Deep 6-DoF Tracking of Unknown Objects for Reactive Grasping

Marc Tuscher[1,2] Julian Hörz[2] Danny Driess[2,3] Marc Toussaint[3,4]

***Abstract*— Robotic manipulation of unknown objects is an important field of research. Practical applications occur in many real-world settings where robots need to interact with an unknown environment. We tackle the problem of reactive grasping by proposing a method for unknown object tracking, grasp point sampling and dynamic trajectory planning. Our object tracking method combines Siamese Networks with an Iterative Closest Point approach for pointcloud registration into a method for 6-DoF unknown object tracking. The method does not require further training and is robust to noise and occlusion. We propose a robotic manipulation system, which is able to grasp a wide variety of formerly unseen objects and is robust against object perturbations and inferior grasping points.**

I. Introduction

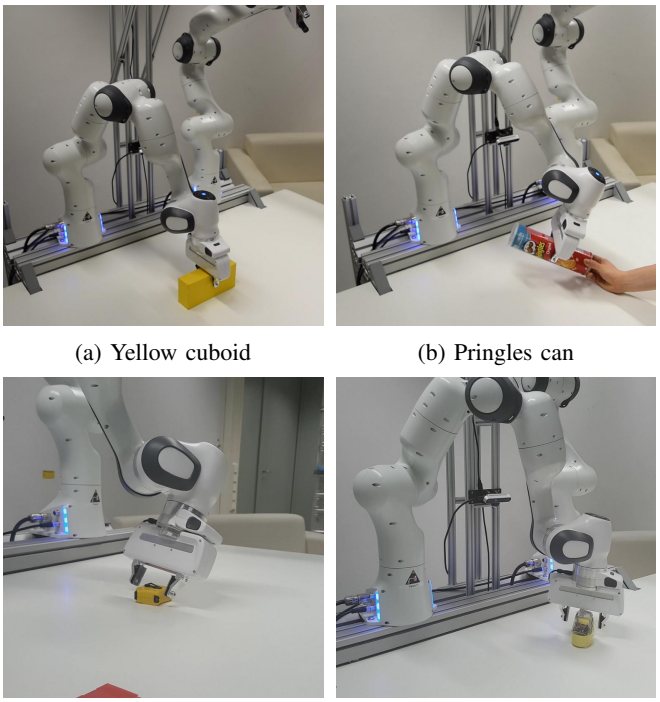
Aiming towards a wide-spread application of robots in natural or unstructured industrial environments, methods for robotic perception need to incorporate more general approaches. Robots manipulating in noisy and cluttered real-world scenarios require the ability to interact with arbitrary, unknown objects. Interacting with objects in the real world requires the ability to efficiently perceive the target object. Target objects need to be tracked in real-time. Versatile and robust tracking algorithms are required. Further, in order to act in a dynamically changing environment, robots need to react to changes and collaborate with other agents.

This work tackles the problem of reactive grasping of unknown objects. That is, grasping of unknown objects under perturbation and changes of the environment. We divide the problem of reactive grasping into three subproblems: detection and real-time tracking of unknown objects, computation of grasp configurations and reactive trajectory planning.

Object tracking methods are often trained on specific classes of objects [1], [2] or require a 3D model of the target object a priori [3], [4], [5], [6]. This heavily limits their usability for practical applications. Existing methods for unknown object tracking either target coarse grained tracking [7], [8], [9], [10], are prone to pose drifting [11] or limited to a specific environment [12]. Existing methods for grasp point detection are too slow for for real-time evaluation [13] or do not take the current robot configuration into account [14], [15]. Common methods for trajectory planning are designed for static scenes. Specifically, standard Rapidly Random Trees for trajectory planning are designed for static scenes only.

We propose a method for 6-DoF tracking of unknown objects based on Siamese Networks for 2D tracking and Iterative Closest Point (ICP) for pointcloud registration. This method is able to track the 6-DoF pose of a wide variety of objects. That is, given a set of RGB templates captured during tracking and a pointcloud template of the object captured at an initialization phase, the method outputs the subsequent poses of the target object in real-time. Further, we propose a method for grasp candidate computation and evaluation, taking the depth of the target object and the current robot state into account. We also propose a method for reactive trajectory planning in the presence of dynamic changes and collisions. This system is able to efficiently manipulate a diverse set of objects. We demonstrate this ability in our accompanying video[1]. We summarize our contributions as follows:

1) Our approach combines Siamese Networks for RGB tracking with ICP to realize a conceptually simple, but powerful 6-DoF tracking method: this method is able to track a wide variety of unknown objects under occlusion in real-time, while being class-agnostic and model-free.
2) We propose a novel method for reactive grasping, able to grasp dynamic objects even under perturbation of the robot or the target object.
3) We combine both approaches with a method for



(a) Yellow cuboid (b) Pringles can

(c) Measuring tape (d) Lashing strap

Fig. 1: Tracking and grasping unknown objects.

