GRASPA-fying the Panda

Easily Deployable, Fully Reproducible Benchmarking of Grasp Planning Algorithms

By Fabrizio Bottarel, Alessandro Altobelli, Ugo Pattacini, and Lorenzo Natale

Although robotic grasp planning has been extensively studied in the literature, comparing the performance of different approaches still proves challenging due to the lack of standardization in hardware setup and benchmarking protocols. This work addresses the issue with a threefold contribution. First, it provides a standardized hardware platform and a software framework integrating a benchmarking protocol for grasp planning algorithms (GRASPA). Second, it uses such a framework to benchmark three state-of-the-art algorithms in a reproducible way. Third, it employs the framework to investigate the effect of camera pose variance in visual-based grasp planning. We show how the proposed benchmarking setup can be used to provide insight into the results, not only to compare different vision-based grasp planners but also to evaluate different parameter configurations within the same grasp planner, for instance, camera viewpoint with respect to the scene. To ease the reproducibility of our results and usability of the platform, we provide extensive information for replicating the experimental setup and installing our software in the supplementary material (available at https://doi.org/10.1109/MRA.2023.3256272). All the software used in this article is freely available online in the form of Docker images.

Digital Object Identifier 10.1109/MRA.2023.3256272
Date of current version: 3 April 2023

INTRODUCTION

Grasping is usually considered a stepping stone for many robotic tasks involving manipulation as a means of interaction with the environment. Its importance as a research field is pivotal to robotics and has seen active study and exploration for decades. Nevertheless, comparing the performance of different approaches to grasp planning in a rigorous and replicable way still proves to be challenging. This can be blamed on the scarcity of completely replicable benchmarking protocols and a lack of standardization in terms of tasks, setups, and metrics.

Despite benchmarks such as Yale-CMU-Berkeley (YCB) [1] having gained traction in vision and robotics literature, identifying benchmarking tasks and scenarios is still an open problem. Competitions such as the Amazon Picking Challenge [2] and RoboCup@Home [3] aim at comparing different grasping strategies in increasingly challenging scenarios, however, the tasks themselves are often difficult to reproduce and the number of competing teams is typically small. To address these limitations, a cloud-based competition [4] was recently

proposed to benchmark manipulation algorithms, although using a fixed, remote platform could reduce the scenario-customization potential.

Even though there are no widely adopted standard protocols and platforms for benchmarking grasp planners, recent literature contains a number of significant contributions. Bekiroglu et al. [5] present a method for comparing the performance of several grasp planning algorithms, even though the proposed metrics do not encompass a way to evaluate the grasp quality nor the capabilities of the robot being used. In [6], the authors focus on end-effector performance by comparing a number of hands and grippers on grasping tasks. Although this work presents an in-depth benchmark of features like grasp and finger strength, it does not factor in robot arm capabilities nor does it consider grasp synthesis in its methodology. Concerning reproducibility, a reproducible low-cost arm benchmark [7] was proposed as a low-cost platform to benchmark grasping pipelines. On the one hand, this constitutes a substantial contribution to the needs of the manipulation research community, on the other, it carries two main limitations. The first being the absence of metrics to evaluate grasp quality, the second being the limited range of shapes and sizes in the object set. With GRASPA [8], we made our own attempt at addressing the need for replicable benchmarking protocols for real robot setups. It provides reproducible layouts and a set of metrics to assess performance of the overall grasping pipeline and the contribution of each component. The protocol is adaptable to different robotic setups and has been deployed on the iCub humanoid robot. At the time of writing this manuscript, GRASPA layouts are also being adopted as a base for BURG-Toolkit [9], a set of open source tools to further enhance reproducibility of scenes for grasp planning tasks.

This article builds upon our previous work with a threefold aim. First, it presents and describes a robotic setup and software framework compliant with GRASPA specifications that can be used for benchmarking of grasp planners. Second, it provides a reproducible set of experiments to compare the performance of three planners according to GRASPA indices. Finally, it investigates the impact of camera pose on the performance of the grasp synthesis process. To maximize the impact on the community, the proposed benchmarking platform is based on hardware currently widespread in the field of robotic manipulation, i.e., the Franka Emika Panda robot arm and Intel RealSense red, green, blue plus depth (RGB-D) cameras. We also provide a Robot Operating System (ROS)-based software framework that can be easily extended to integrate and test other vision-based grasp planning approaches.

The "Benchmarking Setup and Platform" section outlines the main hardware and software components of our replicable benchmarking setup. The "Grasp Planners" section briefly describes the three different grasp planning algorithms that have been benchmarked in this work. The "Experimental Data Collection" section explains the experimental methodology and details how the data are collected. Finally, in the "Results and Discussion" section, such results are presented in the form of GRASPA indices and discussed in detail. The accompanying document provides a step-by-step tutorial to reproduce the proposed robotic platform, install the software, and replicate our experiments.

BENCHMARKING SETUP AND PLATFORM

In this section, we provide an end-to-end overview of the hardware and software platform we propose for benchmarking grasp planning algorithms on real robots. Detailed information on how to reproduce the experimental setup, install all the needed software, and perform experiments is provided in the supplementary material (available at https://doi.org/10.1109/MRA.2023.3256272).

A ROBOTIC PLATFORM

The components of the robotic platform (displayed in Figure 1) were selected by taking into account the following three main aspects:

1) Grasp planners are usually designed to address precision picking tasks using manipulators equipped with parallel jaw grippers.
2) Visual-based grasp planners are usually designed to consume 2D, 2.5D, or 3D visual data.
3) Individual components must be either already widespread in the research community, inexpensive and available in the market, or easily manufacturable (e.g., through 3D printing).

We chose a Franka Emika Panda seven-degree-of-freedom (7-DoF) robot arm as the manipulator, equipped with the

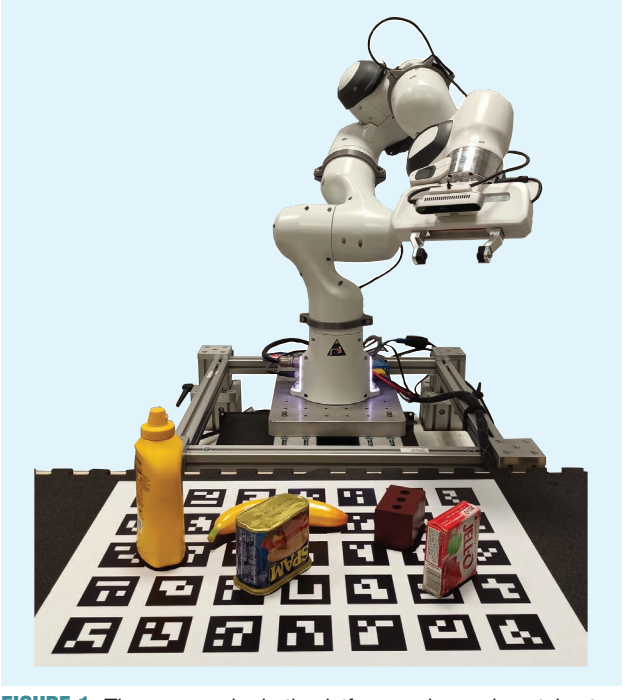

FIGURE 1. The proposed robotic platform and experimental setup used to benchmark GRASPA. It comprises a seven-degree-of-freedom (7-DoF) Franka Emika Panda arm, a Franka Hand gripper, an Intel RealSense D series RGB-D camera, and a custom marker board. The system is controlled via a number of open source ROS packages. The experimental setup involves a subset of YCB objects used as manipulation targets.

default Franka Hand, as the backbone of the platform. An Intel RealSense D415 (although other models of the D RealSense series are compatible) RGB-D camera was attached to the end effector with a custom 3D-printed mount. An additional tripod-mounted RealSense camera is also used for visual calibration assessment (see the "Camera-Calibration Score *S*1" section). The rest of the benchmarking setup comprises a 3D-printable marker support structure and printable marker boards.

