IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2021

BundleTrack: 6D Pose Tracking for Novel Objects without Instance or Category-Level 3D Models

Bowen Wen and Kostas Bekris

***Abstract*— Tracking the 6D pose of objects in video sequences is important for robot manipulation. Most prior efforts, however, often assume that the target object's CAD model, at least at a category-level, is available for offline training or during online template matching. This work proposes *BundleTrack*, a general framework for 6D pose tracking of novel objects, which does not depend upon 3D models, either at the instance or category-level. It leverages the complementary attributes of recent advances in deep learning for segmentation and robust feature extraction, as well as memory-augmented pose graph optimization for spatiotemporal consistency. This enables long-term, low-drift tracking under various challenging scenarios, including significant occlusions and object motions. Comprehensive experiments given two public benchmarks demonstrate that the proposed approach significantly outperforms state-of-art, category-level 6D tracking or dynamic SLAM methods. When compared against state-of-art methods that rely on an object instance CAD model, comparable performance is achieved, despite the proposed method's reduced information requirements. An efficient implementation in CUDA provides a real-time performance of 10Hz for the entire framework. Code is available at: `https://github.com/wenbowen123/BundleTrack`**

I. INTRODUCTION

Robot manipulation often requires information about the pose of the manipulated object. In some cases, this can be achieved through forward kinematics (FK), assuming the object's motion equivalent to the end-effector's motion. Frequently, however, FK is insufficient to accurately estimate the object's pose [1]. This can be due to slippage during grasping or in-hand manipulation [2], or during handoffs or due to the compliance of a suction cup (Fig. 1). In these cases, dynamically estimating an object's pose from visual data is desirable. Single-image 6D pose estimation methods have been studied extensively [3]–[7]. Some of them are fast and can re-estimate poses from scratch for every new frame [8], [9]. Nevertheless, this is redundant, less efficient, leading to less coherent estimations over consecutive frames and negatively impacts planning and control. On the other hand, given an initial pose estimate, tracking 6D object poses over image sequences can improve estimation speed while providing coherent and accurate poses by leveraging temporal consistency [10]–[12].

Most existing 6D object pose estimation or tracking approaches assume access to an object instance's 3D model [3], [9]. Having access to such *instance 3D models* complicates generalization to novel, unseen instances. To overcome this limitation, recent efforts have relaxed this assumption and require only *category-level 3D models* for 6D pose estimation [13]–[16] or tracking [17]. They often achieve this by training over a large number of CAD models from the same category. While promising results have been demonstrated for previously seen object categories, there are still limitations. These methods are constrained by the variety of categories in the training database. Popular 3D model databases, such as *ShapeNet* [18] and *ModelNet40* [19], contain 55 and 40 categories respectively. This is still far from sufficient to cover diverse object categories present in the real world. Furthermore, 3D model databases often require nontrivial manual effort and expert domain knowledge to build, involving steps such as scanning [20], mesh refinement [21] or CAD design.

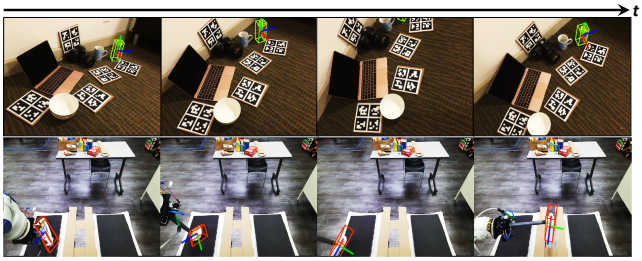


Fig. 1: **Top:** *NOCS Dataset* [13] example: The target object exits the camera's frustum during tracking but *BundleTrack* maintains its estimate without re-initialization. **Bottom:** *YCBInEOAT Dataset* [22] example: The object is successfully tracked during pick and place manipulation by a robotic arm, despite the lack of texture, severe self-occlusion and motions due to the arm and the compliant suction cup. Computing object pose from forward kinematics is unreliable in this setup due to the end-effector.

Another line of work from the SLAM literature has moved to address dynamic, object-aware challenges [23]–[26], where dynamic objects are being reconstructed on-the-fly while being tracked without the need for object 3D models beforehand. However, tracking-via-reconstruction [24], [26] tends to accumulate errors when fusing observations with erroneous pose estimates into the global model. These errors adversely impact model tracking in subsequent frames.

Motivated by the above limitations, this work aims for accurate, robust 6D pose tracking that is generalizable to novel objects without *instance or category-level 3D models*. It exploits recent advances in video segmentation as well as learning-based keypoint detection and matching for a coarse pose estimate, followed by a memory-augmented pose-graph optimization step to achieve spatiotemporal consistent pose output. Instead of aggregating into a global model, representative historical observations are maintained as keyframes in a memory pool, providing candidate nodes for future graphs so as to enable multi-pair data association together with the latest observation. An efficient implementation of this framework in CUDA allows to achieve competitive running times. Extensive experiments have been conducted

The authors are with the Computer Science Dept. of Rutgers in NJ, USA. Email: {bw344,kostas.bekris}@cs.rutgers.edu. This work is supported by NSF NRI award 1734492. The results do not express the sponsor's positions.

on two large-scale public benchmarks, shown in Fig. 1. Both qualitative and quantitative results demonstrate a significant improvement over existing state-of-art approaches, including methods using *instance or category-level 3D models* or SLAM-like methods.

In summary, this work's contributions are the following:

1) A novel integration of methods that result in a 6D pose tracking framework that generalizes to novel objects without access to instance or category-level 3D models.

2) A memory-augmented pose graph optimization for low-drift accurate 6D object pose tracking. In particular, augmenting the memory pool with historical observations enables multi-hop data association and ameliorate the dearth of correspondences between a pair of consecutive frames. Additionally, maintaining keyframes as raw nodes instead of aggregating into a global model significantly reduces tracking drift.

3) An efficient CUDA implementation, which allows to execute online the computationally-heavy multi-pair feature matching as well as pose-graph optimization for 6D object pose tracking (for the first time to the best of the authors' knowledge).

These contributions result in a new state-of-art performance by boosting the previous best accuracy from **33.3%** to **87.4%** under the "5°5cm" metric in the *NOCS Dataset* [13], even when compared against approaches utilizing category-level 3D models for training. They also result in comparable performance on the *YCBInEOAT* dataset [22], even when compared against approaches utilizing instance-level 3D models [22].