[1] sereact.
[2] Machine Learning and Robotics Lab, University of Stuttgart.
[3] Max-Planck Institute for Intelligent Systems, Stuttgart.
[4] Learning and Intelligent Systems, TU Berlin.

[1] https://youtu.be/Hew00rMw8qg

reactive trajectory planning accounting for dynamic changes of the environment.

The paper is structured as follows: in section II related work is presented, section III provides background on the methods we built upon, section IV explains our object tracking methond and section V describes the approach to reactive grasping. In section VI experiments are conducted in simulation and on a real robot.

II. Related Work

A. Object Tracking

a) Model-based tracking: One predominant paradigm for 6-DoF object tracking is model-based object tracking. That is, a 3 dimensional description of the target object is available a priori. A large body of work has been evaluated on this topic. Particle Filtering [16] proves to be useful in model-based object tracking [17]. Particle filter based tracking models often operate directly on pointclouds. Incorporating control inputs from the robot, to give a good estimate on the current object is also successfully applied to particle filter trackers [3]. Other appraoches extend particle filter tracking to multi-target tracking using a mixture in particles [17] and learning the correct correspondence to a certain target object via AdaBoost [18]. [19] uses a particle filter algorithm operating on RGB-pointclouds, thus, integrating texture information into tracking.

Fully integrating the robot kinematics in object tracking helps to reduce the degrees of freedom of the target object and serves as a good prior for vision based object tracking in robotic manipulation [5], [20]. However, such approaches rely on both, an object model and a robot model being present beforehand, and reduce the general applicability of the system.

Rendering multiple simulated views from the 3D model of the target object can be used to leverage deep transfer learning for pose estimation [21]. Garon et. al [4] use convolutional neural networks to directly operate on RGBD data, by training from scratch for each individual object. SegICP [22] successfully combines deep neural networks for semantic segmentation with ICP for known object tracking.

b) Pose estimation: Pose estimation and Object Detection are related fields of research: in [23] a deep learning method for object detection is extended by an object database yielding an object tracking system. Many approaches extend deep learning models for visual object detection to 6-DoF pose estimation from RGB data [24], [25], [26]. Applying deep learning models directly on pointcloud data seems promising since the rise of PointNet feature extractors [27], [28]. VoteNet [29] combines deep learning with Hough voting [30] to predict 3D bounding boxes on pointcloud input data.

However, most approaches to pose estimation lack the ability to run in real-time, and therefore render impractical for robotic manipulation. Tremblay et. al [1] train a deep neural network for pose estimation on synthetic RGB data using domain randomization. The method is suited as a real-time system for real-world robotic grasping of known objects. All approaches in this paragraph, however, are only able to track objects from a set of predefined classes.

c) Unknown objects: Despite the practical importance, only little research is done on tracking of unknown objects in robotics applications. Early work uses optical flow tracking and an integrated eye-in-hand vision system to grasp arbitrary objects with a visual servoing appraoch [31]. Experiments show that the approach only works in a specific environment. Other approaches rely on detecting and segmenting unkown objects in a known environment [12] or do not take the shape of the object into account [32]. Methods based on a large number of different algorithms, each tuned for a specific application, are usually brittle and sensitive to changing environments. A more recent approach to unknown object tracking applies multiple deep learning models to first segment the scene into model segments, then predict the position and orientation relative to the last frame [11]. Therefore, the method bootstraps the current object pose on earlier estimations. Experiments show that this approach is particularly prone to tracking drift: stable tracking of real-world objects is only possible for approximately one second.

B. Grasp Candidate Detection

Deep learning for robotic perception has changed the field of computer vision based grasp candidate detection. Existing approaches show that using neural networks on RGBD input successfully discriminate grasp candidates to evaluate the best grasp to manipulate objects [33], [34]. Neural nets are also used to generate 6-DoF grasps enabling more informed grasping and manipulation of unknown objects [35]. Recent work concentrates on the generation of dense pixel-wise quality maps and pixel-wise grasp information to generate multiple grasps at once [14]. These approaches use small neural networks to achieve real-time generation of grasp candidates and repeatedly set a new state-of-the-art on international grasping benchmarks [15]. However, such methods do not take the robot configuration into account.

C. Dynamic Trajectory Planning

Reactive planning is formerly shown to perform well in complex manipulation tasks [36]. Dynamic changes of the environment can also be efficiently handled by sampling based approaches [37].

Our method for object tracking combines Siamese Networks for template-matching in RGB frames with ICP-based pointcloud registration. Using this object tracking method we disentangle perception from grasp point calculation, providing the flexibilty to exectute complex queries on the scene and taking the current robot configuration into account. We further implement a variant of the conceptually simple bug algorithm in configuration space able to plan collision-free trajectories in a dynamic environment.

III. SiamMask & THOR

The method for 6-DoF tracking proposed in this work is built on SiamMask [38] and THOR [39] for template-matching based RGB object tracking. This section provides background on these methods. For more details please consider the original papers.