BENCHMARKING SOFTWARE ARCHITECTURE

Vision-based grasp planning algorithms consume visual input data and produce grasp candidates (in the form of end-effector configurations) to be executed. To reuse as many software modules as possible and standardize the pipeline (outlined in Figure 2), we designed and implemented a lightweight software framework to be used for benchmarking on the platform defined in the "Robotic Platform" section. Such a framework provides

- base Python classes and helper functions for input and output data (respectively, visual data and 6D grasp candidates). It also defines a common interface class whose methods can be implemented to wrap existing implementations of grasp planners. Such methods involve operations such as planning grasps given visual data, storing configuration data, or displaying planned grasps in a GUI.
- an ROS node template to expose the functionality of the aforementioned Python classes with the rest of the ROS pipeline via standardized messages and services.

In this work, we used our framework to integrate and benchmark three popular grasp planners: Dex-Net [10], grasp pose detection in point cloud (GPD) [11], and 6-DoF Graspnet [12]. Additionally, we provide software for each component of the grasp planning and benchmarking pipeline (in Figure 2) deployed on the proposed robotic platform. Point cloud segmentation is performed by an ROS node, which consumes 3D point clouds coming from the RGB-D camera, and filters them to separate the object point clouds from the rest of the workspace. This is obtained by first cropping the input point cloud around the robot's workspace and then fitting a plane with random sample consensus iterations to detect and finally remove the table surface. The filtered visual data, together with camera parameters (extrinsics and intrinsics), are then passed to the grasping benchmark manager's ROS node whose purpose is to ensure coherent communication between the grasp planners and the motion planning stack through standardized service requests. The motion planning stack is a ROS node built on top of the MoveIt! framework, and it takes care of implementing motion primitives (e.g., homing, grasping, and cartesian motion) and exposing them to the rest of the pipeline. In particular, its main function is to generate feasible, collision-free joint trajectories given 6D goals in the Cartesian space. Finally, an ROS node based on the Augmented Reality University of Cordoba OpenCV library performs GRASPA board detection using RGB images from the in-hand camera to estimate the layout pose in the robot's root reference frame.

GRASPA ON THE PANDA PLATFORM

We employ GRASPA v1.0 [8] as a benchmark for grasp planners. GRASPA defines reproducible setup conditions, an experimental protocol, and a set of metrics to evaluate the performance of grasping pipelines, accounting for platform limitations that might hinder the overall performance. The benchmark specifies a subset of the YCB object set as graspable targets [see Figure 3(a)], selected to encompass a range of shapes, dimensions, and challenges. Reproducible experimental conditions are specified in the form of three scenarios of increasing complexity in terms of number, shape, and pose of the included objects [see Figure 3(b)–(d)]. To ease physical reproducibility, each printable layout board (594 × 420 mm, A2 standard paper size) includes object "footprints," as seen in Figure 3(e). In compliance with the GRASPA experimental protocol, the grasp planner under examination must generate and execute a number of grasps for each graspable target and ensure grasp stability by performing a trajectory while

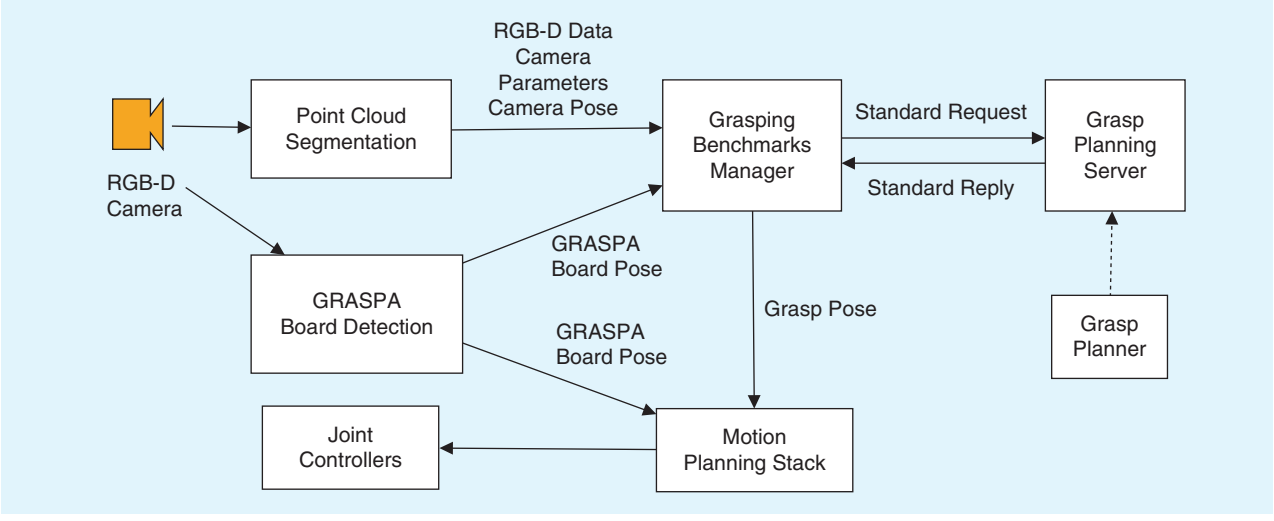

FIGURE 2. A functional diagram of the benchmarking pipeline as deployed on the Franka Emika Panda robotic platform.