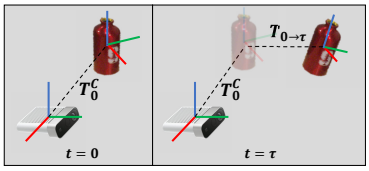
II. RELATED WORK

6D Object Pose Tracking - For setups where object CAD models are available, significant progress has been made in 6D pose tracking. This includes techniques based on hand-crafted probabilistic filtering [11], [27], [28], optimization [12], [29]–[31], and machine learning [10], [22]. The requirements, however, of such *instance-level 3D models*, either for training offline or model-frame registration during tracking, complicate generalization to novel instances. More recently, a 6D pose tracking approach [17] relaxed the assumption to *category-level 3D models* using 3D object CAD model databases for training [18]. During testing, the target object category needs to be identified and the corresponding network for that category is utilized for tracking. Instead of being limited to the number of categories such database is able to include, this work employs deep features that in principle can be trained on arbitrary 2D images. It allows generalization to diverse novel objects, as shown in the accompanying experiments.

Dynamic Object-aware SLAM - In order to track dynamic objects' pose and decouple them from static background, frame-model Iterative Closest Point *(ICP)* combined with color [23]–[26], probabilistic data association [32], or 3D level-set likelihood maximization [33] has been applied. Object models are simultaneously reconstructed on-the-fly by aggregating the observed RGB-D data with the newly tracked pose. Nevertheless, frame-model tracking can be challenging for object reconstruction, since errors in pose estimation transfer to the reconstructed model and adversely affect the subsequent tracking [34]. This work does not fuse observed frames but instead maintains them as nodes in a pose graph, allowing to correct previously erroneous estimates, and reduces drift in long-term tracking. The aforementioned SLAM-family approaches may also face challenges in robot manipulation setups that involve small, textureless, flat or shiny objects due to the dearth of sufficient correspondences between the pair of consecutive frames. To ameliorate this issue, *BundleTrack* searches correspondences among current and multiple historical frames, consisting of both feature and geometric terms, as the edges in the pose graph. Its effectiveness has been shown in extensive experiments including for such challenging manipulation scenarios.

3D Hand-held Object Scanning - Promising results have been demonstrated in scanning dynamic hand-held objects [35]–[39], where the object's motion needs to be taken into account similar to the current setup. In particular, a framework for robot manipulation [37] performs simultaneous object reconstruction and tracking, which leads to similar issues as the aforementioned dynamic SLAM methods. In addition, forward kinematics is required in its Kalman Filtering framework, preventing generalization in scenarios when objects are not held by the robotic manipulator. While estimating object poses is part of the scanning process, there are key differences from online 6D pose tracking. For the scanning application, external assistance including human interaction or deliberate motion is acceptable [36], [38], [39] but it is not assumed in the current work. Furthermore, time consuming global-optimization steps are often adopted at the end of scanning to polish the models and their poses while intermediate erroneous pose estimations and associated frames can be discarded and not fused into the global model [36], [38], [39]. In contrast, this work aims to provide fast and accurate pose tracking output online.

III. PROBLEM FORMULATION



Assume a rigid body for which there is no its corresponding 3D model, nor its category-level 3D model database for training. The objective is to continuously track its 6D pose change relative to the start of tracking, i.e., the relative transformation $T_{0\to\tau} \in SE(3), \tau \in \{1,2,...,t\}$ in the camera's frame C. The input is the following:

- I_τ: A sequence of RGB-D data $I_\tau, \tau \in \{0,...,t\}$.
- M_0: A binary mask on the first image I_0, indicating the target object region to track in the image space.
- T_0^C (optional): The initial pose in the camera's frame C. Used if the objective is to recover the object's absolute pose in C, otherwise set to identity.

The initial mask M_0 can be obtained in multiple different ways to initialize tracking. For instance, via semantic segmentation [40]–[42] or non-semantic methods, such as image

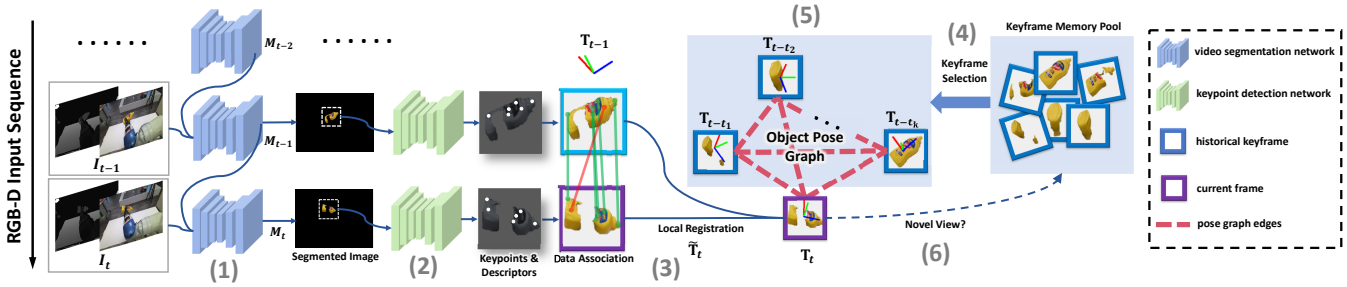


Fig. 2: *BundleTrack* framework from left to right: (1) an image segmentation network returns the object mask given the prior one; (2) a network detects keypoints and their descriptors; (3) keypoints are matched and coarse registration is performed between consecutive frames to estimate an initial relative transform $\tilde{\mathbf{T}}_t$; (4) keyframes are selected from a memory pool to participate in the pose graph optimization; (5) online pose graph optimization outputs a refined spatiotemporal consistent pose $\mathbf{T}_t$; and (6) the latest frame is included in the memory pool, if it is a novel view to enrich diversity.

segmentation, [43]–[45], point cloud segmentation/clustering [46], [47], or plane fitting and removal [46], etc.

The object's pose in the camera's frame C can be recovered at any timestamp by applying the relative transformation $T_{0\to\tau}$ in the camera's frame $\mathbf{T}_\tau = T_\tau^C = T_0^C[(T_0^C)^{-1}T_{0\to\tau}T_0^C] = T_{0\to\tau}T_0^C \in SE(3)$. For simplicity, the rest of this document will refer to $\mathbf{T}_\tau$ as the output of the process but $T_{0\to\tau}$ is what is actually computed as tracking.

IV. APPROACH

An overview of the proposed *BundleTrack* framework is depicted in Fig. 2. The currently observed RGB-D frame I_t and the object segmentation mask computed during the last timestamp M_{t-1} are forwarded to a video segmentation network to compute the current object mask M_t. Based on M_t and M_{t-1} respectively, the target object regions in both I_t and I_{t-1} are cropped, resized and sent to a keypoint detection network to compute keypoints and feature descriptors. A data association process consisting of feature matching and outlier pruning in the manner of *RANSAC* [48] identifies feature correspondences. Based on these correspondences, a registration between I_{t-1} and I_t can be solved in closed-form, which is then used to provide a coarse estimate $\tilde{\mathbf{T}}_t$ for the transform between the two snapshots. The estimate $\tilde{\mathbf{T}}_t$ is used to initialize the current node $\mathbf{T}_t$ as part of a pose graph optimization step. To define the rest of the nodes of the pose graph, no more than $\mathcal{K}$ keyframes are selected from a memory pool to participate in the optimization. The choice of $\mathcal{K}$ is made to balance an efficiency vs. accuracy tradeoff. Pose graph edges include both feature and geometric correspondences, which are computed in parallel on GPU. Given this information, the pose graph step outputs online the optimized pose for the current timestamp $\mathbf{T}_t \in SE(3)$. If the last frame corresponds to a novel view, then it is also included in the memory pool.