Given a template T and an input x, SiamMask computes a cross-correlated feature map

$$g_\theta(T,\ x) = f_\theta(T) \star f_\theta(x), \tag{1}$$

where f_θ denotes the feature extraction model used to process both, the novel input and the template. Equation 1 yields a bounding box in image coordinates and a tracking score. Computing a segmentation mask of the target object in the RGB observation is done using a separate branch in the neural network for mask refinement.

THOR consists of a long-term and a short-term module of RGB templates which can be used alongside SiamMask. In each iteration of tracking a subregion x, corresponding to the estimated position and size of the target object, is cropped from the current frame and fed through the SiamMask model alongside each individual RGB template from the THOR long-term module (LTM) $T_{l1}, \ldots, T_{l5}$ and short-term module (STM) $T_{s1}, \ldots, T_{s5}$

$$g_\theta(T_{l1},\ \ldots,\ T_{l5}; T_{s1},\ \ldots,\ T_{s5}; x). \tag{2}$$

Templates in the STM are updated in a first-in first-out manner. Every 10 iterations a new template is generated from the current crop and appended to the STM, while the oldest template in the STM is removed. T_{l1}, the ground truth template, remains in the LTM.

Let $z_i = f_\theta(T_i)$ denote the features extracted from template T_i. A Gram Matrix

$$\begin{bmatrix} z_1 \star z_1 & z_1 \star z_2 & \cdots & z_1 \star z_m \\ \vdots & \vdots & \ddots & \vdots \\ z_m \star z_1 & z_m \star z_2 & \cdots & z_m \star z_m \end{bmatrix}$$

is constructed, where each entry G_{ij} expresses the similarity of templates T_i and T_j by cross-correlating the features z_i and z_j. If a novel template T_c is similar enough to the ground-truth template T_{l1} and replacing an existing template in the LTM increases the volume of the parallelotope $\Gamma(z_1,\ \ldots, z_5)$ spanned by the feature vectors of each template, it is added to the LTM. A new template T_c is similar enough, if its features z_c satisfy the inequality

$$z_c \star z_1 > l \cdot G_{11} - \gamma, \tag{3}$$

where l is a hyperparameter to trade-off tracking performance against drifting robustness. γ is computed using the templates in the STM

$$\gamma = 1 - \frac{2}{N(N+1)G_{st,max}} \sum_{i<j}^{N} G_{st,ij} \tag{4}$$

where G_{st} is a Gram Matrix for the STM similar to G.

IV. 6-DoF Tracking

This section describes the approach this work takes on tracking arbitrary objects in 6 degrees of freedom. Since this approach is model-free, only information from a single RGBD camera at initialization time is used to describe the target object. Initialization of the tracking method requires an initial (arbitrarily defined) object pose in world coordinates and a 2D bounding box in image coordinates. We assume that we are able to detect an initial pose and a 2D bounding box of an object.

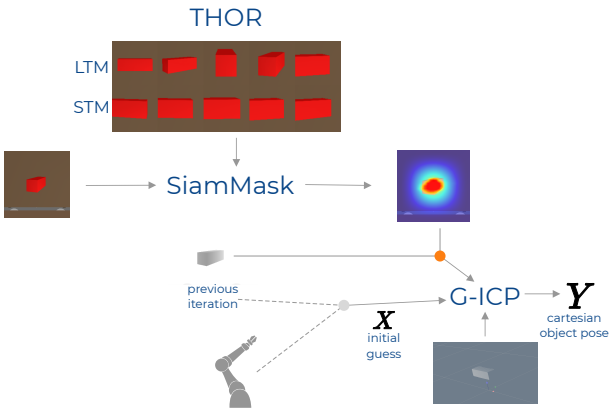


Fig. 2: Overview of our object tracking component.

Briefly, our method works as follows:

1) In an initialization phase, 3D bounding boxes and cartesian poses of arbitrary objects on a table are detected using background subtraction in depth images from a top down view.
2) SiamMask + THOR is started to track the target object in RGB frames. An initial pointcloud is saved for 3D pose estimation via pointcloud registration.
3) SiamMask tracks the target object in new frames in real-time, a pointcloud of this observation is segmented from the depth image. The initial pointcloud is fit to the new observation using Generalized-ICP [40] to retrieve the 6-DoF transform.

Figure 2 shows an overview of the tracking system. The system maps an RGBD input alongside an initial guess (gained from the previous iteration or a kinematic map) of the target object pose to an estimate of the current object pose. We call our object tracking method THOR + G-ICP.

a) Initialization: RGB tracking is initialized by supplying an initial frame and a bounding box containing the target object to the tracker. The bounding box is used to crop the frame to an initial RGB template T_{init}. The THOR long-term module (LTM)

$$T_{l1},\ \ldots,\ T_{l5} \leftarrow T_{init} \tag{5}$$

and short-term module (STM)

$$T_{s1},\ \ldots,\ T_{s5} \leftarrow T_{init} \tag{6}$$

are filled with this initial template of the object.

