Benchmarking In-Hand Manipulation

Silvia Cruciani*[1], Balakumar Sundaralingam*[2], Kaiyu Hang[3], Vikash Kumar[4], Tucker Hermans[2,5], and Danica Kragic[1]

***Abstract*—The purpose of this benchmark is to evaluate the planning and control aspects of robotic in-hand manipulation systems. The goal is to assess the system's ability to change the pose of a hand-held object by either using the fingers, environment or a combination of both. Given an object surface mesh from the YCB data-set, we provide examples of initial and goal states (i.e. static object poses and fingertip locations) for various in-hand manipulation tasks. We further propose metrics that measure the error in reaching the goal state from a specific initial state, which, when aggregated across all tasks, also serves as a measure of the system's in-hand manipulation capability. We provide supporting software, task examples, and evaluation results associated with the benchmark.**

***Index Terms*—Performance Evaluation and Benchmarking; Dexterous Manipulation.**

I. Introduction

IN-HAND manipulation is the task of changing the grasp on an hand-held object without placing it back and picking it up again. In recent years, researchers have demonstrated in-hand manipulation skills on real robotic platforms with end-effectors ranging from simple grippers [1] to anthropomorphic hands [2]–[4], [21] on everyday objects (some examples shown in Fig. 1). This growing interest in in-hand manipulation has created the need for a thorough comparison of the proposed methods using a common set of tasks and metrics. However, the diversity in the methodological approaches and hardware used makes the burden of comparison rather complex for individual research labs to perform. There is thus a need for a benchmarking protocol that allows researchers to evaluate new methodologies in a more standardized manner.

In this work, we propose a benchmarking scheme for in-hand manipulation that can extend to all kinds of robotic systems. All the supporting material is available at https://robot-learning.cs.utah.edu/project/benchmarking_in_hand_manipulation.

Manuscript received: August, 15, 2019; Revised November, 18, 2019; Accepted December, 1, 2019.

This paper was recommended for publication by Editor Han Ding upon evaluation of the Associate Editor and Reviewers' comments.

S. Cruciani was supported by Swedish Foundation for Strategic Research project GMT14-0082 FACT and B. Sundaralingam was supported by NSF Award 1846341

*These two authors contributed equally.

[1]Division of Robotics, Perception & Learning, EECS, KTH Royal Institute of Technology, Stockholm, Sweden

[2]Robotics Center & School of Computing, University of Utah, UT USA.

[3]Dept. of Mechanical Engineering & Material Science, Yale University, New Haven, CT, USA.

[4]Google AI

[5]NVIDIA Research

E-mail:`cruciani@kth.se, bala@cs.utah.edu`

Digital Object Identifier (DOI): see top of this page.

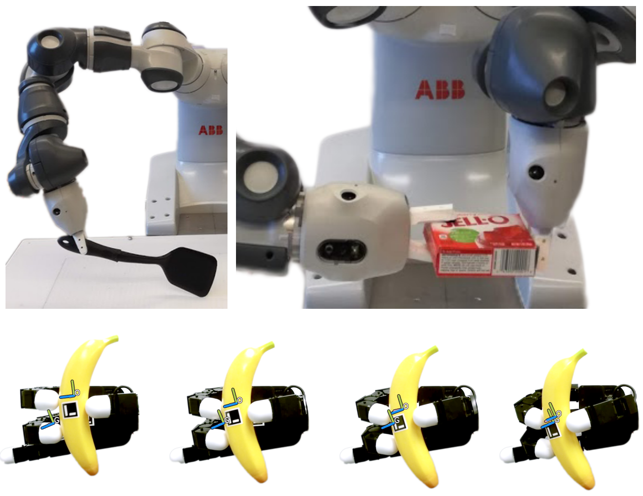


Fig. 1: In-hand manipulation strategy examples: *top left:* pushing against the environment; *top right:* bi-manual push; *bottom:* exploitation of dexterity in a multi-finger hand.

While some benchmark examples are available for evaluating grasp planning in a simulated environment [5], [6], we believe that for contact-rich tasks such as grasping and in-hand manipulation it is of fundamental importance to evaluate the performance of the system in a physical, real-world scenario. Therefore, we propose a series of tasks to evaluate the capabilities of a robotic system to plan and execute in-hand manipulation using a set of objects available from the Yale-CMU-Berkeley (YCB) object and model set [7]. This set provides both object meshes and physical objects, so that different research groups can rely on the same test set.

While our benchmark focuses on physical robot execution, the procedure is still valid in simulation and also for benchmarking of planning algorithms. The availability of object meshes for the YCB data-set enables easy evaluation in common robotics simulators. As such, we will host results for simulation in a separate section of the associated website.

To motivate our choice of protocol and metrics, we briefly review the relevant in-hand manipulation literature with a focus on how researchers perform evaluation and validation of the proposed methods. Active research on in-hand manipulation explores a wide variety of robotic systems, ranging from grippers [8], [9] to dexterous hands [2], [10]. Researchers have also augmented mobile robots [11] and multiple arms to behave as fingertips [12]–[14] to perform in-hand manipulation. Methodologies for analyzing the dexterity of robotic hands were developed in [15] and [16]; however, they focus on the hands' kinematic reachability and actuation, rather than on execution with different objects and tasks.

With multi-fingered dexterous hands, in-hand manipulation has been performed leveraging the redundancy in the fingers to move the object without completely releasing the grasp [2], [10], [17]–[21]. For under-actuated hands, model based control has been successfully employed [22]–[25]. Alternatively, task-specific design of under-actuated hands enables a limited set of re-positioning possibilities [26]–[28]. Researchers have leveraged learning methods to manipulate objects [3], [29], [30], and validated machine learning approaches in physics simulation [31], [32]. In-hand manipulation has also been explored as a planning problem [19], [33].