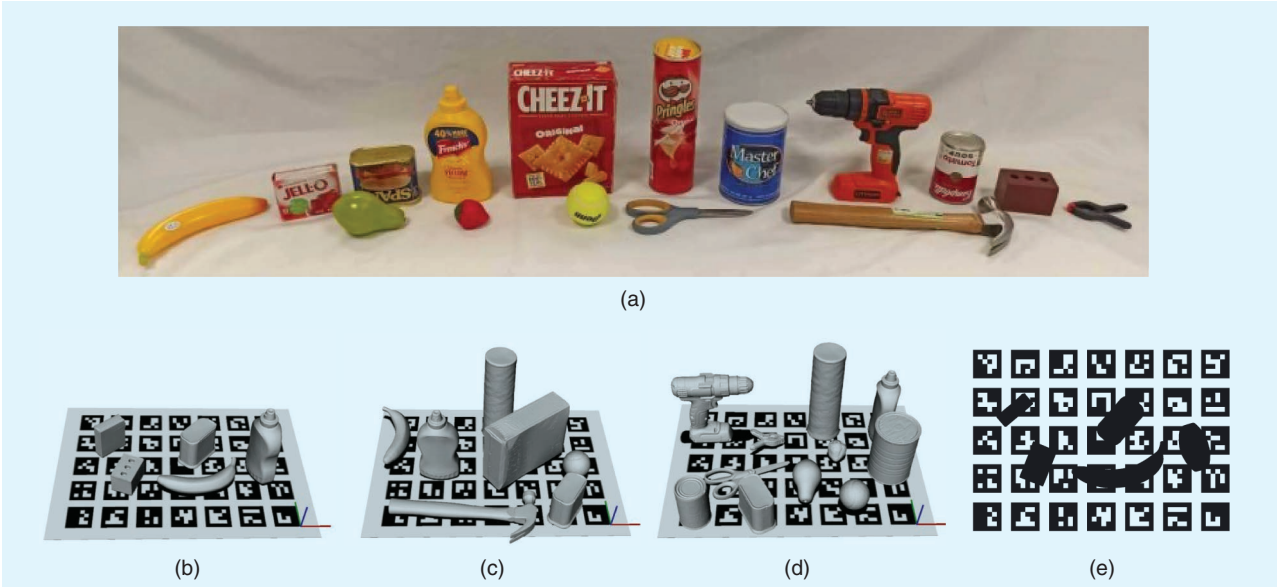

FIGURE 3. (a) The YCB object subset used in GRASPA and (b)–(d) GRASPA layouts. Each layout can be reproduced by printing the marker board, including a set of object footprints (relative to each specific layout). For instance, (e) can be printed to reproduce (b).

the object is being held. Moreover, GRASPA specifies a procedure to evaluate camera calibration accuracy and the robot reachability over the marker board workspace to assess any limitations of the robotic platform. The benchmarking protocol summarizes the performance of grasp planners through a number of metrics, reported in Table 1 for ease of reference. We recommend consulting the original work [8] for further details.

Although the iCub humanoid (used as a platform in the original GRASPA article) and the Franka Panda arm are very different robotic platforms, there are only two main differences when it comes to the benchmarking procedure. First, the Panda setup cannot visually assess the position of the end effector with the in-hand camera, requiring an additional setup-mounted camera (further explained in the "Reachability Score $S0$" and "Camera-Calibration Score $S1$" sections). Second, the Franka Hand has to be integrated in the Simox [13] GraspStudio library to analyze the Grasp Wrench Space (GWS) [14] of each grasp. The collision engine employed internally by Simox can detect only one contact point for each mesh pair, and a hard finger (point contact with friction) contact model is assumed. Given this, and the Franka Hand model comprising three meshes, parallel-jaw grasps produce at most two (almost) collinear contact normals in simulation. Given the contact model, no torque can be applied by the grasp around

TABLE 1. A description of the GRASPA benchmark metrics as defined in [8]. Footer k refers to the kth object in layout L.

SCORE	SCORE NOMENCLATURE	BRIEF DESCRIPTION
$S0_k^L \in [0,1]$	Reachability score	Accounts for whether the object is located in a region characterized by a good reachability of the robot
$S1_k^L \in [0,1]$	Camera calibration score	Accounts for whether the object is located in a region characterized by a good calibration of the vision system
$S2_k^L \in \{0,1\}$	Graspability score	Accounts for whether the object can be physically grasped and lifted by the robot, considering its shape and weight
$\overline{S3}_k^L \in [0,1]$	Grasp quality score	Accounts for how contacts are distributed on the object by simulating grasp closure in simulation and computing the grasp wrench space
$\overline{S4}_k^L \in [0,1]$	Binary success score	Accounts for whether the robot actually managed to grasp the object in real tests
$\overline{S5}_k^L \in [0,1]$	Grasp stability score	Evaluates the stability of the grasp during the execution of a fixed trajectory
$\overline{S6}_k^L \in [0,1]$	Obstacle avoidance score	(Only in cluttered mode) Accounts for how many objects the robot has hit while executing the grasp
$\bar{S}_k^L \in [0,2]$	Compound per object score	Combines all the scores to evaluate the grasping pipeline performance, taking into account any limitation of the robotic platform used in real-world tests
$\bar{S}_L \in [0,2]$	Compound per layout score	Combines scores for each graspable object in layout L

such normals and one dimension of the GWS collapses, causing null grasp quality. To address this issue without changing the inner workings of Simox, we force multiple contacts on each fingertip by splitting the collision mesh [Figure 4(b)] to approximate a soft finger contact model. This is further discussed in the "Grasp Quality *S*3" section.

REPRODUCIBILITY

Every part of the benchmarking platform proposed in this work has been designed with ease of reproducibility in mind. The robot arm, end effector, and cameras have been selected for their availability in the current market and their large diffusion in the robotics research community. YCB objects were chosen in GRASPA due to the set being already widespread in the manipulation research community. The rest of the physical setup (i.e., marker boards and 3D printer parts) was designed to be easily manufacturable with consumer-grade equipment. All the software for actuating the robot, replicating experiments, and computing results is hosted on GitHub and packaged in freely available and easily deployable Docker containers to minimize installation effort by the user. Detailed setup and operative instructions can be found in the supplementary downloadable material available at https://doi.org/10.1109/MRA.2023.3256272.

GRASP PLANNERS

Given the recent rise in popularity and performance of data-driven grasp planning approaches, especially when dealing with partial views of unknown objects, we focus on benchmarking three visual-based planners drawn from the state of the art of such a field [15], [16]. We selected Dex-Net [10], GPD [11], and 6-DoF Graspnet [12] for their impact on the community, state of the published code, and adaptability to our framework. Each uses visual data to produce an ordered list of grasp candidates, ordered using a quality metric specific to the approach. We hereby provide a brief description of the three planners, highlighting their differences (see also Table 2). Unless otherwise stated, we use the implementation of each approach provided by the original authors.

DEX-NET

Dex-Net consumes visual data in the form of depth maps and was originally intended to perform bin-picking tasks in cluttered scenes. It requires a setup where the camera is mounted on the top of the scene, with the image plane parallel to the surface, and produces 4-DoF planar grasps (i.e., the approach axis is orthogonal to the horizontal scene plane) that are parameterized with 3D position and angle around the approach axis. Dex-Net has seen different revisions and the addition of features in recent years (e.g., suction cup instead of a parallel jaw gripper structure, or a mix of the two with different policies), but at its core it is an end-to-end grasping pipeline trained on a synthetic grasp database that uses convolutional neural networks (CNNs) to extract features from depth maps. An example of a Dex-Net-generated grasp can be seen in Figure 5(a).