A. Propagating Object Segmentation

The first step is to segment the object's image region from the background. Prior work [24] used *Mask-RCNN* [49] to compute the object mask in every frame of the video. It deals with each new frame independently, which is less efficient and results in temporal inconsistencies.

To avoid these limitations, this work adopts an off-the-shelf *transductive-VOS* network [50] for video object segmentation, which is trained on the *Davis 2017* [51] and *Youtube-VOS* [52] datasets. The network uses dense long-term similarity dependencies between current and past feature embeddings to propagate the previous object mask to the latest frame. The object mask needed by *BundleTrack* is simply binary, i.e., $M_\tau = \{0,1\}^{H\times W}, \tau \in \{0,1,...,t\}$ and distinguishes the object region from the background. The only requirement is an initial mask M_0 of interest. Neither the *transductive-VOS* network nor the following steps of *BundleTrack* require M_0 to come from semantic/instance segmentation. Therefore, it can also be obtained in alternative ways depending on the application, e.g., low-level image segmentation [43], [53], point cloud segmentation/clustering [46], [47], or plane fitting and removal [46], etc.

While the current implementation uses *transductive-VOS*, the following techniques do not depend on this specific network. If the object mask can be computed via simpler means, such as computing a region of interest (ROI) from forward kinematics followed by point cloud filtering in robot manipulation scenarios [2], the segmentation module can be replaced.

B. Keypoint Detection, Matching and Local Registration

Local registration is performed between consecutive frames I_{t-1} and I_t to compute a initial pose $\tilde{\mathbf{T}}_t$. To do so, correspondence between keyframes detected on each image is performed. Different from prior work [17], which relies on *category-level 3D models* to learn a fixed number of category-level semantic keypoints, this work aims to use generalizable features not specific to certain instances or categories. The *LF-Net* [54] is chosen given its satisfactory balance between performance and inference speed. It only requires training on general 2D images, such as the *ScanNet dataset* [55] used here, and generalizes to novel scenes. During testing, for the newly observed frame I_t, *LF-Net* receives the segmented image (Sec. IV-A) as input. It then outputs n keypoints $x_i, i \in \{0,1,...,n-1\}$ along with the feature descriptor $D_i \in R^{128}$, where n is 500 in all experiments. Due to the potentially imperfect segmentation in previous step, outlier keypoints can arise from the background. It is thus critical to perform feature matching and outlier pruning via *RANSAC* [48], executed in parallel on GPU in this work. Each registration sample consists of 3 pairs of keypoints matched between the two images. A pose hypothesis is generated from a sample via least squares [56]. When evaluating samples, inlier correspondences have

a distance between transformed point pairs below a threshold δ and an angle formed by the normals within a threshold α. The values of δ and α are empirically set to $5mm$ and 45° in all experiments. After *RANSAC*, a preliminary pose is computed by $\tilde{\mathbf{T}}_t = \mathbf{T}_{t-1} T_t^{t-1}$ where T_t^{t-1} is the best sampled correspondence hypothesis.

C. Keyframe Selection

$\tilde{\mathbf{T}}_t$ is then refined during a pose graph optimization step. The number of keyframes participating in the optimization is limited to $k \leqslant \mathcal{K}$ for the sake of efficiency, where $\mathcal{K} = 15$ is the number used in the experiments. When the size of the keyframe memory pool $\mathcal{N}$ is larger than $\mathcal{K}$, the objective is to find the set of keyframes with the largest mutual viewing overlap to make good use of multi-view consistency. This challenge can be formulated as the minimum H-subgraph of an edge-weighted graph problem [57]:

$$\underset{x}{\operatorname{argmin}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}, j \neq i} x_i x_j \cdot arccos\left(\frac{tr(R_i^T R_j) - 1}{2}\right)$$
$$\text{so that}: \sum_{i \in \mathcal{N}} x_i = \mathcal{K} \text{ and } x_i \in \{0,1\}, i \in \mathcal{N},$$

where R_i is the rotation matrix of the corresponding keyframe's pose. The goal is to find the optimal binary vector $x \in R^N$ that indicates the selections. The weight of the edge between frame pair (i, j) is the geodesic distance of their rotations. Mutual viewing overlap is maximized when the mutual rotation difference relative to the camera is minimized. Combinatorial optimization algorithms for solving this problem have a complexity of $O(\mathcal{N}^{\mathcal{K}}/log\mathcal{N})$ [57]. In practice, an iterative greedy selection is followed by starting with the keyframe set $\{I_0\}$ until the number of selected keyframes reaches $\mathcal{K}$. I_0 is chosen since the initial frame does not suffer from any tracking drift and serves as the reference frame. In each iteration, the keyframe with the smallest sum of geodesic distances against I_t as well as all previously selected keyframes is added. This reduces complexity to $O(\mathcal{N}\mathcal{K}^3 + \mathcal{N}\mathcal{K}^2)$, making the selection practical (under a millisecond) without degrading performance.

D. Online Pose Graph Optimization

The pose graph can be denoted as $G = \{V, E\}, |V| = k + 1$, where each node corresponds to the object pose in the camera's frame at the current and k selected timestamps $\tau \in \{t, t - t_1, t - t_2, ..., t - t_k\}$. For simplicity, the subscripts of graph nodes will be denoted as simple indices $i \in |V|$ instead of the actual timestamp $t - t_i$. Each node's pose can then be denoted as $\mathbf{T}_i, i \in |V|$. Inspired by [58], for the edges between each pair of nodes, two types of energies $\mathbf{E}_f$ and $\mathbf{E}_g$ are considered. The energy $\mathbf{E}_f$ relates to the residuals computed from feature correspondences and $\mathbf{E}_g$ relates to the geometric residuals measured by dense pixel-wise point-to-plane distance. The spatiotemporal consistency is achieved when the total energy of the graph $\mathbf{E}$ is minimized:

$$\mathbf{E} = \sum_{i \in |V|} \sum_{j \in |V|, j \neq i} (\lambda_1 \mathbf{E}_f(i,j) + \lambda_2 \mathbf{E}_g(i,j)) \tag{1}$$

$$\mathbf{E}_f(i,j) = \sum_{(m,n) \in C_{i,j}} \rho\left(\left\| \mathbf{T}_i^{-1} p_m - \mathbf{T}_j^{-1} p_n \right\|_2\right) \tag{2}$$