With parallel grippers, in-hand manipulation has been achieved by exploiting the concept of *extrinsic dexterity* [1], in which the degrees of freedom of a certain gripper are enhanced by means of external supports such as contact with the environment, gravity and friction. Some of these works exploit gravity and inertial forces to initiate the motion of the object [30], [34]–[37], while others rely on pushes against an external contact. This contact can be either a fixture or a surface present in the environment [8], [38], [39], or a second gripper in a dual-arm manipulation scenario [9].

One of the main challenges in in-hand manipulation is balancing the object in a stable configuration during the execution. An alternative to keep stability is dynamic in-hand manipulation by tossing the object in the air and catching it in the desired pose [1], [40]. A more conservative approach is to balance the object with extra support during execution. Researchers have used a planar surface [4], [41] and even the palm of the hand [21] to balance the object.

Across these various methods for in-hand manipulation, we found that the goal was defined as either a target pose [10], [21], [42], desired contact locations on the object [19], or both [9]. The target pose can range from single dimensional rotation [36], through planar [3], [29] to full 3D poses [10]. When desired contacts are defined, the goal is the position of each specific contact point on the object that the robot needs to reach [19]. A combination of target pose and desired contact is commonly user-specified [9]. We summarize the related work in terms of the goal definition, use of extra object support, and real-world execution in Table I.

Recently, Rajeswaran *et al.* [31] proposed a set of manipulation tasks to benchmark dexterous manipulation. The benchmark primarily focuses on simulated dexterous hands. The proposed tasks do not account for all kinds of manipulation platforms (e.g. the proposed hammer task may be impractical with a parallel gripper). They also do not propose qualitative error metrics to measure the capabilities of a system. In contrast, we design platform-agnostic in-hand manipulation tasks and we propose qualitative metrics to evaluate the system's performance.

We organize the remainder of this letter as follows. We describe our protocol in Sec. II, followed by a discussion of the benchmarking guidelines in Sec. III. We demonstrate example results for representative methods of in-hand manipulation, benchmarked using the proposed protocol in Sec. IV, and conclude the letter in Sec. V.

Article	**Goal**		**Obj. support**	**Real-world**
	Pose	**Contact**		
[34]–[36]	1D rot.		No	Yes
[3], [43], [44]	2D tran.		No	Yes
[14], [28]	SE(2)		Ext. surface	Yes
[8], [37], [42]	SE(2)		No	Yes
[4]	SE(3)	Designed	Ext. surface	Yes
[2], [21]	SE(3)		Palm	Yes
[10], [45]	SE(3)		No	Yes
[9]	SE(3)	Points	No	Yes
[19]		Points	NA	No
[31]	SE(3)	Demonstration	NA	No

Table I: Research articles categorized in terms of goal definition and other relevant features to protocol formulation.

II. Protocol Design

Our objective is to assess the capability of a system to execute in-hand manipulation tasks in a goal-directed manner. We define an in-hand manipulation task as the problem of changing the grasp on an object without placing it on a support surface. In this given task, the goal is to change the object's configuration inside the robot's hand, from an initial grasp to a final desired grasp. A grasp, G, is defined by the hand pose, H, with respect to the object's frame and the contact points, P, made between the hand and the object. Since different robot geometries will require different hand poses and contact points to successfully grasp the object of interest, we specify the protocol in such a way that it can be adapted to different grasps according to the available setup.

We highlight that the specification of H w.r.t. the object is equivalent to specifying the object's pose inside the hand. We do not constrain the *absolute* object's pose in a different, fixed reference frame because the focus is on the *relative* pose between the hand and the object.

A. Task Description

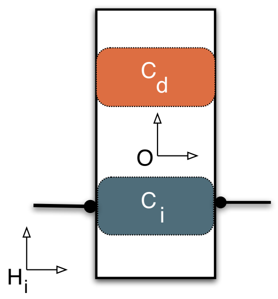
We describe the task with the aid of Fig. 2. An in-hand manipulation task is defined by:

- An initial contact region, C_i, and a desired contact region, C_d, defined on the object's surface; examples of some of these contact regions on the YCB objects meshes are shown in Fig. 3.
- An initial grasp pose, H_i, and the desired grasp pose, H_d, identifying the hand's pose with respect to the object's reference frame.

The initial contact points, P_i, and the final contact points, P_d, between the hand and the object must lie inside the respective contact regions, in which they can be adjusted according to their feasibility with the hardware used.

Based on our literature review, we define three levels of performing an in-hand manipulation task, depending on which part of the desired configuration is targeted:

- **Level I** desired hand pose H_d only;
- **Level II** desired contact region C_d only;
- **Level III** both C_d and H_d.



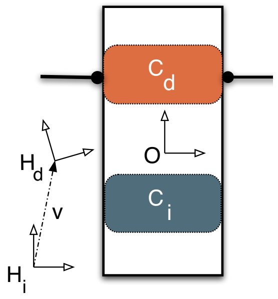


Fig. 2: We illustrate the robot's fingertips as lines with solid circles defining contact with the rectangular object. We illustrate the task in 2D for clarity, but define it in 3D. Frames H_i and H_d define the hand's initial and desired final poses respectively. The left image shows the initial task setup, while the right shows a solution reaching the desired state.

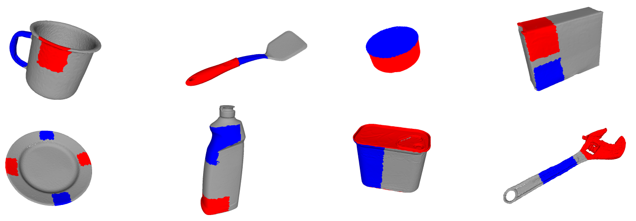


Fig. 3: Example contact regions defining in-hand manipulation tasks on YCB objects. The contact points on the object must be moved from the area in red to the area in blue.