In order to find a template pointcloud of the target object, the segmentation mask provided by SiamMask in the first iteration of RGB tracking is used to retrieve the pixels $d_{ij} \in \mathbb{R}$ in the depth image, which correspond to points on the target object's surface. We project the depth values d_{ij} to points $x^k \in \mathbb{R}^3$ in world coordinates using the inverse camera projection $\hat{P} \in \mathbb{R}^{3\times4}$ via

$$x^k = \hat{P} \cdot (i \cdot d_{ij} \quad j \cdot d_{ij} \quad d_{ij} \quad 1)^T. \tag{7}$$

The set of these points $\mathcal{P}_{temp} = \{x^k\}_{k=1}^N$ is the template pointcloud, which is registered to a novel observation to

retrieve the transformation of the target object in world coordinates.

b) Tracking: RGB tracking is performed according to SiamMask with THOR as template module. In every step, SiamMask relies on cropping the correct subwindow from the image to perform template matching on a smaller region of the image. Cropping the correct subwindow, depends on prior information on the target object's location and its size in the image frame. If the robot is not moving the object, the position and size of the object in the previous frame is used. If the robot is moving the object, a kinematic map $\phi : \mathbb{R}^n \to \mathbb{R}^3$ is used to compute an estimate of the current object pose and a 3D bounding box containing the object. A 2D bounding box in image coordinates is then found by projecting the corners of the 3D bounding box to image coordinates and computing the bounding rectangle containing all corners. A step of RGB tracking yields a segmentation mask containing the target object in the current frame. This segmentation mask is used to select the corresponding pixels in the depth image. These pixels are then projected to points on the object's surface in world coordinates using equation 7. This set of points $\mathcal{P}_{obs}$ is used as an observation to fit the template pointcloud $\mathcal{P}_{temp}$.

To reduce the computational cost of pointcloud registration, both pointclouds are preprocessed by voxel grid filtering, reducing the dimesionality of the pointcloud while keeping the geometrical properties.

Pointcloud fitting is done using the Generalized-ICP algorithm, setting either the last known pose of the object or the pose computed using a kinematic map, in case the robot is handling the target object, as an initial guess. Formally, this can be seen as a function mapping from a set of pointclouds and an initial guess X to final transformation Y

$$\begin{aligned} h_{\text{G-ICP}} :\ & \mathbb{R}^{n\times 3} \times \mathbb{R}^{m\times 3} \times \mathbb{R}^{4\times 4} \to \mathbb{R}^{4\times 4} \\ h_{\text{G-ICP}} :\ & (\mathcal{P}_{obs}, \mathcal{P}_{temp}, X) \mapsto Y. \end{aligned} \tag{8}$$

We use this 6-DoF tracking method to perform reactive grasping on a real robot.

V. Reactive Grasping

In this section we develop a method to grasp (cuboid-shaped) objects, which is able to dynamically react to disturbances, both in the environment and the robot configuration. While our tracking method is able to process arbitrarily defined poses, this method requires the pose to be sensibly defined. That is, the coordinate system axes should be aligned with the edges of a bounding box surrounding the target object.

A. Grasp Configuration Observer

We propose a dynamic grasp configuration observer to find a variety of N grasp candidates for a cuboid-shaped object. Since our tracking module outputs the pose of a cuboid-shaped bounding box around an object, this approach is also used to manipulate arbitrarily shaped objects.

Let $q \in \mathbb{R}^n$ be the robot configuration. A grasp configuration candidate $g = (q, c, a)$ is defined by a joint configuration q, a cost $c \in \mathbb{R}$, and an age $a \in \mathbb{N}$. Grasp candidates are determined by minimizing an inverse kinematics problem

$$q^* = \underset{q\in\mathbb{R}^n}{\arg\min} \|q\|_W^2 + \sum_{i=1}^{M} \lambda_i \left\|\phi_i(q)\right\|^2, \tag{9}$$

where $\phi_i :\ \mathbb{R}^n \to \mathbb{R}^{d_i},\ \ y_i = \phi_i(q)$ are task maps that map a robot configuration to a d_i-dimensional space and $\lambda_i \in \mathbb{R}$ their weighting factors. W is a positive definite matrix providing regularization.

The following cost terms are considered during optimization.

a) Position: The position cost is defined as

$$\phi_{pos}(q) = p_o(q) - p_e(q), \tag{10}$$

where $p_o : \mathbb{R}^n \to \mathbb{R}^3$ and $p_e : \mathbb{R}^n \to \mathbb{R}^3$ are functions mapping the configuration q to the position of the target object and the end-effector, respectively.