In order to compute $\mathbf{E}_f$, feature correspondences $C_{i,j}$ between each pair of nodes (i, j) are determined. If $C_{i,j}$ has been built during a previous pose graph optimization, it is reused. Otherwise, the data association process of Sec. IV-B is performed to compute $C_{i,j}$. These multi-pair feature correspondences are built in parallel on GPU. In Eq. (2) and (3), p represents the unprojected 3D points in the camera's frame, ρ is the M-estimator, where Huber loss is used.

$$\mathbf{E}_g(i,j) = \sum_{p \in |I_i|} \rho\left(\left\| n_i(x) \cdot (\mathbf{T}_i \mathbf{T}_j^{-1} \pi_D^{-1}(\pi(\mathbf{T}_j \mathbf{T}_i^{-1} p)) - p) \right\|_2\right) \tag{3}$$

For $\mathbf{E}_g$, dense pixel-wise correspondences are associated by point re-projection, while outliers are filtered based on the distance between the point pair and the angle formed by their normals; $\pi(\cdot)$ is the perspective projection operation; $\pi_D^{-1}(\cdot)$ denotes the unprojection mapping, which recovers a 3D point in the camera's frame by looking up the depth value on the pixel location; $n_i(\cdot)$ returns the normal of the pixel on the frame $I_i, i \in |V|$.

In Eq. (1), λ_1 and λ_2 are the weights balancing $\mathbf{E}_f$ and $\mathbf{E}_g$. To emphasize the lack of sensitivity to the choice of these values, λ_1 and λ_2 are set to 1 in all experiments unless otherwise specified. Then, the goal is to find the optimal poses, such that:

$$\xi^* = \underset{\xi}{\operatorname{argmin}} \rho(\bar{\mathbf{E}}(\xi))$$

where $\bar{\mathbf{E}}(\xi)$ is the stacked energy residual vector, $\xi = (\xi_t, \xi_{t-t_1}, \xi_{t-t_2}, ..., \xi_{t-t_k})^T \in R^{6 \times (k+1)}$ is the stacked pose vector corresponding to the current frame and k selected past keyframes, while the pose corresponding to the initial frame I_0 is kept constant as reference. Each block $\xi_i = log(\mathbf{T}_i) \in \mathfrak{se}(3)$ is parametrized in Lie Algebra [59], consisting of 3 parameters for translation and 3 parameters for rotation. A common approach is to apply first-order Taylor expansion around ξ, such that the iteratively re-weighted nonlinear least squares can be solved by a Gauss-Newton update:

$$(\mathbf{J}^T \mathbf{W} \mathbf{J}) \Delta\xi = \mathbf{J}^T \mathbf{W} \bar{\mathbf{E}}$$

where $\mathbf{J}$ is the Jacobian matrix with respect to ξ, $\mathbf{W}$ is a diagonal weight matrix computed by the M-estimator ρ and residual, which is updated in each iteration. To better take advantage of the sparsity of $\mathbf{J}$ and $\mathbf{W}$, inside each Gauss-Newton step, an iterative PCG (Preconditioned Conjugate Gradient) [60] solver is leveraged, where the diagonal matrix $\mathbf{J}^T \mathbf{W} \mathbf{J}$ is used as the preconditioner. Incremental pose updates are accumulated in the tangent space after each iteration $\xi \leftarrow \xi \boxplus \Delta\xi$. The entire pose graph optimization is implemented in CUDA for parallel computation.

At the end of the optimization, the object pose corresponding to each graph node is obtained by $\mathbf{T}_i = exp(\xi_i) \in SE(3), i \in |V|$. The one corresponding to the current timestamp t becomes the output tracked pose $\mathbf{T}_t$, while poses corresponding to the historical keyframes are updated in the memory pool. The entire process is causal, i.e. past frames' corrected poses cannot be updated in the output. However, their corrected pose estimates provide better initialization in following pose graph optimization steps to benefit the solution of new observations. This significantly reduces long-

term drift compared against tracking-via-reconstruction [24], where any intermediate erroneous pose estimation introduces noise when fused into the global model and adversely affects the subsequent tracking.

E. Augmenting the Keyframe Memory Pool

The initial frame I_0 is always selected as it does not suffer from any tracking drift. For later frames, once the current object pose $\mathbf{T}_t$ is determined, its rotation geodesic distance against each existing keyframe in the pool is compared. If all pair-wise distances are larger than α ($arccos(10°)$ in all experiments), I_t is added into the keyframe memory pool. This encourages to add frames from novel views, such that multi-view diversity is enriched.

V. EXPERIMENTS

This section evaluates the proposed approach and compares against state-of-the-art 6D pose tracking and estimation methods on two public benchmarks, the *NOCS dataset* [13] and the *YCBInEOAT dataset* [22]. Experiments are performed over diverse types of objects and various tracking scenarios (e.g., moving camera or moving objects). Both quantitative and qualitative results demonstrate that *BundleTrack* achieves comparable or even superior performance relative to alternatives, although it does not require *instance or category-level 3D models*. Concretely, no CAD models or training data from a 3D object database are used by *BundleTrack*. All experiments are conducted on a standard desktop with Intel Xeon(R) E5-1660 v3@3.00GHz processor and a single NVIDIA RTX 2080 Ti GPU.

A. Datasets

NOCS dataset [13]: Among existing datasets, this is the closest to the setup here, where *instance 3D models* are not provided during evaluation. The dataset contains 6 object categories: bottle, bowl, camera, can, laptop, and mug. The training set consists of: (1) 7 real videos containing 3 instances of each category in total, annotated with ground truth poses; and (2) 275K frames of synthetic data generated using 1085 instances from the above 6 categories using a 3D model database *ShapeNetCore* [18] with random poses and object combinations in each scene. The testing set has 6 real videos containing 3 different unseen instances within each category, resulting in 18 different object instances and 3,200 frames in total.

YCBInEOAT dataset [22]: This dataset helps verify the effectiveness of 6D pose tracking during robot manipulation. It was originally developed to evaluate approaches relying on CAD models. The available CAD models, however, are not used by *BundleTrack*. In contrast to the *NOCS dataset* where objects are statically placed on a tabletop and captured by a moving camera, *YCBInEOAT* contains 9 video sequences captured by a static RGB-D camera, while objects are dynamically manipulated. There are three types of manipulation: (1) single arm pick-and-place, (2) within-hand manipulation, and (3) pick to hand-off between arms to placement. These scenarios and the end-effectors used make directly computing poses from forward kinematics unreliable. The manipulation videos involve 5 *YCB Objects* [61]: mustard bottle, tomato soup can, sugar box, bleach cleanser and cracker box.