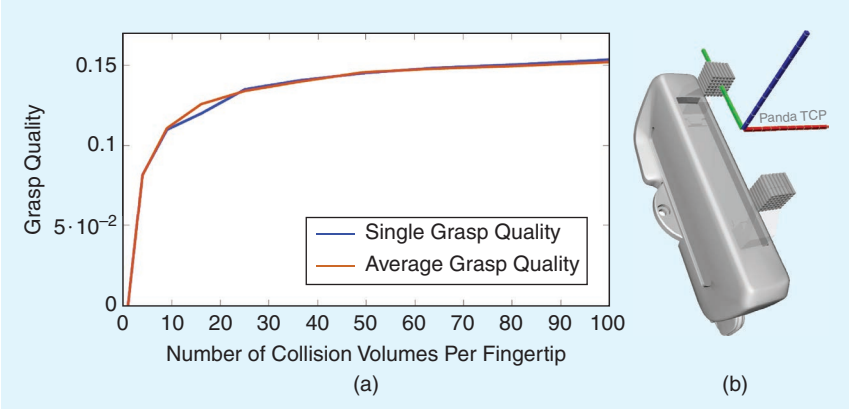

FIGURE 4. Grasp quality index values (GWS) for an increasing number of collision volumes. To understand the trend, a batch of grasps was evaluated using the *S*3 GRASPA metric while varying the number of contacts on each fingertip. (a) *S*3 was evaluated for each number of contacts on a single grasp (blue line) or on average (red line) of *S*3 of the same grasp pose with random perturbations. (b) The collision volumes (7 × 7 matrix) used to model the gripper fingertip. The Tool Center Point reference frame is highlighted (approach axis in blue).

TABLE 2. A brief overview of the grasp candidate generators benchmarked in this article. These methods have been selected from the state of the art according to their popularity within the research community, code availability and usability, and adaptability to the platform considered in this work.

GRASP PLANNER	USES DEEP LEARNING	INPUT	OUTPUT
Dex-Net	Candidate generation, candidate ranking	Scene depth map	Planar grasps (4-DoF)
GPD	Candidate ranking	Scene point cloud (optionally segmented)	6-DoF grasps
6-DoF GraspNet	Candidate generation, candidate ranking, candidate refinement	Segmented scene point cloud	6-DoF grasps

GPD

This approach detects 6-DoF grasps assuming a parallel jaw gripper by consuming visual data in the form of point clouds. It works by uniformly sampling the point cloud (that can be either partial or complete) and creating a candidate in each sampled location by setting the approach direction to the surface normal. The grasp candidates are filtered out using a list of geometric criteria such as approach direction, gripper size, maximum aperture, and desired workspace. The quality of each candidate is estimated by considering the cloud points that fall into the grasped volume of the gripper and projecting them in different ways to obtain a tensor that is fed into a simple CNN. The output of the CNN is the grasp quality, and it is trained on a number of synthetic grasps. An example of a GPD-generated grasp can be found in Figure 5(b).

6-DoF GRASPNET

6-DoF Graspnet consumes partial object point clouds to produce 6-DoF grasps, assuming a parallel jaw gripper. This approach exploits a variational auto encoder (VAE) to encode the input point cloud in a latent space that is then sampled to obtain diverse sets of possible grasps for the target. An evaluator network is used on the samples to obtain a measure of grasp quality, which is iteratively improved by performing a number of back-propagation passes to align the grasp pose to the observed point cloud. The VAE and evaluator networks are trained in simulation but are shown to perform very well on real-world examples without further fine-tuning. An example of the grasp detected by 6-DoF Graspnet is shown in Figure 5(c).

6-DoF GRASPNET CANDIDATE FILTERING

The 6-DoF Graspnet implementation provided by the original authors generates candidates regardless of scene constraints (e.g., table surface and robot workspace). Hence, this is the only approach out of the three that often produces unfeasible grasps. As GPD and Dex-Net already have built-in mechanisms to avoid this, we add a simple candidate rejection filter to 6-DoF Graspnet for the sake of usability and fairness. By enclosing the Franka Hand CAD model in a bounding box [Figure 5(d)], grasps that would result in a collision with the table surface can be detected by checking whether any of the box corners are below the marker board height.

EXPERIMENTAL DATA COLLECTION

This section documents the experimental procedure carried out to set up the benchmarking platform and collect experimental data from it, following the GRASPA protocol. For the sake of clarity, we describe the process for each metric. Further details on how the metrics are computed from the experimental data can be found in the published GRASPA article [8].

REACHABILITY SCORE S0

This score evaluates the precision with which the robot attains a set of predefined configurations in the workspace. The robot setup was placed in front of the GRASPA board, as shown in Figure 6(a), positioning the end effector so that the gripper-mounted camera could detect the board pose ${}^{r}T_{b} \in \mathrm{SE}(3)$ with respect to the robot's reference frame. At that point, the robot can move its end effector toward the reachability targets and record the reached 6D poses using only the direct kinematics.

CAMERA-CALIBRATION SCORE S1

This metric assesses the precision attainable by the robot in reaching a given configuration, measured with visual feedback. To compute $S1$, the robot's Tool Center Point [(TCP) a reference point on the hand] must be brought in a fixed array of poses over the GRASPA marker board, and the reached poses must be assessed visually. On the proposed robotic platform, this involves using a second RealSense camera (denoted as c_2 in this section) because the in-hand camera (denoted as c_1) is fixed with respect to the TCP. We rigidly mount a fiducial marker on the Franka Hand to enable detection of the TCP with the tripod-mounted camera c_2 (see Figure 6). This setup allows the estimation of ${}^{b}T_{h}$, i.e., the TCP pose in the board reference frame

$$ {}^{b}T_{h} = {}^{b}T_{c_2}\ {}^{c_2}T_{h} \qquad {}^{b}T_{h} \in \mathrm{SE}(3) \tag{1} $$

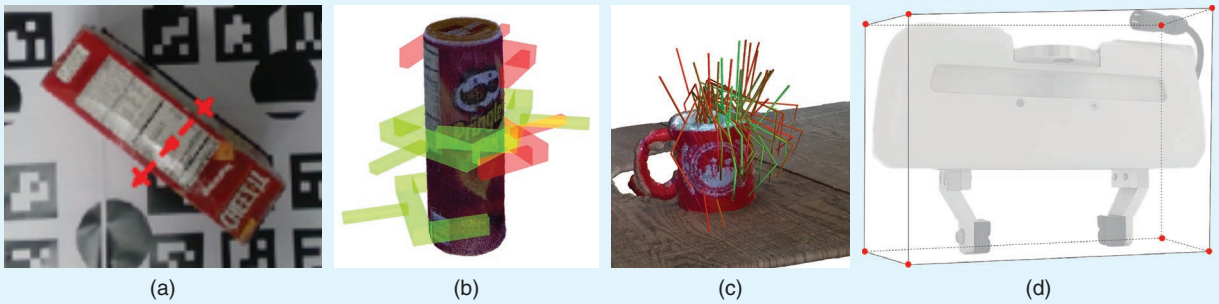

FIGURE 5. (a) The grasp pose examples collected by Dex-Net, (b) GPD, and (c) 6-DoF Graspnet. The check points [in red in (d)] used to filter 6-DoF Graspnet candidates.