b) Alignment: Performing successful grasipng requires the end-effector to be aligned with the object. We formulate the alignment cost as

$$\phi_{align}(q) = \left(1 - \left(v_{o,x}(q)^T v_{e,x}(q)\right)^2\right) \tag{11}$$

$$\left(1 - \left(v_{o,y}(q)^T v_{e,x}(q)\right)^2\right) \tag{12}$$

$$\left(1 - \left(v_{o,z}(q)^T v_{e,x}(q)\right)^2\right), \tag{13}$$

where $v_{o,x} : \mathbb{R}^n \to \mathbb{R}^3$ maps the configuration to the unit vector described by the x-axis of the target object coordinate system, analogously $v_{o,y}(q)$ and $v_{o,z}(q)$ for the y- and z-axis. $v_{e,x}(q)$ maps to the unit vector describing x-axis of the end-effector coordinate system. Minimizing this cost term aligns $v_{e,x}$ to one of the axes of the target object coordinate system.

c) Collision: In order to successfully grasp the object, we need to avoid collisions with the object itself. The collision cost term is defined as

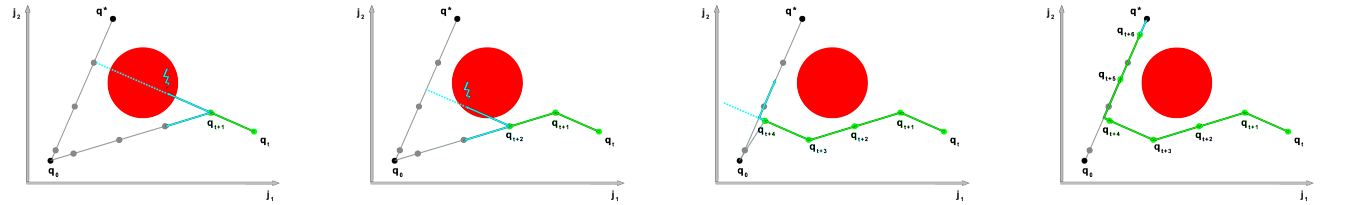
$$\phi_{coll}(q) = (d(q) - m)[d(q) < m] \tag{14}$$

where d maps q to the pairwise distances of all shapes of the robot and the target object. m is an upper threshold on the minimum distance. If the distance of two shapes is less than m this cost term becomes active.

d) Joint Limits & Homing: We further add a cost term $\phi(q)_{limit}$ increasing when joint limits of the robot are violated and a homing cost term $\phi(q)_{home}$.

q^* is the configuration of a feasible grasp. However, computing N grasp candidates starting at the same configuration q_0 results in all candidates defining the same configuration q^*. This is especially relevant for the alignment term ϕ_{align}, which has multiple local minima. Instead of explicitly enumerating possible different alignments as in [41], [42], we diversify grasp candidate computation by introducing a random alignment term, effectively randomizing the approach axis of a grasp candidate. That is, at initialization we compute a random orthonomal basis

$$R = \begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix} \in \mathbb{R}^{3\times 3}. \tag{15}$$



(a) Linearly interpolating between q_t and q^* would lead to collisions with the object.

(b) After backstepping two steps, the shortest path would still end in collisions.

(c) After four backsteps we can take the path to our target trajectory.

(d) We follow the target trajectory towards q^*.

Fig. 3: Procedure of backstepping if collisions are encountered on the linear interpolation between q_t and q^*.

We use R to introduce an additional cost term

$$\phi_{rand}(q) = \begin{pmatrix} 1 - \left(v_1(q)^T v_{x,e}(q)\right)\left(v_1(q)^T v_{x,e}(q)\right) \\ 1 - \left(v_2(q)^T v_{y,e}(q)\right)\left(v_2(q)^T v_{y,e}(q)\right) \\ 1 - \left(v_3(q)^T v_{z,e}(q)\right)\left(v_3(q)^T v_{z,e}(q)\right) \end{pmatrix}.$$

We further introduce two additional weighting functions

$$f_{inc}(a) = \frac{1}{1 + \exp(\frac{\alpha}{2} - a)} \tag{16}$$

$$f_{dec}(a) = \frac{1}{1 + \exp(a - \frac{\alpha}{2})} \tag{17}$$

depending on the age a of a grasp candidate. α is a constant threshold. During optimization we weight $\phi_{pos}(q)$, $\phi_{align}(q)$, $\phi_{coll}(q)$ with $f_{inc}(a)$ and $\phi_{rand}(q)$ with $f_{dec}(a)$ ensuring that only the approach axis is randomized and no force balance arises between $\phi_{align}(q)$ and $\phi_{rand}(q)$.

B. Grasp Candidate Ranking

Choosing the best grasp from N grasp candidates is non-trivial. The essential and hard requirement for a good grasp candidate is whether the grasp configuration ends in an end-effector position inside the target cuboid. We compute this using the end-effector position relative to the cuboids coordinate system and the shape of the cuboid. We assume that success of grasping increases with decreasing distance to the center of the cuboid. If two candidates are both inside the cuboid and also share the same distance to the center, we choose the one with lower homing costs.

C. Updating Grasp Candidates

Since an online reactive grasping approach needs to take the changing environment into account, we continously update the grasp candidates. At initialization all grasp candidates have the same cost vector c. In each iteration the age a of each grasp is incremented and the cost vector of each grasp is evaluated using the current configuration q_t.