Assumption	Methods	Metrics	bottle	bowl	camera	can	laptop	mug	Overall
Category -Level 3D Model	NOCS [13]	5°5cm	5.5	**62.2**	0.6	7.1	25.5	0.9	17.0
		IoU25	48.7	99.6	90.6	77.0	94.7	82.8	82.2
		R_{err}	25.6	**4.7**	**33.8**	16.9	8.6	31.5	20.2
		T_{err}	14.4	**1.2**	**3.1**	**4.0**	**2.4**	4.0	4.9
	KeypointNet [62]	5°5cm	5.9	16.8	1.8	4.3	49.2	3.1	13.5
		IoU25	23.1	74.7	30.9	42.6	94.6	52.0	53.0
		R_{err}	28.5	9.8	45.2	28.8	6.5	61.2	30.0
		T_{err}	9.5	8.2	8.5	13.1	4.4	6.7	8.4
	6-PACK w/o temporal [17]	5°5cm	23.7	53.0	8.4	**25.0**	62.4	22.4	32.5
		IoU25	**92.0**	**100.0**	**91.0**	89.9	97.8	**100.0**	**95.1**
		R_{err}	15.7	5.3	43.9	**12.5**	4.9	**20.3**	17.1
		T_{err}	4.2	1.6	5.5	5.0	2.5	**1.8**	**3.4**
	6-PACK [17]	5°5cm	**24.5**	55.0	**10.1**	22.6	**63.5**	**24.1**	**33.3**
		IoU25	91.1	**100.0**	87.6	**92.6**	**98.1**	95.2	94.2
		R_{err}	**15.6**	5.2	35.7	13.9	**4.7**	21.3	**16.0**
		T_{err}	**4.0**	1.7	5.6	4.8	2.5	2.3	3.5
No Model	ICP [63]	5°5cm	10.1	40.3	12.6	17.2	14.8	6.2	16.9
		IoU25	29.9	79.7	53.1	40.5	50.9	27.7	47.0
		R_{err}	48.0	19.0	80.5	47.1	37.7	56.3	48.1
		T_{err}	15.7	4.7	12.2	9.4	9.2	9.2	10.5
	TEASER++* [64]	5°5cm	13.9	35.5	10.7	11.7	40.9	7.5	20.0
		IoU25	**100.0**	**99.9**	**99.9**	**100.0**	**99.9**	**99.9**	**99.9**
		R_{err}	17.0	10.6	18.8	20.4	7.2	23.0	16.2
		T_{err}	2.7	1.8	2.8	2.7	2.6	2.4	2.5
	MaskFusion [24]	5°5cm	15.5	32.3	11.7	8.8	73.9	16.4	26.5
		IoU25	51.4	71.4	60.8	49.7	**99.9**	56.2	64.9
		R_{err}	36.7	12.3	43.0	34.9	3.4	40.6	28.5
		T_{err}	11.3	5.3	11.1	9.3	3.5	9.2	8.3
	BundleTrack (Ours)	5°5cm	**86.5**	**99.6**	**85.8**	**99.2**	**99.9**	**53.6**	**87.4**
		IoU25	**100.0**	**99.9**	**99.9**	**100.0**	**99.9**	**99.9**	**99.9**
		R_{err}	**1.6**	**1.7**	**3.0**	**1.5**	**1.5**	**5.2**	**2.4**
		T_{err}	**2.3**	**2.1**	**2.1**	**2.1**	**2.2**	**2.2**	**2.1**

TABLE I: Results on the *NOCS dataset* [13]. For the metrics of 5°5cm and IoU25, a higher value is preferable. For the metrics of R_{err} and T_{err}, a lower value is preferable. Under each type of 3D model assumption, the best results are highlighted in bold font. TEASER++* denotes TEASER++ [64] operating over the same segmented point cloud and feature correspondences as in the proposed *BundleTrack*.

B. Results on the NOCS Dataset

Table I and Fig. 3 present the quantitative and qualitative results of state-of-art methods on the *NOCS dataset* respectively. The comparison points include learning-based methods relying on a *category-level prior*, such as *NOCS* [13], *KeypointNet* [62], and *6-PACK* with or without temporal prediction [17]. These methods are offline trained on both real and synthetic training sets, which are rendered with 3D object models extracted from the same categories of *ShapeNetCore* [18]. In contrast, *ICP* [63], *MaskFusion* [24], *TEASER++** [64] and the proposed *BundleTrack* have no access to any training data based on 3D models.

The evaluation protocol is the same as in prior work [17]. A perturbed ground-truth object pose is used for initialization. The perturbation adds a uniformly sampled random translation within a 4cm range to evaluate robustness against a noisy initial pose [17]. No re-initialization is allowed during tracking. To evaluate robustness against missing frames, the same uniformly sampled 450 frames out of 3200 in the testing videos are dropped [17]. Four metrics are adopted: 1) **5°5cm**: percentage of estimates with orientation error < 5°and translation error < 5cm - the higher the better; 2) **IoU25** (Intersection over Union): percentage of cases where the overlapping prediction and ground-truth 3D bounding box volume is larger than 25% of their union - the higher the better; 3) $\mathbf{R_{err}}$: mean orientation error in degrees - the lower the better; and 4) $\mathbf{T_{err}}$: mean translation error in centimeters - the lower the better. For R_{err} and T_{err}, estimates with IoU≤25 are not counted when computing averages[1] [17].

[1]https://github.com/j96w/6-PACK/blob/master/benchmark.py

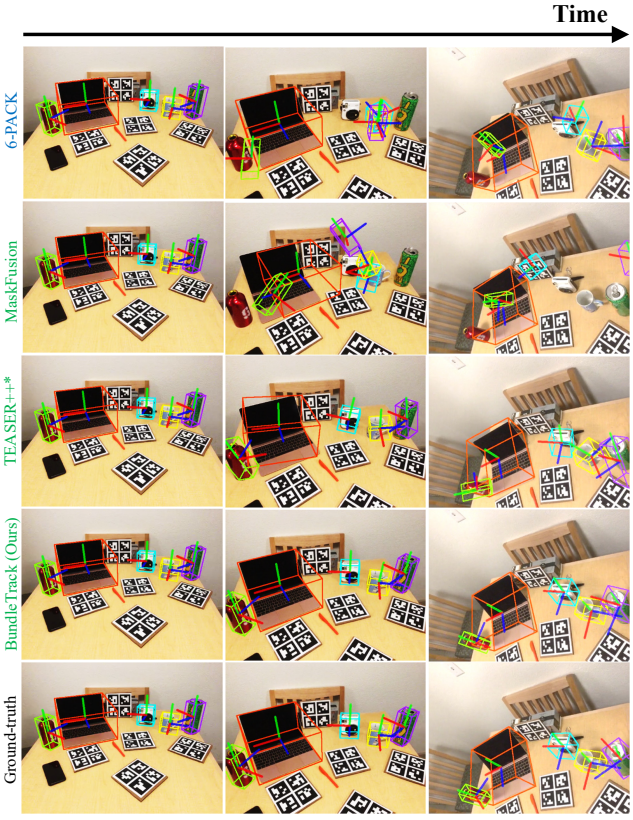


Fig. 3: Example qualitative results of *BundleTrack* and representative comparison points on *NOCS Dataset*. In all methods, each object is tracked individually and depicted in the same image for visualization. Methods' names are colored in blue and green to denote assumption on *category-level 3D model* and *no model* respectively. For more qualitative results, please refer to the supplementary video.