which is the input for computation of $S1$. It is worth noting that, although camera c_2 is used to estimate the input for $S1$, the hand-mounted c_1 is used to acquire visual data for the grasp planning phase, and a miscalibration of the c_1 extrinsics may lead to inaccurate grasp executions. To show that $S1$ is actually evaluating errors in ${}^{h}T_{c_1}$ i.e., the extrinsic parameters of c_1, we can express ${}^{b}T_{h}$ through c_1

$$ {}^{b}T_{h} = {}^{h}T_{b}^{-1} = {}^{h}T_{c_1}\,{}^{c_1}T_{b} \qquad {}^{b}T_{h} \in \mathrm{SE}(3) \tag{2} $$

as c_1 is used to estimate the GRASPA board pose with respect to the TCP ($r_b^T \in \mathrm{SE}(3)$). Hence, if errors introduced by the marker estimation are negligible, ${}^{h}T_{c_1}$ is the only source of error that can be estimated in this stage.

It is worth noting that because both metrics $S0$ and $S1$ require guiding the robot's end effector through the same pose targets, data collection for both can be performed at the same time.

GRASPABILITY S2

For a parallel jaw gripper, object graspability is mainly limited by the maximum finger aperture, i.e., 0.08 mm for the Franka Hand. Any object that does not fit in the fully open hand with some orientation is deemed ungraspable by the Panda Hand.

GRASP QUALITY S3

The GRASPA protocol requires that five grasps are computed by each approach, for each object in each layout. As every grasp used to compute GRASPA metrics has to be executed, our pipeline queries the grasp plannner for a large number of candidates and executes the first kinematically feasible one. This ensures that candidates contributing to $S3$ also contribute to $S4$ and $S5$. It is worth remarking that $S3$ is computed in simulation, and it is not the same metric used by each algorithm to rank the proposed candidates.

As previously discussed in the "GRASPA on the Panda Platform" section, assuming a parallel jaw gripper kinematic and a hard finger contact model implies that the GWS volume is limited by the maximum distance between contact normals. This, coupled with the aforementioned limitations of the collison engine, causes a need to model the fingertip surface with a large number of different collision volumes. At one extreme, using a single contact for each fingertip causes the GWS to collapse along one dimension. At the other extreme, a large number (e.g., 100) of contacts for each fingertip exponentially increases the time needed for computing collisions and computing the GWS. As a tradeoff between the two extremes, we chose to model each fingertip with 49 contact volumes (7×7 grid), covering the whole fingertip pad surface. As shown in Figure 4(a), using 49 contacts instead of 100 leads to a 5% loss in grasp quality, which is an acceptable approximation for this use case.

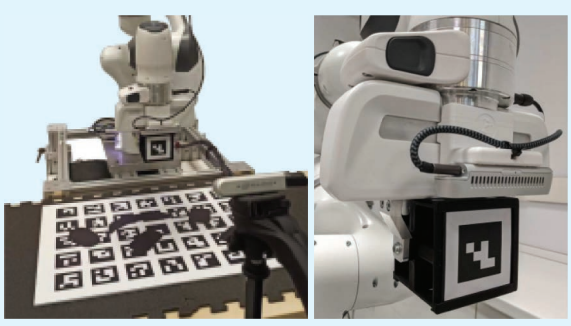

FIGURE 6. The Panda arm setup for the computation of metrics $S0$ and $S1$. (a) An additional camera was mounted in front of the robot to visually estimate (b) the end-effector pose.

BINARY SUCCESS AND STABILITY SCORES: S4 AND S5

Grasps are planned and executed for objects in isolation (no clutter). Grasp stability is assessed by lifting the object and moving it through the protocol stability trajectories. During the whole process, the force exerted by the gripper fingers is limited to 10 N to avoid damaging the objects while simultaneously applying enough normal force to cause surface friction.

GRASP SYNTHESIS WITH DIFFERENT CAMERA VIEWPOINTS

To investigate the effect of the camera pose on the performance of vision-based methods, we collect grasp candidates from the 6-DoF Graspnet pipeline using four different end-effector configurations, changing the height and the angle of the camera with respect to the GRASPA marker board. Such configurations are shown in Figure 7.

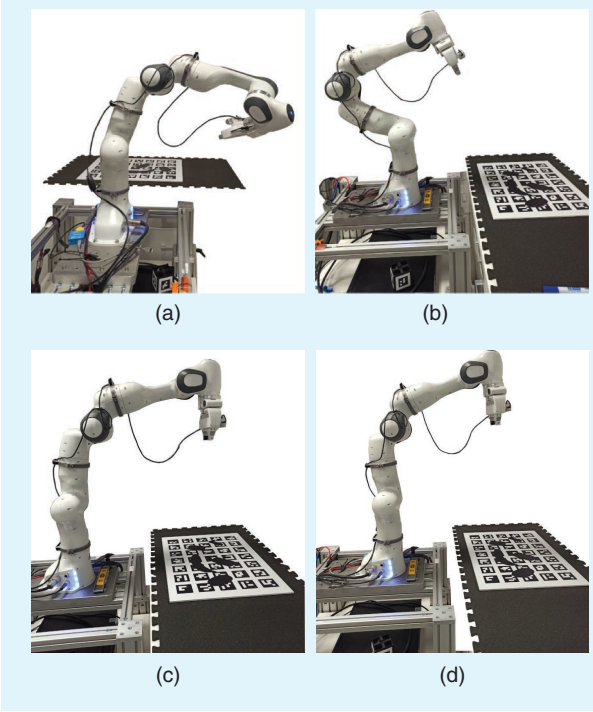

FIGURE 7. The robot configurations used to investigate the effect of camera pose on the performance of the 6-DoF Graspnet pipeline. Quantitative results are discussed in the "Results and Discussion" section and Table 4. (a) Camera pose 1, (b) camera pose 2, (c) camera pose 3, and (d) camera pose 4.

TABLE 3. GRASPA benchmarking results for GPD, Dex-Net, and 6-DoF Graspnet. The scores are computed according to the GRASPA protocol metrics. Cyan cells mark scores pertaining to the Mustard Bottle object. Green cells show the performance for the Chips Can object, for which only 6-DoF Graspnet can generate successful grasps. Orange cells show how a change in object orientation with respect to camera viewpoint affect the performance of 6-DoF Graspnet on the same object. Magenta cells mark the Hammer object, which proved to be challenging for every tested grasp planner. Yellow cells mark objects for which GPD is outperformed by the other tested approaches.