D. Trajectory Planning

Our goal is to follow a trajectory starting in the current configuration q_t and ending in the configuration q^* of the best grasp candidate q^*. Due to the possibilty of a changing environment, and more specifically the change of the target object pose, we cannot plan a full trajectory in advance. Movements of the target object can lead to collisions during a grasp approach following a pre-planned trajectory. We mitigate this problem by using a technique we call *backstepping*.

Our goal is to follow a target trajectory T defined by the linear interpolation with stepsize τ between the homing configuration q_0 and the configuration q^* of the best grasp candidate. At each step we find the closest configuration q_s on the target trajectory T to our current configuration q_t. If a step of stepsize τ ends up in a collision with an object, we take a step towards q_0. The procedure of backstepping is depicted in figure 3.

VI. Experiments

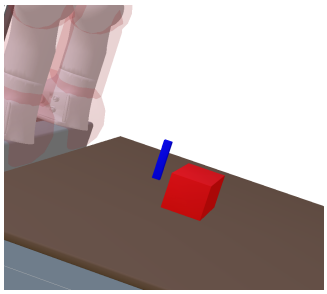
This section describes experiments conducted on object tracking using THOR + G-ICP in simulation and experiments using the complete system for reactive grasping on a real robot.

A. Object Tracking

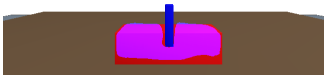
For quantitive comparison to an existing method we conduct experiments in simulation. We render a camera view on the scene which is located at the same pose as our camera on the real world setup. We compare THOR + G-ICP to a particle-filter based object tracking method from the PCL library [43] (named particle in the plots). Errors are reported in terms of position and rotation. Rotational errors are reported in terms of the geodesic loss.

a) Occlusion: Occlusions occur frequently during robotic manipulation. When grasping an object with a parallel jaw gripper, parts of the object are already occluded by the gripper fingers. In this experiment a small box moving into the scene from the top occludes the target object, as seen in figure 4a. Results, seen in figure 4c, show that occlusion has almost no impact on the performance of THOR + G-ICP. The accuracy of the particle tracker, however, is reduced proportional to the amount of occlusion. The stability of THOR + G-ICP relies on the fact that SiamMask is able to accurately segment the tracked object from the occluding object. As seen in figure 4b, the segmentation mask of SiamMask contains parts of the target object only. G-ICP is able to fit the template pointcloud to the observations seamlessly, since the observation pointcloud omits the occluded part.

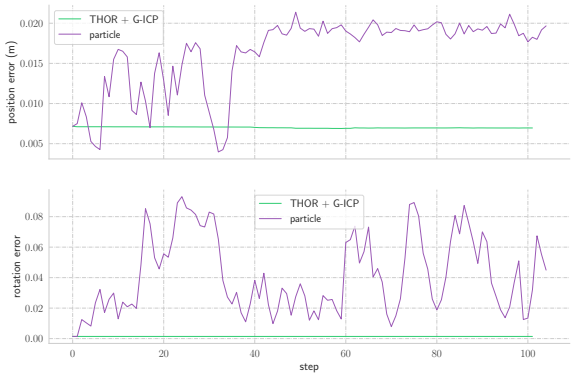
b) Manipulation: Supporting robotic manipulation is the main purpose of our object tracking method and therefore an important experiment. It incorporates many different possibilites which can occur during object tracking, e.g. translational and orientational movements as well as partial occlusion due to the gripper fingers or the second robot. Figure 5 shows the manipulation process and results of object tracking during manipulation. Results of object tracking are depicted in figure 5e. These show that both methods are able



(a) A small rectangle occludes the target object.



(b) Camera view on the scene. The object, which occludes the target object, is not contained in the segmentation mask.



(c) Results of object tracking during occlusion. THOR + G-ICP remains stable during the complete tracking sequence.

Fig. 4: Object tracking during occlusion.

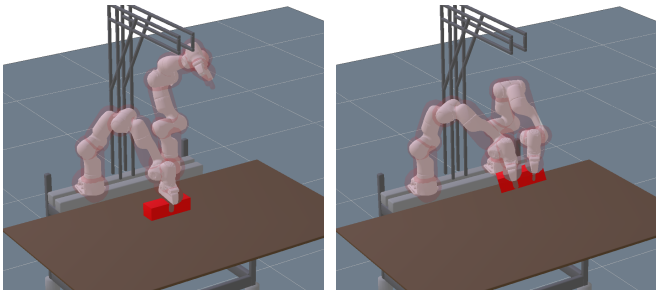
to track the target object during manipulation. However, the particle filter tracker loses the object once completely. This corresponds to the moment of the handover (figure 5b) and is due to the occlusion both gripper fingers generate on the object. Tracking with THOR + G-ICP is stable during the whole manipulation process and also generates almost no rotational error.