With level I, the hand can make contact at any region on the object and with level-II, the initial and final hand pose are not constrained. These different tasks allow for the exploration of different in-hand manipulation behaviors (e.g. finger gating vs in-hand rolling) without specifying how the manipulation should be performed. Additionally, we aim for our tasks to adapt to the needs and capabilities of different robots.

B. Procedure

For each task, the procedure runs as follows:

1) The object is set at the initial grasp in the robot's hand (e.g. a human places the object, the robot autonomously grasps the object, etc.). The contacts P_i between the hand and the object must lie in the initial contact region C_i. The object pose at this initial pose is recorded as $\hat{H}_i$. If the position error% (computed as $\frac{||s_i - \hat{s}_i||}{||sr_d - s_i||} \times 100$ or orientation error% computed using Eq. 3, between the setup initial hand pose $\hat{H}_i$ and the initial hand pose H_i from the dataset is more than 10%, this experiment must be discarded.
2) The in-hand manipulation method is run, to move the object toward the desired configuration, defined by contact region C_d (or giving specific points P_d in the region C_d), and by H_d.
3) The hand reaches a new pose on the object H_r and new contacts P_r. This final configuration is recorded. The times for planning and executing are recorded.

C. Setup Description

We defined several tasks for many of the YCB objects [7]. These tasks contain initial and desired contact regions C_i, C_d, and are available in our protocol's associated website. The given contact regions are meshes that represent parts of the object's surface. If a method requires specific contact points to be defined instead of contact regions, researchers can define the corresponding contact points P_i, P_d within these regions to match their hardware. In particular, the chosen YCB objects cover a wide spectrum in terms of size which enables benchmarking hands of different sizes.

The hand poses are strictly dependent on the robot's hand frame. We additionally provide some examples of H_i and H_d that identify the required change in grasp pose. The main objective is to obtain a relative change T between the initial grasp H_i^{rob} and the final grasp H_d^{rob} of your specific robot that matches the transformation T from H_i to H_d (i.e.$T = H_d H_i^{-1}$). Hence, the absolute hand poses for a specific robot should be adapted , but still impose the same relative change to the given frames ($H_d^{rob} = T H_i^{rob}$).

Our examples work for end-effectors roughly the size of adult human hands, but most of them can be executed with bigger hands as well.

D. Robot/Hardware Description

This benchmark is targeted towards manipulation systems equipped with either a gripper or a multi-fingered hand. The tasks that we describe can also be performed with multiple arms, which could behave together as a dexterous hand.

A perception system is required to estimate the initial and final pose of the object and the hand (with pose of the links for level II and level III tasks). For computing the error metrics, we additionally require the mesh of the links of the hand. Given all this information, we provide software to estimate the errors as shown in Fig. 4.

E. Execution Constraints

During the in-hand manipulation planning and execution (step 2 in section II-B), a human cannot intervene once the object has been placed inside the robot's hand. The object should be in a stable grasp at the initial and final configurations. For the purpose of this benchmark, a grasp is stable if the hand pose with respect to the object is constant for 5 seconds after execution. During this time, the object must be in contact only with the robot's hand, without any help from external supports.

During the in-hand manipulation execution, the object can break contact with the robot. However, the contact cannot be broken to favour stable contacts with support surfaces. In fact, we do not allow several pick-and-place executions to change the grasp (i.e. regrasping [46]), because this does not classify as in-hand manipulation. In contrast, exploiting external surfaces to push the object inside the hand and performing non-prehensile manipulation is allowed, because the contact between the object and the robot is maintained while the object is in contact with the external surface.

Similarly, the object can be thrown in the air and caught in a different configuration, since no contact with external surfaces happens once the robot releases the grasp. A second robot hand can be used as a support surface but not for holding the object, for regrasping between hands is not allowed.

III. Benchmark Guidelines

A. Scoring

We score the task success based on the error between the reached grasp $G_r = [H_r, P_r]$ and the desired grasp $G_d = [H_d, C_d]$. To ensure the robustness of a method, each grasp set needs to be run on the robot five times for the same object. We propose two separate sets of metrics to quantify error in hand pose and reached contact.

To quantify hand pose error, we split the hand pose H into position vector s and orientation quaternion q and compute the error in position and orientation separately. Due to differences in the object and hand sizes, we also compute an error normalized by the distance between initial and final poses. The error between the reached hand position s_r and the desired hand position s_d is the Euclidean distance l_2 norm

$$err_{pos} = ||s_d - s_r||_2. \tag{1}$$

To compute the percentage position error, we divide the error by the distance between the initial hand position s_i and desired hand position s_d,

$$err_{pos}\% = 100 \times \frac{||s_d - s_r||_2}{||s_d - s_i||_2}. \tag{2}$$

To measure the orientation error, we use the sum of quaternions [47]. Given the desired hand orientation q_d and the orientation of the reached hand pose q_r, the orientation error is defined as

$$err_{or}\% = 100 \times \frac{\min(||q_d - q_r||_2, ||q_d + q_r||_2)}{\sqrt{2}}. \tag{3}$$

To quantify error in reaching the desired contact regions C_d, we propose two metrics: Euclidean and Geodesic. We want all contacts made by the robot to be inside the desired contact region C_d. Hence these metrics are used to find the largest distance max_d between the robot links and the desired contact region. We will first discuss these metrics to compute error between two points and then describe how to compute them between meshes.

Given two points, p_1 and p_2, the Euclidean metric $G_{euc}(\cdot)$ is computed as the l_2 norm

$$G_{euc} = ||p_2 - p_1||_2. \tag{4}$$

The Geodesic metric $G_{geo}(\cdot)$ is the shortest path between the points on the object surface ψ

$$G_{geo} = ||p_2 - p_1||_\psi. \tag{5}$$

Efficient geodesic computation methods with software implementations are available [48].