		COMMON			GPD					DEX-NET					6-DoF GRASPNET				
LAYOUT	OBJECT	$S0_k^L$	$S1_k^L$	$S2_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$
0	—	—	—	—	**1.12**	—	—	—	—	0.73	—	—	—	—	0.91	—	—	—	—
	Banana	1	1	1	—	0.15	1	1	1.15	—	0.11	1	1	1.11	—	0.16	1	1	**1.17**
	Foam Brick	1	1	1	—	0.14	1	1	**1.14**	—	0.05	1	0.88	0.93	—	0.17	0.6	0.6	0.7
	Gelatin Box	1	1	1	—	0.1	1	1	**1.1**	—	0.05	0.8	0.68	0.72	—	0.13	0.8	0.8	0.9
	Mustard Bottle	1	1	1	—	0.11	1	1	**1.11**	—	0.08	0.2	0.08	0.09	—	0.11	0.6	0.6	0.66
	Potted Meat Can	1	1	1	—	0.1	1	1	1.1	—	0.01	0.8	0.8	0.81	—	0.11	1	1	**1.11**
1	—	—	—	—	0.63	—	—	—	—	0.53	—	—	—	—	**0.82**	—	—	—	—
	Banana	0.92	1	1	—	0.16	0.8	0.7	0.88	—	0.18	0.8	0.8	0.94	—	0.18	0.8	0.8	**0.95**
	Chips Can	0.92	1	1	—	0	0	0	0	—	0	0	0	0	—	0.06	1	0.84	**0.9**
	Cracker Box	1	1	1	—	0.04	1	1	**1.04**	—	0.02	0.6	0.6	0.6	—	0.04	1	1	**1.04**
	Hammer	1	1	1	—	0.04	0.8	0.25	0.29	—	0.03	0.6	0	0.01	—	0.06	0	0	0
	Mustard Bottle	1	1	1	—	0.13	0.8	0.8	0.9	—	0.08	0.6	0.6	0.66	—	0.09	1	0.96	**1.05**
	Potted Meat Can	1	1	1	—	0.1	1	1	**1.1**	—	0.06	0.8	0.8	0.84	—	0.11	0.8	0.8	0.89
	Tennis Ball	1	1	1	—	0.2	0.2	0.2	0.23	—	0.15	0.6	0.6	0.68	—	0.13	0.8	0.8	**0.92**
2	—	—	—	—	**0.71**	—	—	—	—	0.56	—	—	—	—	0.63	—	—	—	—
	Chips Can	0.92	1	1	—	0	0	0	0	—	0	0	0	0	—	0.36	0.8	0.8	**0.83**
	Master Chef Can	1	1	0	—	N/A	N/A	N/A	N/A	—	N/A	N/A	N/A	N/A	—	N/A	N/A	N/A	N/A
	Medium Clamp	1	1	1	—	0.15	0.8	0.8	0.95	—	0.08	0.6	0.6	0.67	—	0.04	1	1	**1.04**
	Mustard Bottle	0.92	1	1	—	0.09	1	1	**1.09**	—	0.08	0.8	0.8	0.88	—	0.1	0.2	0.2	0.22
	Pear	1	1	1	—	0.09	0.2	0.2	0.24	—	0.09	0	0	0	—	0.15	0.8	0.8	0.95
	Potted Meat Can	1	1	1	—	0.1	1	1	**1.1**	—	0.07	0.8	0.8	0.87	—	0.1	0	0	0
	Power Drill	0.92	1	1	—	0.06	1	0.95	**1.01**	—	0.05	1	0.85	0.9	—	0.06	0.2	0.2	0.21
	Scissors	1	1	1	—	0.04	0.6	0.55	0.56	—	0.05	1	1	**1.05**	—	0.09	0.8	0.8	0.87
	Strawberry	1	1	1	—	0.18	0.6	0.5	0.59	—	0.23	1	1	**1.23**	—	0.22	1	1	1.22
	Tennis Ball	1	1	1	—	0.13	0.4	0.4	0.43	—	0.1	0	0	0	—	0.11	0.8	0.8	**0.91**
	Tomato Soup Can	1	1	1	—	0.09	1	1	**1.09**	—	0.07	0	0	0	—	0.04	0	0	0

N/A: not applicable.

RESULTS AND DISCUSSION

All the experimental data gathered according to the "Experimental Data Collection" section have been processed to obtain GRASPA performance metrics, as listed in Table 3. We hereby discuss these results, using GRASPA indexes to break down the overall performance scores of the benchmarked approaches. As highlighted by the reachability metric ($S0_k^L \in [0,1]$, the higher the better), the robot can easily reach the objects over the GRASPA board with a variety of orientations (Figure 8). Values of the camera calibration metric ($S1_k^L \in [0,1]$, the higher the better) confirm that camera calibration is also very reliable. The graspability index $S2_k^L$ shows that only one object (i.e., "Master Chef Can") was left out for being too large for the Franka Hand to grasp. Scores $S0$, $S1$, and $S2$ are strictly related to the hardware platform and are only reported once as all experiments were performed on the same setup. Thanks to multiple contacts modeling (the "Grasp Quality $S3$ section"), the grasp quality metric ($\bar{S}3_k^L \in [0,1]$) is consistent in range over all objects and never collapses to zero as long as force closure conditions are present.

In terms of raw grasp success performance, synthesized in $S4_k^L$ and $S5_k^L$ ($S4_k^L \in [0,1], S5_k^L \in [0,1]$, the higher the better), all the algorithms seem to be good performers. GPD shows the best overall performance, i.e., the underlined values of $\bar{S}_L \in [0,2]$ in Table 3, followed by 6-DoF Graspnet and Dex-Net. The gap between Dex-Net and the others is also confirmed by analyzing the results on the single objects and layouts, i.e., $\bar{S}_k^L \in [0,2]$. A possible explanation for this lies in the design of the approaches themselves. Dex-Net is a top-down planner providing 4-DoF candidates whose approach axis is always normal to the tabletop surface, while GPD and 6-DoF Graspnet detect full 6D grasp poses. This is especially advantageous for objects that are challenging to grasp from the top but are easily graspable from the side. This is particularly evident in the case of the "Mustard Bottle" (cyan cells in Table 3), whose cap is thin and slippery. Examples of this behavior can be seen in Figure 9. Nevertheless, although GPD performs very well in layout 0 and 2, 6-DoF Graspnet shows a significant edge over the others in layout 1, as suggested by its higher $\bar{S}_L$ value. An interesting case to discuss is the handling of "Chips Can." Despite the diameter of the can fitting in the Franka Hand (hence, it is deemed graspable), only 6-DoF Graspnet can generate good grasp candidates for it in any layout (highlighted in green in Table 3). To the best of our knowledge, there is no clear explanation for this behavior and it suggests further investigation is needed on the matter. Possibly, configuration parameters for each algorithm might be tweaked to compensate for this shortcoming in ways that are not evident in the existing documentation. Interestingly, Dex-Net and 6-DoF Graspnet generate the best grasps for some of the most challenging objects in terms of size or shape