B. Reactive Grasping

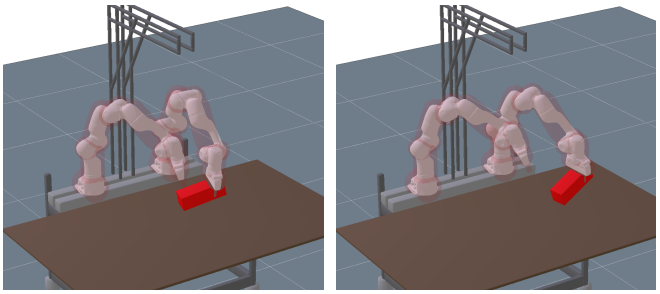
We perform reactive grasping with a Franka Emika Panda Robot. Objects on the table are detected from a top down view using a *ASUS Xtion PRO LIVE RGBD* camera. Once an object on the table is detected, THOR + G-ICP is initialized with an *Intel RealSense D415* camera from a side view. These experiments are also shown in our accompanying video.

a) Unknown Objects: Experiments on grasping of unknown objects are conducted in this section. Figure 1 shows the objects which are manipulated in our experiments. THOR + G-ICP is able to track each of these objects during the complete grasping process. Further, the approach to reactive grasping presented in this work is able to grasp each object even under disturbances. The pringles can is moved by a human during the manipulation process. However, our approach is able to grasp the object successfully. Notably, the gripper finger occludes the measuring tape by more than half of its size in the camera view. The lashing strap is surrounded by a plastic firm, exposing heavy reflections. Although reflections impose difficulties on classical computer vision techniques, THOR + G-ICP is able to track the lashing strap effortlessly.

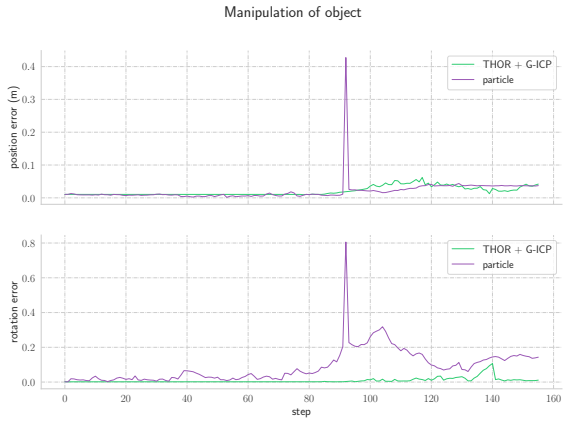
b) Collaborative Manipulation: Experiments on grasping under heavy disturbances are conducted. During theses



(a) (b)



(c) (d)



(e) Results of tracking during manipulation.

Fig. 5: Object tracking during manipulation. In a) the first robot picks up the target object and hands it over to the second robot in b). The second robot moves the object to an intermediate position in c) and finally moves to the goal position d) while keeping the object in hand.

experiment the robot and the target object are moved and re-oriented by a human during grasp approaches. The robot is also prevented from grasping the object and moved to a position beside the object, likely leading to collisions with the object. Target objects are successfully tracked throughout the process and the path planning component is able to recover from each state, leading to a successful grasp in each experiment.

VII. Conclusion

We present a system to track and dynamically manipulate unknown objects. Our experiments show that our approach to reactive grasping is successful even under occlusion of the target object and under perturbation of either the robot or the object.

REFERENCES

[1] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects," *arXiv:1809.10790 [cs]*, Sept. 2018.

[2] S. Giancola, J. Zarzar, and B. Ghanem, "Leveraging Shape Completion for 3D Siamese Tracking," *arXiv:1903.01784 [cs]*, Mar. 2019.

[3] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic Object Tracking using a Range Camera," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3195–3202, Nov. 2013.

[4] M. Garon and J.-F. Lalonde, "Deep 6-DOF Tracking," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 11, pp. 2410–2418, Nov. 2017.

[5] T. Schmidt, R. Newcombe, and D. Fox, "DART: Dense Articulated Real-Time Tracking," in *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation, July 2014.

[6] J. Schulman, A. Lee, J. Ho, and P. Abbeel, "Tracking deformable objects with point clouds," in *2013 IEEE International Conference on Robotics and Automation*. Karlsruhe, Germany: IEEE, May 2013, pp. 1130–1137.

[7] A. Feldman, T. Balch, M. Hybinette, and R. Cavallaro, "The Multi-ICP Tracker: An Online Algorithm for Tracking Multiple Interacting Targets," p. 22.

[8] J. Cho, S. Jin, X. Pham, J. Jeon, J. Byun, and H. Kang, "A Real-Time Object Tracking System Using a Particle Filter," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China: IEEE, Oct. 2006, pp. 2822–2827.

[9] A. Almeida, J. Almeida, and Rui Araujo, "Real-Time Tracking of Moving Objects Using Particle Filters," in *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005. ISIE 2005.* Dubrovnik, Croatia: IEEE, 2005, pp. 1327–1332.

[10] M. Fleder and S. Pillai, "3D Object Tracking Using the Kinect," p. 8.