For using these metrics with meshes, we propose the algorithm shown in Algorithm 1. We use the object mesh O, the meshes of the N links of the robot $L_{i\in N}$ making contact with the object, and the mesh of the desired contact region C_d. To compute these metrics, we first project the desired contact region C_d and the robot meshes $L_{i\in N}$ onto the object mesh O as shown in lines 5-10. Given the desired contact region C_d, we compute the points P on the faces of C_d that are intersecting with the faces of the object mesh O using the function `intersect(·)`. Many algorithms exist for performing this computation [49], [50]. For every point in P, we find a vertex in object mesh O with the `min-vertex(·)` function. This function finds the vertex that has the shortest Euclidean distance to the point. The set of vertices are stored as the desired contact vertices V_c. This is also done for the robot meshes to obtain P_r (lines 7-10).

Algorithm 1: Contact region error computation

```
   Data: $O$,$L_{i\in N}$,$C_d$
   Result: max_d
 1 $P_r \leftarrow$[]
 2 $V_c \leftarrow$[]
 3 max_d$\leftarrow 0$
 4 $P \leftarrow$ intersect($C_d$,$O$)
 5 for $v \in P$ do
 6     $V_c$.append(min-vertex($v$,$O$))
 7 for $i \in N$ do
 8     $V \leftarrow$ intersect($L_i$,$O$)
 9     for $v \in V$ do
10         $P_r$.append(min-vertex($v$,$O$))
11 for $v \in P_r$ do
12     min_d$\leftarrow \infty$
13     for $p \in V_c$ do
14         d$\leftarrow G(v,p)$
15         if d<min_d then
16             min_d$\leftarrow$d
17     if min_d>max_d then
18         max_d$\leftarrow$min_d
19 return max_d
```

For each vertex in P_r, we find the shortest distance to the desired contact set of vertices V_c (lines 11-18). The distance function $G(\cdot)$ in line 14 is replaced by the Euclidean G_{euc} or geodesic G_{geo} metrics to report the Euclidean contact region error or Geodesic contact region error respectively. We find the vertex in P_r that is furthest from the desired contact region and report the distance as the error. These steps are visually illustrated in Fig. 4. Since our contact metric is based on vertices on the object mesh, we also report the Euclidean distance between the two closest vertices in the object mesh as G_{min} to give readers an idea of accuracy of the contact error. We provide software to compute all of our proposed error metrics at our associated website.

B. Details of Setup

Researchers following the protocol need to report the robotic platform used and any additional information about the experimental setup. Object's weights, if different from what was given in [7], must be reported. Any noise in

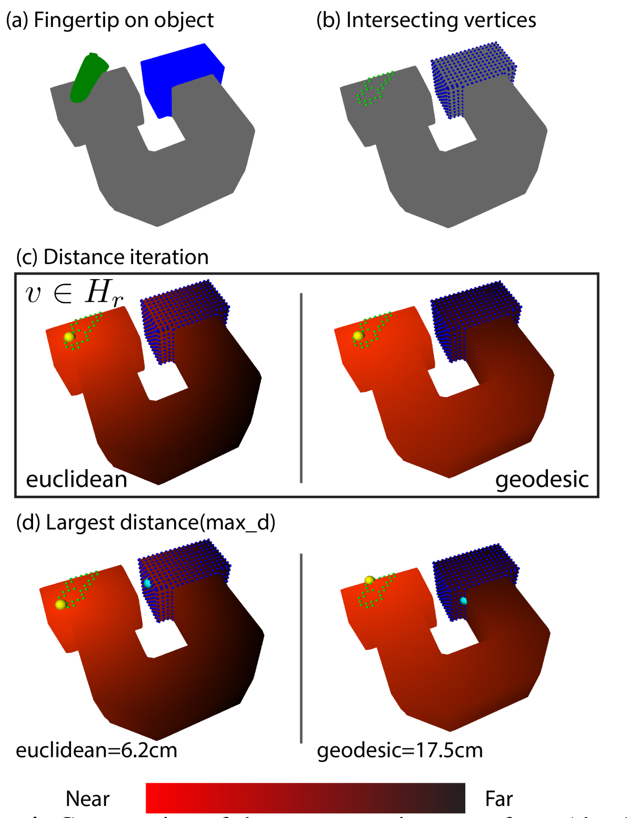


Fig. 4: Computation of the contact region error from Alg. 1 is illustrated. The reached fingertip is shown as the green mesh on the object (gray mesh) is shown in (a) along with the desired contact region in blue. (b) shows intersecting vertices computation (lines 7-10).(c) shows the computation of distance from one vertex of the fingertip as a heatmap along the object surface. The shortest distance from the intersecting vertices of the fingertip P_r to the desired contact region V_c is shown in (d). The cyan vertex in the desired contact region is the closest vertex to the robot hand.

the perception system should be mentioned along with a discussion of if it affects the results, if any.

C. Results to Report

Researchers are encouraged to submit their initial and desired grasp sets along with object names for enabling others to use the same grasps for direct comparison. Additionally, we require the reporting of different values in the conducted experiments, which vary depending on the chosen task level to address:

1) The error between the initial hand pose and the "human-setup" hand pose, computed as err_{pos}, err_{or} % (Level I, III), and G_{euc}, G_{geo} (Level II, III).
2) The error between the reached grasp and the desired grasp, computed as err_{pos}, err_{pos}%, err_{or}% (Level I, III), and G_{euc}, G_{geo}, as well as G_{min} for accuracy of contact error (Level II, III).
3) The time spent to plan and execute the in-hand manipulation method across the chosen object set. If the chosen method requires an offline planning step, we suggest to report it separately from the execution time.
4) The percent of unsuccessful executions, specifying which trials failed and the cause (e.g. object dropped).