The results of comparison points other than *MaskFusion* and *TEASER++*[*] come from the literature [17]. The open-sourced code[2] of *MaskFusion* is used for evaluation, where the global SLAM module is disabled to avoid inferring object poses from the camera's estimated ego-motion. The dynamic object tracking module is kept to solely evaluate object pose tracking effectiveness. Its original segmentation module *Mask-RCNN* [49] is fine-tuned on the real training data provided in the *NOCS dataset* for better performance while the synthetic data rendered using category-level 3D models are not used, as this method is also agnostic to any 3D models [24]. In addition to *ICP* reported in [17], another state-of-art 3D registration approach [64] is included for comparison and denoted as *TEASER++*[*], which is robust to outlier correspondences and agnostic to 3D models. It takes as input the segmented point cloud and feature correspondences that are computed using the same modules proposed in *BundleTrack*. For *BundleTrack*, an initial mask M_0 is required as input to the framework and is provided via the aforementioned *Mask-RCNN*. During execution, *BundleTrack* does not require external mask input nor any form of re-initialization. As exhibited in Table I, *BundleTrack* significantly outperforms the comparison points under all metrics and over all object categories, despite not accessing *instance or category-level 3D models*.

[2]https://github.com/martinruenz/maskfusion

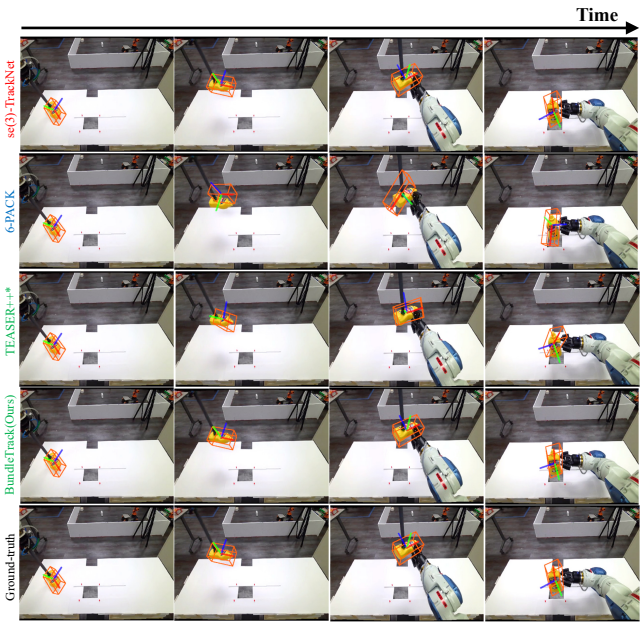


Fig. 4: Example qualitative results of *BundleTrack* and representative comparison points on *YCBInEOAT Dataset*. Methods' names are colored in red, blue and green to denote assumption on *instance 3D model*, *category-level 3D model* and *no model* respectively. For more qualitative results, please refer to the supplementary video.

C. Results on YCBInEOAT Dataset

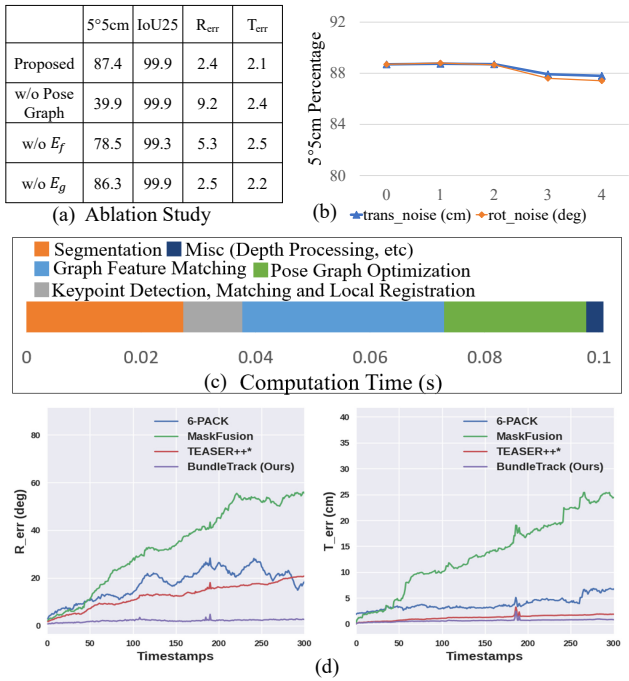
Evaluation exclusively on static objects captured by a moving camera cannot completely reflect the properties of a 6D pose tracking method [22]. For this reason, the *YCBInEOAT dataset* is chosen to evaluate tracking in scenarios where objects are moving in front of the camera. The same evaluation protocol is followed as in prior work [22]. Results are computed from accuracy-threshold AUC (Area Under Curve) measured by $ADD = \frac{1}{m}\sum_{x\in M} ||Rx+T-(\hat{R}x+\hat{T})||$, which performs exact model matching, and $ADD\text{-}S = \frac{1}{m}\sum_{x_1\in M}\min_{x_2\in M}||Rx_1+T-(\hat{R}x_2+\hat{T})||$ [3] designed for evaluating symmetric objects. Similar to prior work [22], the ground-truth object's pose in the camera's frame is provided as initialization. No re-initialization is allowed during the tracking process.

Quantitative and qualitative results are shown in Table II and Fig. 4 respectively. Comparison points include state-of-art 6D pose tracking methods that use object CAD models, such as *RGF* [28], *dbot PF* [11] and *se(3)-TrackNet* [22]. *6-PACK* [17] is a state-of-art 6D pose tracking approach relying on *category-level 3D models*. Its evaluation on objects "021_bleach_cleanser", "006_mustard_bottle" and "005_tomato_soup_can" are performed by using the officially released[3] networks trained on "bottle" and "can" category respectively . For the rest of the objects "003_cracker_box" and "004_sugar_box", no suitable corresponding category can be found in existing 3D model database [18] and thus *6-PACK* is not able to be retrained and evaluated on them. For *6-PACK*, 3D bounding box of the object model, computed from forward kinematics, is provided in every frame to crop ROI from point cloud, since it is more reliable than its default module of extrapolating the 3D bounding box by estimated motion. For *MaskFusion* [24] and *BundleTrack*,