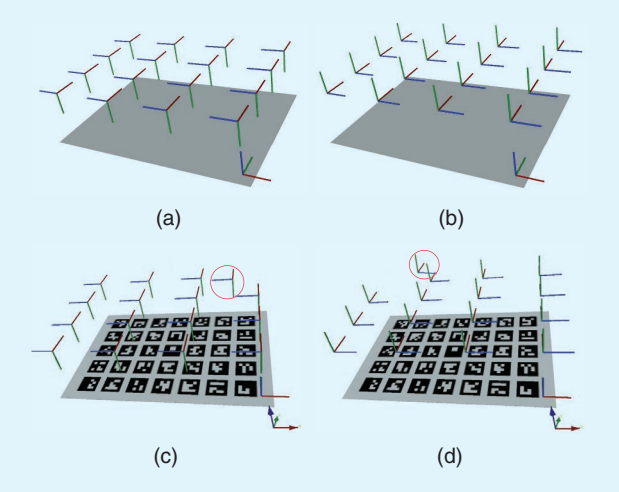

FIGURE 8. (a) and (b) Target poses and (c) and (d) the poses reached by the Panda arm during reachability tests. The blue axis corresponds to the end-effector approach axis (i.e., the axis perpendicular to the finger closure direction, passing through the center of the end effector), as highlighted in Figure 5(b). Except for one pose in (c) and one in (d), both are surrounded by a red circle, and the reached poses are close to the targets, showing good reachability. (a) Target pose set 0, (b) target pose set 2, (c) reached poses for set 0, and (d) reached poses for set 2.

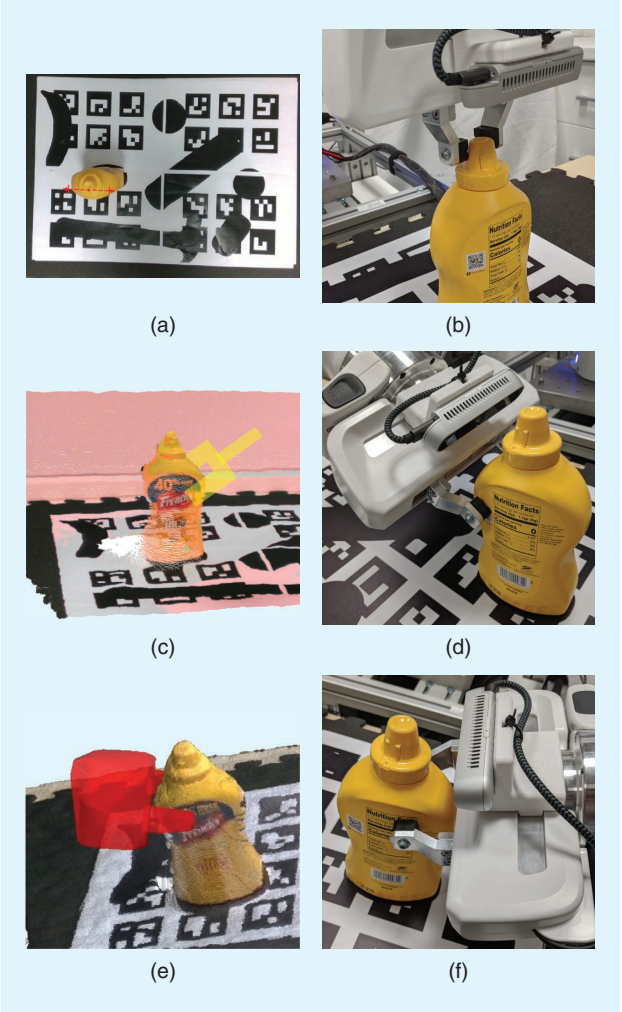

FIGURE 9. An example of the grasp candidates used to grasp the same objects. (a) and (b) A failed grasp from a Dex-Net candidate and (c) and (d) and (e) and (f) two examples of successful grasps obtained from GPD and 6-DoF Graspnet, respectively.

TABLE 4. GRASPA metrics obtained by testing 6-DoF Graspnet with four different camera viewpoints, as shown in Figure 7. The cells marked in orange show the only cases where this grasp planner performs best in pose 1, which is similar to the one used in the original article. The green cells show cases where the performance of the grasp planner on the same objects vary the most when the camera point of view is changed.

		POSE 1					POSE 2					POSE 3					POSE 4				
L.	OBJECT	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$	$\bar{S}_L$	$\overline{S3}_k^L$	$\overline{S4}_k^L$	$\overline{S5}_k^L$	$\bar{S}_k^L$
0	—	0.6	—	—	—	—	0.91	—	—	—	—	**1**	—	—	—	—	**0.95**	—	—	—	—
	Banana	—	0.17	1	1	**1.17**	—	0.16	1	1	1.16	—	0.15	1	1	1.15	—	0.17	1	1	**1.17**
	Foam Brick	—	0.11	0.6	0.6	0.7	—	0.17	0.6	0.6	0.7	—	0.16	1	1	**1.16**	—	0.16	0.8	0.8	0.93
	Gelatin Box	—	0	0	0	0	—	0.13	0.8	0.8	0.9	—	0.14	1	1	**1.14**	—	0.13	1	1	1.13
	Mustard Bottle	—	0.11	0.8	0.8	**0.89**	—	0.11	0.6	0.6	0.66	—	0.12	0.4	0.4	0.44	—	0.02	0.4	0.4	0.4
	Potted Meat Can	—	0.03	0.2	0.2	0.22	—	0.11	1	1	**1.11**	—	0.11	1	1	**1.11**	—	0.11	1	1	**1.11**
1	—	0.34	—	—	—	—	**0.82**	—	—	—	—	0.72	—	—	—	—	0.68	—	—	—	—
	Banana	—	0	0	0	0	—	0.18	0.8	0.8	0.95	—	0.17	1	1	1.17	—	0.14	1	1	**1.4**
	Chips Can	—	0.1	0.4	0.4	0.44	—	0.06	1	0.84	**0.9**	—	0.12	0.6	0.28	0.34	—	0	0	0	0
	Cracker Box	—	0.01	0.8	0.8	0.81	—	0.04	1	1	1.04	—	0.06	1	1	**1.06**	—	0.06	1	1	**1.06**
	Hammer	—	0	0	0	0	—	0.06	0	0	0	—	0.05	0	0	0	—	0.06	0.2	0.08	**0.09**
	Mustard Bottle	—	0.02	0	0	0	—	0.09	1	0.96	**1.05**	—	0.1	0.6	0.6	0.65	—	0.12	1	0.8	0.92
	Potted Meat Can	—	0.08	0.4	0.4	0.44	—	0.11	0.8	0.8	**0.89**	—	0.12	1	1	1.2	—	0.1	0.8	0.8	**0.89**
	Tennis Ball	—	0.09	0.6	0.6	0.7	—	0.13	0.8	0.8	**0.92**	—	0.16	0.6	0.6	0.7	—	0.08	0.6	0.6	0.67
2	—	0.54	—	—	—	—	0.63	—	—	—	—	0.59	—	—	—	—	**0.71**	—	—	—	—
	Chips Can	—	0.17	0.4	0.4	0.46	—	0.36	0.8	0.8	**0.83**	—	0	0.4	0.4	0.4	—	0.03	0.8	0.8	**0.83**
	Master Chef Can	—	N/A	N/A	N/A	N/A	—	N/A	N/A	N/A	N/A	—	N/A	N/A	0	N/A	—	N/A	N/A	N/A	N/A
	Medium Clamp	—	0.07	0.8	0.8	0.87	—	0.04	1	1	1.04	—	0.11	0.6	0.6	0.63	—	0.06	1	1	**1.06**
	Mustard Bottle	—	0.12	1	1	**1.13**	—	0.1	0.2	0.2	0.22	—	0.11	0.8	0.68	0.77	—	0.04	0.2	0.2	0.2
	Pear	—	0.04	0.4	0.4	0.4	—	0.15	0.8	0.8	**0.95**	—	0.12	0.4	0.4	0.45	—	0.17	0.8	0.8	0.93
	Potted Meat Can	—	0.12	0.6	0.6	0.67	—	0.1	0	0	0	—	0.12	1	1	**1.12**	—	0.1	1	1	1.1
	Power Drill	—	0.04	0	0	0	—	0.06	0.2	0.2	0.21	—	0.06	0.4	0.24	0.26	—	0.03	0.6	0.44	**0.46**
	Scissors	—	0.08	0.6	0.6	0.65	—	0.09	0.8	0.8	**0.87**	—	0.05	0.4	0.4	0.42	—	0.03	0.8	0.8	0.83
	Strawberry	—	0.23	0.6	0.6	0.73	—	0.22	1	1	**1.22**	—	0.15	0.4	0.4	0.49	—	0.23	0.8	0.8	0.99
	Tennis Ball	—	0.04	0.4	0.4	0.44	—	0.11	0.8	0.8	0.91	—	0.17	1	1	**1.17**	—	0.12	0.4	0.4	0.45
	Tomato Soup Can	—	0	0	0	0	—	0.04	0	0	0	—	0.09	0.2	0.2	**0.22**	—	0	0.2	0.2	0.2