Apart from using these results to provide a quantitative analysis of their methods, we encourage researchers to submit the computed errors in the given website, where we provide code to visualize the errors as a box plot. In this box plot, the middle line defines the median error; the bottom and top borders indicate the first and third quartiles; the whiskers indicate the extremes of the inliers within 1.5 times the interquartile range. An example plot is shown in Fig. 7.

D. Comparing Results

Given the measurements proposed in section III-C, good methods have low errors, low planning and execution times, and low failure rates. Comparisons can be made between methods that address the same level of tasks, and level III can be compared with levels I and II with respect to their intersecting error types. Systems with similar hardware (e.g. parallel gripper vs parallel gripper, 3-finger hand vs 3-finger hand) can rely on the same error comparison. Alternatively, The tradeoff between hardware complexity and result accuracy can be considered as an additional measure for comparing the results of different methods.

Due to the highly heterogeneous nature of the systems we aim at evaluating, it is difficult to find an *absolute best*. As such, we believe users can compare the different methods according to the reported results and confront them with the requirements of the system. In particular, this allows for a choiche of the *relative best* system for the particular application or requirements.

IV. Demonstration

We benchmark methods for level I and level III tasks in Sec. IV-A and Sec. IV-B respectively. Since the level III task demonstrate the metrics used for level II tasks, we do not explicitly demonstrate a separate method for level II. We refer readers to [19] for an example approach that could be easily extended for evaluation using the level II protocol.

A. Level I task

Using the proposed framework, we compare five different in-hand manipulation solutions using Level I tasks:

- The *relaxed-rigidity, relaxed-position, & relaxed-position-orientation* in-hand manipulation methods by Sundaralingam and Hermans [45]; these methods enable a robotic system to repose a grasped object without breaking/making new contacts on the object.
- The *IK-rigid* method, in which a rigid contact model between the object and the fingertips is assumed.
- The *point-contact* method, which assumes a point contact with friction model for the fingertips. That is, the contact position is assumed fixed, while the relative orientation can change. This is a simplification of the model formulated in [51].

The desired grasp is given by a desired palm pose P_d with respect to the object.

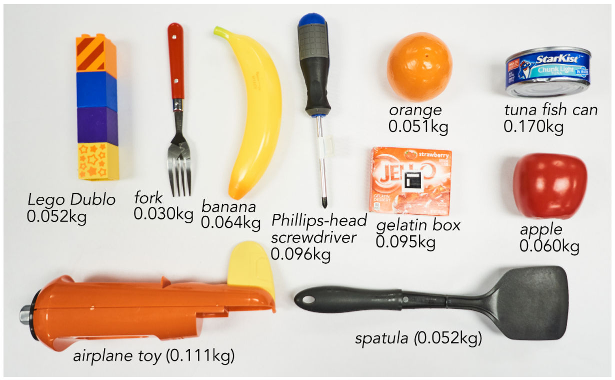


Fig. 5: Objects from the YCB dataset used in the experiments with corresponding labels and measured weights below them.

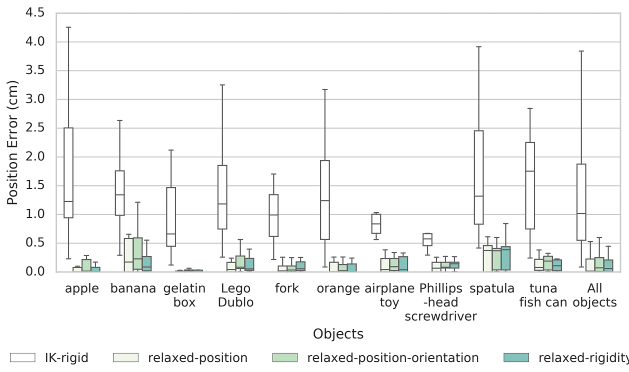


Fig. 6: Comparison of planning results across four Level I methods: *IK-rigid, relaxed-position, relaxed-position-orientation* and *relaxed-rigidity*. Results show the position error between the desired and expected final hand pose obtained by the planner (error in planning).

1) Setup Details: The methods are first compared within a trajectory optimization framework offline. Then, they are executed on the Allegro hand–a multi-fingered hand attached to a box frame. For the evaluation, we used ten objects from the YCB dataset shown in Fig. 5. The object and hand were tracked using ARUCO markers [52] using an ASUS XTION camera. A human initialized the object in the initial hand pose. We setup a rigid transformation between the human setup pose and the planned initial pose to account for the human error in the reached pose. Each generated trajectory was executed five times. We used two different initial grasps and five different desired grasps per object. Five trials were run for each generated trajectory, accounting for 50 executions per object. In total, 500 trajectories were executed on the robot per method. The goal positions range from 0.8 to 8.33cm, with a mean of 4.87cm, from their respective initial positions. The goal orientations range from 1.53% to 30.7%, with a mean of 11.96%, from their respective initial positions. The generated grasp sets are available at https://robot-learning.cs.utah.edu/project/in_hand_manipulation.

2) Results: For every trajectory that run on the robot, we record the position and orientation error. The median planning time across all objects were 14.8s, 4.4s, 0.5 s, 0.3s, and 0.5s for *point-contact*, *IK-rigid*, *relaxed-position*, *relaxed-position-orientation*, and *relaxed-rigidity* respectively. Since all trajectories are run without replanning, the execution time is fixed at 1.67s. As suggested in our benchmarking framework, the errors are plotted as a box plot (showing first quartile, median error, third quartile) with whiskers indicating the extremes of the inliers within 1.5 times the interquartile range. In all plots results correspond to objects grasped with three fingers. We will first report the error between the planned hand pose and the desired hand pose, followed by results on executing the generated trajectories on the real robot.