[3]https://github.com/j96w/6-PACK

Assumption	Methods	003_cracker_box		021_bleach_cleanser		004_sugar_box		005_tomato_soup_can		006_mustard_bottle		ALL	
		ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S
Instance 3D Model	RGF [28]	34.78	55.44	29.40	45.03	15.82	16.87	15.13	26.44	56.49	60.17	29.98	39.90
	dbot PF [11]	79.00	88.13	61.47	68.96	86.78	92.75	63.71	93.17	91.31	95.31	78.28	89.18
	se(3)-TrackNet [22]	**90.76**	**94.06**	**89.58**	**94.44**	**92.43**	**94.80**	**93.40**	**96.95**	**97.00**	**97.92**	**92.66**	**95.53**
Category-Level 3D Model	6-PACK [17]	-	-	4.18	18.00	-	-	12.82	60.32	34.49	80.76	-	-
No Model	MaskFusion [24]	79.74	88.28	29.83	43.31	36.18	45.62	5.65	6.45	11.55	13.11	35.07	41.88
	TEASER++* [64]	63.24	81.35	61.83	82.45	51.91	81.42	41.36	71.61	71.92	88.53	57.91	81.17
	BundleTrack (Ours)	**85.07**	**89.41**	**89.34**	**94.72**	**85.56**	**90.22**	**86.00**	**95.13**	**92.26**	**95.35**	**87.34**	**92.53**

TABLE II: Results of AUC measured by ADD and ADD-S metrics on *YCBInEOAT Dataset* [22]. Under each type of 3D model assumption, best results are in bold. TEASER++* denotes TEASER++ [64] operating over the same segmented point cloud and feature correspondences as in the proposed *BundleTrack*.



	5°5cm	IoU25	R_{err}	T_{err}
Proposed	87.4	99.9	2.4	2.1
w/o Pose Graph	39.9	99.9	9.2	2.4
w/o E_f	78.5	99.3	5.3	2.5
w/o E_g	86.3	99.9	2.5	2.2

(a) Ablation Study (b) (c) Computation Time (s) (d)

Fig. 5: Experimental analysis performed on *NOCS dataset* as described in Sec V-D. (a) Ablation study investigating effectiveness of pose graph optimization and each energy term. (b) Sensitivity of *BundleTrack* to inaccurate initial pose by deliberately introducing different translation and rotation noise levels. (c) Average running time decomposition of different modules. (d) Rotation and translation error w.r.t. timestamps compared against representative related works [17], [24], [64] for tracking drift study.

the initial object mask is obtained by table fitting and removal, followed by Euclidean Clustering implemented in PCL [46]. The original *MaskFusion*'s segmentation module *Mask-RCNN* cannot be retrained on this benchmark due to the lack of training set. Therefore, during tracking, the target object mask is computed by segmenting out the region of robot arm and end-effector from forward kinematics. For instances of irregular shapes or colors ("021_bleach_cleanser", "006_mustard_bottle") within the "bottle" category that *6-PACK* has been trained on, it struggles to get satisfactory result. Nevertheless, *BundleTrack* consistently demonstrates high quality tracking without any retraining or fine-tuning. This establishes generalizability of *BundleTrack* to novel object instances regardless of their out-of-distribution properties within the category. *BundleTrack* also achieves comparable or superior performance even when compared against methods relying on object instance CAD models [11], [22], [28].

D. Analysis

Ablations Study: An ablation study investigates the effectiveness of the online global pose graph optimization and each energy term, presented in Fig. 5 (a).

Sensitivity to Initial Pose: As mentioned, random translation noise within 4cm range is added to the initial pose. This part further investigates robustness under different translation and rotation noise levels, shown in Fig. 5 (b).

Computation Time: The average running time of modules are given in Fig. 5 (c). The entire framework runs at 10Hz on average including video segmentation. The *6-PACK* [17], *TEASER++** [64] and *MaskFusion* [24] methods from related work run at 4Hz, 11Hz and 17Hz respectively on the same machine.

Tracking Drift Analysis: Fig. 5 (d) presents the rotation and translation error w.r.t. timestamps compared against representative related works [17], [24], [64]. Results are averaged across all videos on the *NOCS Dataset*.

Generalization: The neural networks' weights and hyperparameters in *BundleTrack* are fixed without any retraining or fine-tuning across all evaluations (Sec. V-B, V-C). When applied to novel instances, the framework does not require access to *instance or category-level 3D models* for training or registration.

Failure Cases: While *BundleTrack* is able to robustly keep tracking in all experiments without lost or re-initialization, intermediate imprecise estimates are observed, such as the cases illustrated in Fig. 6.

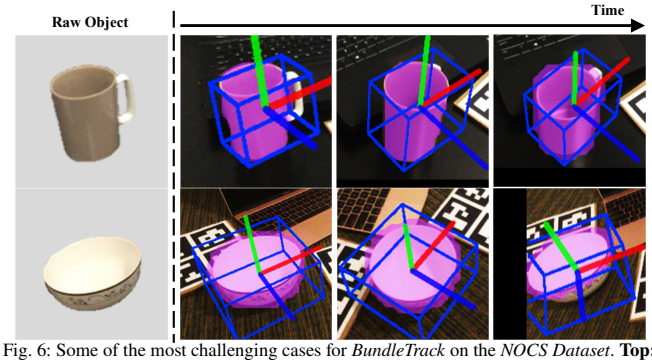


Fig. 6: Some of the most challenging cases for *BundleTrack* on the *NOCS Dataset*. **Top:** Severe self-occlusion prevents data association around the mug's handle, introducing challenges for solving the orientation around the green axis. Nevertheless, with better visibility in subsequent frames, *BundleTrack* is able to recover from drifts and continue tracking, thanks to the memory-augmented pose graph optimization. **Bottom:** Near the end of video, noisy segmentation (purple mask) falsely ignores the side of the bowl, preventing relevant feature extraction and leads to slight translation offset. With future development of more advanced segmentation module, the overall tracking performance is expected to be boosted.

VI. CONCLUSION

This work presents *BundleTrack*, a general framework for tracking the 6D pose of novel objects without any assumptions on *instance or category-level 3D models*. Extensive experiments demonstrate that it is able to perform long-term accurate tracking under various challenging scenarios. It even

achieves comparable performance to state-of-art methods that depend on the target object's CAD model. Future research includes the exploration of combining *BundleTrack* with model-free grasping methods [65], [66], to perform robust pick-and-place [67], [68] or in-hand dexterous manipulation for a wide variety of novel objects.

REFERENCES

[1] D. Kappler and et al, "Real-time perception meets reactive motion generation," *IEEE RAL*, 2018.

[2] B. Wen *et al.*, "Robust, occlusion-aware pose estimation for objects grasped by adaptive hands," *ICRA*, 2020.

[3] Y. Xiang and et al., "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *RSS*, 2018.

[4] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *CVPR*, 2019.

[5] Z. Li and et al., "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation," in *ICCV 2019*.

[6] Y. He and et al., "Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation," in *CVPR*, 2020, pp. 11 632–11 641.

[7] C. Mitash and et al, "Scene-level pose estimation for multiple instances of densely packed objects," in *CoRL*, 2020.

[8] J. Tremblay and et al., "Deep object pose estimation for semantic robotic grasping of household objects," *CoRL*, 2018.

[9] C. Wang and et al., "Densefusion: 6d object pose estimation by iterative dense fusion," *CVPR*, 2019.

