 712–729, 2018. 2

[59] David J. Tan, Nassir Navab, and Federico Tombari. Looking beyond the simple scenarios: Combining learners and optimizers in 3D temporal tracking. *IEEE Transactions on Visualization and Computer Graphics*, 23(11):2399–2409, 2017. 1, 2, 3, 8

[60] Henning Tjaden, Ulrich Schwanecke, Elmar Schömer, and Daniel Cremers. A region-based Gauss-Newton approach to real-time monocular multiple object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1797–1812, 2018. 1, 2, 7

[61] Luca Vaccetti, Vincent Lepetit, and Pascal Fua. Stable real-time 3D tracking using online and offline information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, 2004. 1

[62] Chen Wang, Roberto Martín-Martín, Danfei Xu, Jun Lv, Cewu Lu, Li Fei-Fei, Silvio Savarese, and Yuke Zhu. 6-PACK: Category-level 6D pose tracker with anchor-based keypoints. In *IEEE International Conference on Robotics and Automation*, pages 10059–10066, 2020. 1, 2

[63] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. DenseFusion: 6D object pose estimation by iterative dense fusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3338–3347, 2019. 2

[64] Bowen Wen, Chaitanya Mitash, Baozhang Ren, and Kostas E. Bekris. se(3)-TrackNet: Data-driven 6D pose tracking by calibrating image residuals in synthetic domains. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 10367–10373, 2020. 1, 2, 6, 7

[65] Thomas Whelan, Stefan Leutenegger, Renato S. Moreno, Ben Glocker, and Andrew Davison. ElasticFusion: Dense SLAM without a pose graph. In *Robotics: Science and Systems*, 2015. 7

[66] Manuel Wüthrich, Peter Pastor, Mrinal Kalakrishnan, Jeanette Bohg, and Stefan Schaal. Probabilistic object tracking using a range camera. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3195–3202, 2013. 1, 2, 6, 7

[67] Po-Chen Wu, Yueh-Ying Lee, Hung-Yu Tseng, Hsuan-I Ho, Ming-Hsuan Yang, and Shao-Yi Chien. A benchmark dataset for 6DoF object pose tracking. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 186–191, 2017. 6, 7, 8

[68] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *Robotics: Science and Systems*, 2018. 2, 6, 7, 8

[69] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13, 2006. 1

[70] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994. 4

[71] Song Zhao, Lingfeng Wang, Wei Sui, Huai-yu Wu, and Chunhong Pan. 3D object tracking via boundary constrained region-based model. In *IEEE International Conference on Image Processing*, pages 486–490, 2014. 2

[72] Leisheng Zhong and Li Zhang. A robust monocular 3D object tracking method combining statistical and photometric constraints. *International Journal of Computer Vision*, 127(8):973–992, 2019. 2

[73] Leisheng Zhong, Xiaolin Zhao, Yu Zhang, Shunli Zhang, and Li Zhang. Occlusion-aware region-based 3D pose tracking of objects with temporally consistent polar-based local partitioning. *IEEE Transactions on Image Processing*, 29:5065–5078, 2020. 1, 2, 7