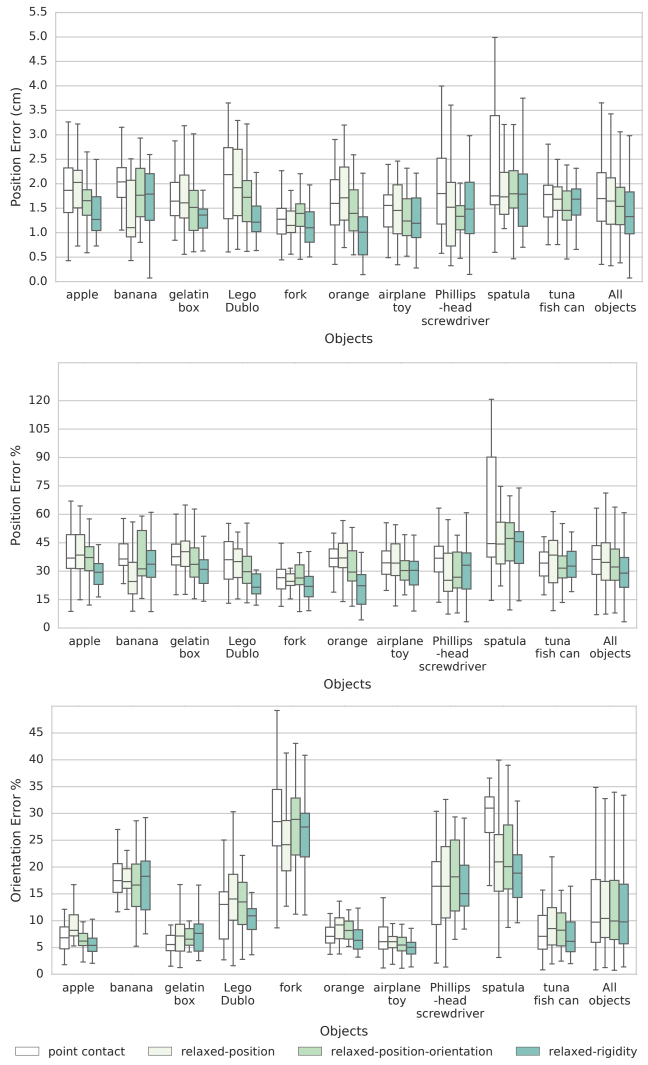


Fig. 7: A comparison of the different methods on real-world executions. Top: Position error Middle: Position error% Bottom: Orientation error%. The median position error decreases for all objects with the *relaxed-rigidity* method. Except for *banana* and *gelatin box*, the orientation error% improves for the *relaxed-rigidity* method for all objects.

We report the error between the planned hand pose and the desired hand pose across these methods: *IK-rigid, relaxed-position, relaxed-position-orientation* and *relaxed-rigidity*. We do not show offline results for the *point-contact* method as computing the object pose from the solution is not

Table II: Summary of results with the best value in bold text. The errors are the median of all trials.

Method	drops%	err_{pos} (cm)	err_{pos} %	err_{or}%
point-contact	5	1.69	36.81	**9.74**
relaxed-position	9	1.64	30.95	10.43
relaxed-position-orientation	7	1.54	29.19	9.84
relaxed-rigidity	**0**	**1.32**	**28.67**	9.86

Table III: Metrics for the DMG method.

Object	*gelatin box*	*cracker box*	*spatula*	*potted meat can*
err_{pos} (cm)	0.505	0.267	0.513	0.610
err_{pos}%	9.9	2.7	8.8	11.0
err_{or}%	0.016	0.044	0.023	0.049
G_{euc} (cm)	1.125	0.862	0.746	0.663
G_{geo} (cm)	1.127	0.862	1.505	0.663
G_{min} (cm)	0.013	0.057	0.034	0.004
DMG time (s)	10.312	15.467	13.406	18.295
Plan time (s)	0.023	0.004	6.7e-05	0.002

possible since the optimization does not internally simulate the object's pose. However, we will report the results of *point-contact* method in the real-robot experiments. The errors are plotted in Fig. 6. It is evident that IK-rigid has difficulty reaching the desired object position, a result of the problem being over-constrained, as such we do not report experimental results for this method on the real robot.

The position error and orientation error for all trials across all objects are shown in Fig. 7. The *relaxed-rigidity* method has the lowest median position error across all objects. Its maximum error across all objects is also much smaller than the *point-contact* method. Additionally, one can see that the *relaxed-rigidity* method has a lower variance in final position than the competing methods across nearly all objects. We report the median errors and the percentage of object drops in Table II. The *relaxed-rigidity* method never dropped any object across the 500 trials that were executed while all other methods dropped the object at least 5 times. The median errors $[err_{pos}, err_{or}\%]$ between initial pose and human setup initial pose are [0.58cm, 3.44%] for *point-contact*, [0.47cm, 3.19%] for *relaxed-position*, [0.52cm, 3.32%] for *relaxed-position-orientation*, and [0.54cm, 3.55%] for *relaxed-rigidity*. The contact points based metrics, G_{euc} and G_{geo}, are not reported because these methods perform Level I tasks.

B. Level III task

In this section, we show example evaluations for tasks that involve both desired hand pose and desired contact points with the object. We use the Dexterous Manipulation Graph method (DMG), which is a planner for in-hand manipulation that is based on a graph representation of the object's surface, obtained through the object's discretization into small areas. The DMG contains information on possible motions of a finger on the object's surface. Through this graph, in-hand manipulation is planned as a sequence of rotations and translations. A detailed explanation of the method is found in [9]. Since the motion execution is treated separately from planning (e.g. it can use pushing against the environment, bi-manual pushing, etc.), we focus only on evaluating the planned solution.