[11] F. Leeb, A. Byravan, and D. Fox, "Motion-Nets: 6D Tracking of Unknown Objects in Unseen Environments using RGB," *arXiv:1910.13942 [cs]*, Oct. 2019.

[12] D. Kragic, M. Björkman, H. I. Christensen, and J.-O. Eklundh, "Vision for robotic object manipulation in domestic settings," *Robotics and Autonomous Systems*, vol. 52, no. 1, pp. 85–100, July 2005.

[13] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics," *arXiv:1703.09312 [cs]*, Mar. 2017.

[14] D. Morrison, P. Corke, and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," in *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[15] S. Kumra, S. Joshi, and F. Sahin, "Antipodal robotic grasping using generative residual convolutional neural network," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020.

[16] A. Doucet, N. de Freitas, and N. Gordon, "Sequential Monte Carlo Methods in Practice," p. 12.

[17] Vermaak, Doucet, and Perez, "Maintaining multimodality through mixture tracking," in *Proceedings Ninth IEEE International Conference on Computer Vision*. Nice, France: IEEE, 2003, pp. 1110–1116 vol.2.

[18] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe, "A Boosted Particle Filter: Multitarget Detection and Tracking," in *Computer Vision - ECCV 2004*, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, T. Pajdla, and J. Matas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, vol. 3021, pp. 28–39.

[19] R. B. Rusu, "Tracking 3D objects with Point Cloud Library," Jan. 2012.

[20] W. Gao and R. Tedrake, "FilterReg: Robust and Efficient Probabilistic Point-Set Registration using Gaussian Filter and Twist Parameterization," *arXiv:1811.10136 [cs]*, July 2019.

[21] Y. Xiao, X. Qiu, P.-A. Langlois, M. Aubry, and R. Marlet, "Pose from Shape: Deep Pose Estimation for Arbitrary 3D Objects," *arXiv:1906.05105 [cs]*, Aug. 2019.

[22] J. M. Wong, V. Kee, T. Le, S. Wagner, G.-L. Mariottini, A. Schneider, L. Hamilton, R. Chipalkatty, M. Hebert, D. M. Johnson, *et al.*, "Segicp: Integrated deep semantic segmentation and pose estimation," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[23] B. A. Erol, A. Majumdar, J. Lwowski, P. Benavidez, P. Rad, and M. Jamshidi, "Improved Deep Neural Network Object Tracking System for Applications in Home Robotics," in *Computational Intelligence for Pattern Recognition*, W. Pedrycz and S.-M. Chen, Eds. Cham: Springer International Publishing, 2018, vol. 777, pp. 369–395.

[24] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-DoF Object Pose from Semantic Keypoints," *arXiv:1703.04670 [cs]*, Mar. 2017.

[25] S. Song and J. Xiao, "Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 808–816.

[26] J. Hou, A. Dai, and M. Nießner, "3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans," *arXiv:1812.07003 [cs]*, Apr. 2019.

[27] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," *arXiv:1612.00593 [cs]*, Apr. 2017.

[28] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," *arXiv:1706.02413 [cs]*, June 2017.

[29] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep Hough Voting for 3D Object Detection in Point Clouds," *arXiv:1904.09664 [cs]*, Aug. 2019.

[30] B. Leibe, A. Leonardis, and B. Schiele, "Combined Object Categorization and Segmentation with an Implicit Shape Model," p. 16.

[31] R. Luo, R. Mullen, and D. Wessell, "An adaptive robotic tracking system using optical flow," in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. Philadelphia, PA, USA: IEEE Comput. Soc. Press, 1988, pp. 568–573.

[32] A. Pieropan, N. Bergstrom, M. Ishikawa, and H. Kjellstrom, "Robust 3D tracking of unknown objects," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 2410–2417.

[33] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg, "Learning ambidextrous robot grasping policies," *Science Robotics*, vol. 4, no. 26, p. eaau4984, 2019.

[34] D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint, "Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[35] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *International Conference on Computer Vision (ICCV)*, 2019.

[36] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wüthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg, "Real-time perception meets reactive motion generation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, 2018.

[37] M. Otte and E. Frazzoli, "Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[38] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast Online Object Tracking and Segmentation: A Unifying Approach," *arXiv:1812.05050 [cs]*, Dec. 2018.

[39] A. Sauer, E. Aljalbout, and S. Haddadin, "Tracking Holistic Object Representations," *arXiv:1907.12920 [cs, stat]*, Aug. 2019.

[40] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: Science and Systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[41] D. Driess, O. Oguz, and M. Toussaint, "Hierarchical task and motion planning using logic-geometric programming (hlgp)," RSS Workshop on Robust Task and Motion Planning, 2019.

[42] D. Driess, J.-S. Ha, and M. Toussaint, "Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image," in *Robotics: Science and Systems (R:SS)*, 2020.

[43] "Tracking 3D objects with Point Cloud Library - Point Cloud Library (PCL)," http://www.pointclouds.org/news/2012/01/17/tracking-3d-objects-with-point-cloud-library/.