[10] X. Deng, A. Mousavian, Y. Xiang, and et al., "Poserbpf: A rao-blackwellized particle filter for 6d object pose tracking," *RSS*, 2019.

[11] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," *IROS 2013*.

[12] T. Schmidt, R. A. Newcombe, and D. Fox, "Dart: Dense articulated real-time tracking." in *RSS 2014*.

[13] H. Wang and et al., "Normalized object coordinate space for category-level 6d object pose and size estimation," in *CVPR*, 2019.

[14] K. Park and et al., "Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation," in *CVPR*, 2020.

[15] X. Chen and et al., "Category level object pose estimation via neural analysis-by-synthesis," *arXiv preprint arXiv:2008.08145*, 2020.

[16] D. Chen, J. Li, Z. Wang, and K. Xu, "Learning canonical shape space for category-level 6d object pose and size estimation," in *CVPR*, 2020.

[17] C. Wang and et al., "6-pack: Category-level 6d pose tracker with anchor-based keypoints," in *ICRA*, 2020.

[18] A. X. Chang and et al., "Shapenet: An information-rich 3d model repository," *arXiv:1512.03012*, 2015.

[19] Z. Wu and et al., "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015.

[20] R. A. Newcombe and et al, "Kinectfusion: Real-time dense surface mapping and tracking," in *ISMAR*, 2011.

[21] P. Cignoni and et al, "Meshlab: an open-source mesh processing tool." in *Eurographics Italian chapter conference*, 2008.

[22] B. Wen and et al, "se (3)-tracknet: Data-driven 6d pose tracking by calibrating image residuals in synthetic domains," in *IROS*, 2020.

[23] B. Xu and et al., "Mid-fusion: Octree-based object-level multi-instance dynamic slam," in *ICRA*, 2019.

[24] M. Runz and et al., "Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *ISMAR*, 2018.

[25] L. Ma *et al.*, "Simultaneous localization, mapping, and manipulation for unsupervised object discovery," in *ICRA*, 2015.

[26] M. Rünz and L. Agapito, "Co-fusion: Real-time segmentation, tracking and fusion of multiple objects," in *ICRA*. IEEE, 2017, pp. 4471–4478.

[27] C. Choi and H. I. Christensen, "Rgb-d object tracking: A particle filter approach on gpu," in *IROS*, 2013.

[28] J. Issac and et al., "Depth-based object tracking using a robust gaussian filter," in *ICRA*, 2016.

[29] D. Joseph Tan, F. Tombari, S. Ilic, and N. Navab, "A versatile learning-based 3d temporal tracker: Scalable, robust, online," in *ICCV 2015*.

[30] L. Zhong and L. Zhang, "A robust monocular 3d object tracking method combining statistical and photometric constraints," *IJCV 2019*.

[31] H. Tjaden and et al., "A region-based gauss-newton approach to real-time monocular multiple object tracking," *TPAMI 2018*.

[32] M. Strecke and J. Stuckler, "Em-fusion: Dynamic object-level slam with probabilistic data association," in *CVPR*, 2019.

[33] Y. Ren and et al, "Star3d: Simultaneous tracking and reconstruction of 3d objects using rgb-d data," in *CVPR*, 2013.

[34] M. Slavcheva and et al., "Sdf-2-sdf: Highly accurate 3d object reconstruction," in *ECCV*, 2016.

[35] D. Tzionas and J. Gall, "3d object reconstruction from hand-object interactions," in *ICCV*, 2015, pp. 729–737.

[36] T. Weise *et al.*, "Online loop closure for real-time interactive 3d scanning," *Computer Vision and Image Understanding*, 2011.

[37] M. Krainin and et al., "Manipulator and object tracking for in hand model acquisition," in *ICRA*, 2010.

[38] F. Wang and K. Hauser, "In-hand object scanning via rgb-d video segmentation," in *ICRA*, 2019.

[39] T. Weise, B. Leibe, and L. Van Gool, "Accurate and robust registration for in-hand modeling," in *CVPR*. IEEE, 2008, pp. 1–8.

[40] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[41] L.-C. Chen and et al., "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *TPAMI*, 2017.

[42] T. H. N. Le and et al, "Deep contextual recurrent residual networks for scene labeling," *Pattern Recognition*, 2018.

[43] F. Meyer, "Color image segmentation," in *International Conference on Image Processing and its Applications*, 1992.

[44] M. Danielczuk *et al.*, "Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data," in *ICRA*, 2019.

[45] Y. Xiang, C. Xie, A. Mousavian, and D. Fox, "Learning rgb-d feature embeddings for unseen object instance segmentation," *CoRL*, 2020.

[46] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *ICRA*. IEEE, 2011, pp. 1–4.

[47] J. Papon and et al., "Voxel cloud connectivity segmentation - supervoxels for point clouds," in *CVPR*, 2013.

[48] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[49] K. He and et al., "Mask r-cnn," in *ICCV*, 2017.

[50] Y. Zhang, Z. Wu, H. Peng, and S. Lin, "A transductive approach for video object segmentation," in *CVPR*, 2020.

[51] J. Pont-Tuset and et al., "The 2017 davis challenge on video object segmentation," *arXiv:1704.00675*, 2017.

[52] N. Xu and et al., "Youtube-vos: A large-scale video object segmentation benchmark," *arXiv:1809.03327*, 2018.

[53] J. Lafferty and et al., "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[54] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "Lf-net: learning local features from images," in *NIPS*, 2018, pp. 6234–6244.

[55] A. Dai and et al., "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017.

[56] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *TPAMI*, no. 5, pp. 698–700, 1987.

[57] V. Vassilevska and et al, "Finding the smallest h-subgraph in real weighted graphs and related problems," in *International Colloquium on Automata, Languages, and Programming*, 2006.

[58] A. Dai and et al, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM ToG*, 2017.

[59] N. Bourbaki, *Lie groups and Lie algebras: chapters 7-9*. Springer Science & Business Media, 2008.

[60] F. B. Hildebrand, *Introduction to numerical analysis*. Courier Corporation, 1987.

[61] B. Calli and et al., "Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols," *IEEE Robotics and Automation Magazine*, 2015.

[62] S. Suwajanakorn and et al., "Discovery of latent 3d keypoints via end-to-end geometric reasoning," in *NIPS*, 2018.

[63] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018.

[64] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and Certifiable Point Cloud Registration," *IEEE Trans. Robotics*, 2020.

[65] A. ten Pas *et al.*, "Grasp pose detection in point clouds," *IJRR*, 2017.

[66] A. Murali and et al., "6-dof grasping for target-driven object manipulation in clutter," in *ICRA*, 2020.

[67] A. S. Morgan and et al, "Vision-driven compliant manipulation for reliable, high-precision assembly tasks," in *RSS*, 2021.

[68] C. Mitash and et al, "Task-driven perception and manipulation for constrained placement of unknown objects," *IEEE RAL*, 2020.