1) Setup Details: The DMG planner is used to find an in-hand manipulation solution for an ABB Yumi smart gripper. We select some of the contact region tasks we defined in section II, and defined H_i and H_d accordingly. Since the DMG defined in [9] is designed for parallel grippers, two contact points per hand are defined. We also define initial and desired poses P_i, P_d. Due to the gripper's structure, its position can be derived using a translation from the middle point between the two fingertips. All the executed tasks are available on our website.

2) Results: Table III shows the results for planning in-hand manipulation paths. Each column corresponds to a different task with a different object. Each row shows the metrics proposed in section III-A, and the planning time. The evaluated method requires offline computation of the Dexterous Manipulation Graph structure, reported as DMG time. Once the offline step is executed, the planning time for the given tasks is also reported.

V. Conclusion

We proposed a benchmarking scheme for quantifying in-hand manipulation capabilities in a robotic system. We designed tasks for in-hand manipulation systems using the widely available YCB objects set, and we provided suggestions for adapting these tasks given the constraints of the hardware used for the evaluation. We have shown example results to demonstrate the outcome of the proposed benchmarking scheme. These results also serve as baselines for comparison with different methods in the future. By using this standardized evaluation we enable a comparison between different in-hand manipulation techniques that also considers different kinds of hardware platforms.

References

[1] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *IEEE Intl. Conf. on Robotics and Automation*, 2014, pp. 1578–1585.

[2] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *IEEE Intl. Conf. on Robotics and Automation*, 2016, pp. 378–383.

[3] H. Van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *IEEE-RAS Intl. Conf. on Humanoid Robotics*, 2015, pp. 121–127.

[4] M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *preprint arXiv:1808.00177*, 2017.

[5] R. Higo, Y. Yamakawa, T. Senoo, and M. Ishikawa, "Rubik's cube handling using a high-speed multi-fingered hand and a high-speed vision system," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2018, pp. 6609–6614.

[6] S. Ulbrich, D. Kappler, T. Asfour, N. Vahrenkamp, A. Bierbaum, M. Przybylski, and R. Dillmann, "The opengrasp benchmarking suite: An environment for the comparative analysis of grasping and dexterous manipulation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Sep. 2011, pp. 1761–1767.

[7] G. Kootstra, M. Popović, J. A. Jørgensen, D. Kragic, H. G. Petersen, and N. Krüger, “Visgrab: A benchmark for vision-based grasping,” *Paladyn*, vol. 3, no. 2, pp. 54–62, Jun 2012.

[8] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set,” *IEEE Robotics Automation Magazine*, vol. 22, no. 3, pp. 36–52, Sep. 2015.

[9] N. C. Dafle, R. Holladay, and A. Rodriguez, “In-hand manipulation via motion cones,” in *Robotics: Science and Systems*, Pittsburgh, Pennsylvania, 2018.

[10] S. Cruciani, C. Smith, D. Kragic, and K. Hang, “Dexterous manipulation graphs,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2018, pp. 2040–2047.

[11] B. Sundaralingam and T. Hermans, “Relaxed-rigidity constraints: Kinematic trajectory optimization and collision avoidance for in-grasp manipulation,” *Autonomous Robots*, vol. 43, no. 2, pp. 469–483, 2019.

[12] D. Rus, “Coordinated manipulation of objects in a plane,” *Algorithmica*, vol. 19, no. 1-2, pp. 129–147, 1997.

[13] Q. Li, C. Elbrechter, R. Haschke, and H. Ritter, “Integrating vision, haptics and proprioception into a feedback controller for in-hand manipulation of unknown objects,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2013, pp. 2466–2471.

[14] L. Han and J. C. Trinkle, “Dextrous manipulation by rolling and finger gaiting,” in *IEEE Intl. Conf. on Robotics and Automation*, vol. 1, 1998, pp. 730–735 vol.1.

[15] K. Lowrey, S. Kolev, J. Dao, A. Rajeswaran, and E. Todorov, “Reinforcement learning for non-prehensile manipulation: Transfer from simulation to physical system,” in *IEEE Intl. Conf. on Simulation, Modeling, and Programming for Autonomous Robots*, 2018.

[16] M. A. Roa, Z. Chen, I. C. Staal, J. N. Muirhead, A. Maier, B. Pleintinger, C. Borst, and N. Y. Lii, “Towards a functional evaluation of manipulation performance in dexterous robotic hand design,” in *IEEE Intl. Conf. on Robotics and Automation*, 2014, pp. 6800–6807.

[17] L. U. Odhner and A. M. Dollar, “Dexterous manipulation with underactuated elastic hands,” in *IEEE Intl. Conf. on Robotics and Automation*, 2011, pp. 5254–5260.

[18] P. Michelman and P. Allen, “Compliant manipulation with a dextrous robot hand,” in *IEEE Intl. Conf. on Robotics and Automation*, 1993, pp. 711–716.

[19] D. Rus, “In-hand dexterous manipulation of piecewise-smooth 3-d objects,” *Intl. Journal of Robotics Research*, vol. 18, no. 4, pp. 355–381, 1999.

[20] B. Sundaralingam and T. Hermans, “Geometric in-hand regrasp planning: Alternating optimization of finger gaits and in-grasp manipulation,” in *IEEE Intl. Conf. on Robotics and Automation*, 2018.

[21] E. Psomopoulou, D. Karashima, Z. Doulgeri, and K. Tahara, “Stable pinching by controlling finger relative orientation of robotic fingers with rolling soft tips,” *Robotica*, vol. 36, no. 2, pp. 204–224, 2018.

[22] B. Calli and A. M. Dollar, “Vision-based model predictive control for within-hand precision manipulation with underactuated grippers,” in *IEEE Intl. Conf. on Robotics and Automation*, 2017, pp. 2839–2845.

[23] M. V. Liarokapis and A. M. Dollar, “Learning task-specific models for dexterous, in-hand manipulation with simple, adaptive robot hands,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2016, pp. 2534–2541.

[24] N. Rojas, R. R. Ma, and A. M. Dollar, “The gr2 gripper: An underactuated hand for open-loop in-hand planar manipulation,” *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 763–770, 2016.

[25] N. Chavan-Dafle, K. Lee, and A. Rodriguez, “Pneumatic shape-shifting fingers to reorient and grasp,” *preprint arXiv:1809.08420*, 2018.

[26] N. Rahman, L. Carbonari, M. D’Imperio, C. Canali, D. G. Caldwell, and F. Cannella, “A dexterous gripper for in-hand manipulation,” in *IEEE Intl. Conf. on Advanced Intelligent Mechatronics*, 2016, pp. 377–382.

[27] W. G. Bircher, A. M. Dollar, and N. Rojas, “A two-fingered robot gripper with large object reorientation range,” in *IEEE Intl. Conf. on Robotics and Automation*, 2017, pp. 3453–3460.

[28] A. J. Spiers, B. Calli, and A. M. Dollar, “Variable-friction finger surfaces to enable within-hand manipulation via gripping and sliding,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4116–4123, 2018.

[29] B. Calli, K. Srinivasan, A. Morgan, and A. M. Dollar, “Learning modes of within-hand manipulation,” in *IEEE Intl. Conf. on Robotics and Automation*, 2018, pp. 3145–3151.

[30] R. Antonova, S. Cruciani, C. Smith, and D. Kragic, “Reinforcement learning for pivoting task,” *preprint arXiv:1703.00472*, 2017.

[31] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, “Learning complex dexterous manipulation with deep reinforcement learning and demonstrations,” in *Robotics: Science and Systems*, Pittsburgh, Pennsylvania, 2018.

[32] R. Akrour, F. Veiga, J. Peters, and G. Neumann, “Regularizing reinforcement learning with state abstraction,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2018, pp. 534–539.

[33] I. Mordatch, E. Todorov, and Z. Popović, “Discovery of complex behaviors through contact-invariant optimization,” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 43, 2012.

[34] F. E. Viña, Y. Karayiannidis, C. Smith, and D. Kragic, “Adaptive control for pivoting with visual and tactile feedback,” in *IEEE Intl. Conf. on Robotics and Automation*, 2016, pp. 399–406.

[35] A. Sintov, O. Tslil, and A. Shapiro, “Robotic swing-up regrasping manipulation based on the impulse-momentum approach and clqr control,” *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1079–1090, 2016.

[36] S. Cruciani and C. Smith, “In-hand manipulation using three-stages open loop pivoting,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2017, pp. 1244–1251.

[37] J. Shi, J. Z. Woodruff, P. B. Umbanhowar, and K. M. Lynch, “Dynamic in-hand sliding manipulation,” *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 778–795, 2017.

[38] Z. J. Yifan Hou and M. T. Mason, “Fast planning for 3d any-pose-reorienting using pivoting,” in *IEEE Intl. Conf. on Robotics and Automation*, 2018, pp. 1631–1638.

[39] D. Almeida and Y. Karayiannidis, “Dexterous manipulation with compliant grasps and external contacts,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2017, pp. 1913–1920.

[40] N. Furukawa, A. Namiki, S. Taku, and M. Ishikawa, “Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system,” in *IEEE Intl. Conf. on Robotics and Automation*, 2006, pp. 181–187.

[41] R. K. Katzschmann, A. D. Marchese, and D. Rus, “Autonomous object manipulation using a soft planar grasping manipulator,” *Soft robotics*, vol. 2, no. 4, pp. 155–164, 2015.

[42] N. Chavan-Dafle and A. Rodriguez, “Stable prehensile pushing: In-hand manipulation with alternating sticking contacts,” in *IEEE Intl. Conf. on Robotics and Automation*, 2018, pp. 254–261.

[43] B. Calli, A. Kimmel, K. Hang, K. Bekris, and A. Dollar, “Path planning for within-hand manipulation over learned representations of safe states,” in *Intl. Symp. on Experimental Robotics*, 2018.

[44] B. Calli and A. M. Dollar, “Robust precision manipulation with simple process models using visual servoing techniques with disturbance rejection,” *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 406–419, 2018.

[45] B. Sundaralingam and T. Hermans, “Relaxed-rigidity constraints: In-grasp manipulation using purely kinematic trajectory optimization,” in *Robotics: Science and Systems*, 2017.

[46] P. Tournassoud, T. Lozano-Perez, and E. Mazer, “Regrasping,” in *IEEE Intl. Conf. on Robotics and Automation*, 1987.

[47] D. Q. Huynh, “Metrics for 3d rotations: Comparison and analysis,” *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[48] K. Crane, C. Weischedel, and M. Wardetzky, “The heat method for distance computation,” *Commun. ACM*, vol. 60, no. 11, pp. 90–99, 2017.

[49] S. Gottschalk, M. C. Lin, and D. Manocha, “Obbtree: A hierarchical structure for rapid interference detection,” in *Conf. on Computer graphics and interactive techniques*, 1996, pp. 171–180.

[50] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, “Fast distance queries with rectangular swept sphere volumes,” in *IEEE Intl. Conf. on Robotics and Automation*, vol. 4, 2000, pp. 3719–3726.

[51] Z. Li, P. Hsu, and S. Sastry, “Grasping and coordinated manipulation by a multifingered robot hand,” *Intl. Journal of Robotics Research*, vol. 8, no. 4, pp. 33–50, 1989.

[52] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280 – 2292, 2014.