(highlighted in yellow in Table 3). None of the tested algorithms could, however, generate good candidates for "Hammer" (highlighted in magenta). The small values of $S3$ (that evaluate the GWS) suggest this is caused by the peculiar density distribution of the object, corroborated by the fact that none of the three grasp planners explicitly evaluates global object properties such as center of mass.

We also observed that orientation of the objects has a relevant impact on performance. For instance, GPD obtained consistent $\bar{S}_k^L$ values for "Potted Meat Can" and "Mustard Bottle" over the three layouts, while the same cannot be said for 6-DoF Graspnet. As all areas of the GRASPA workspace can be reached with good accuracy (as demonstrated by the metrics $S0$ and $S1$), this effect is solely due to variability in the relative pose between the object and the camera. In fact, for vision-based grasp planners working with a single scene view, a variation in object pose is equivalent to a change in the camera's viewpoint as different object parts might provide different grasps with different quality. To investigate this, we considered the difference in GRASPA indicators for 6-DoF Graspnet candidates generated from a number of different camera poses (Figure 7). Such results are collected in Table 4. Considering the overall $\Sigma_L \bar{S}_L,\ L \in \{0,1,2\}$ for each of the four tested poses, the performance for poses 2–4 is quite similar, while pose 1 proves to be the worst in the set by a substantial margin. Nevertheless, the algorithm still performs better in pose 1 in a few cases (highlighted in orange in Table 4). Interestingly, despite our results, demonstration videos for 6-DoF Graspnet show a low-elevation camera pose (similar to pose 1) being used in real-world testing. It is worth noting that similar $\bar{S}^L$ scores do not necessarily guarantee similar S_k^L scores for all the objects in the layouts (highlighted in green). These results confirm the intuitive insight that camera point of view affects the performance of vision-based grasp planners.

CONCLUSIONS

In this work, we presented an easily deployable framework aimed at benchmarking grasp planning algorithms in a reproducible way on a real robotic setup, using the GRASPA benchmark. We showed how the framework contributes to addressing common difficulties when benchmarking grasp planners by reducing the effort required for the integration of additional algorithms in the environment, and to providing a fully reproducible software and hardware platform. We proved the usability of the proposed framework by benchmarking a selection of state-of-the-art vision-based grasp planning approaches (Dex-Net, GPD, and 6-DoF Graspnet), and by investigating the effect of a varying camera pose on 6-DoF Graspnet. Our results demonstrate that GPD is the best performer according to the GRASPA indices. We also provide experimental evidence that demonstrates that the choice of camera pose affects performance in a nonnegligible way. It is also important to point out that these results were obtained by using the original implementations of the methods to be as faithful as possible to the published results. Parameters were kept as out of the box as much as possible, and every necessary change was made to the best of our understanding, given the available documentation.

In an attempt to target a large portion of the community, the robot setup used in this article is compliant with the GRASPA benchmark, and it is based on hardware that is currently widespread in the field of robotic manipulation. All the code used for this article is open source and has been packaged in a way that is straightforward to deploy and used to reproduce the experiments. The experimental setup and methodology in this work are meant to be a pilot example for the research community, and they can be replicated and followed by others to benchmark other planning algorithms. Future work involves further development of the framework in terms of supported platforms. A more advanced contact modeling and grasp simulation can improve quality metrics that are currently a part of GRASPA. Finally, GRASPA itself could be used as a starting point to create a range of benchmarking scenarios and protocols for specific applications and trending topics in research (e.g., garbage sorting, bin picking, household tasks, and nuclear decommissioning).

ACKNOWLEDGMENT

We thank Elena Rampone for supporting the initial implementation of the framework. This work was supported by European Union Horizon 2020 Program for Research and Innovation project 730994 (The European Robotics Research Infrastructures Network) and ERA-NET CHIST-ERA call 2017 project Human-Guided Learning and Benchmarking of Robotic Heap Sorting. This article has supplementary downloadable material available at https://doi.org/ 10.1109/MRA.2023.3256272.

AUTHORS

Fabrizio Bottarel, Istituto Italiano di Tecnologia, 16163 Genova, Italy. Email: fabrizio.bottarel@iit.it.

Alessandro Altobelli, Istituto Italiano di Tecnologia, 16163 Genova, Italy. Email: alessandro.altobelli@iit.it.

Ugo Pattacini, Istituto Italiano di Tecnologia, 16163 Genova, Italy. Email: ugo.pattacini@iit.it.

Lorenzo Natale, Istituto Italiano di Tecnologia, 16163 Genova, Italy. Email: lorenzo.natale@iit.it.

REFERENCES

[1] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proc. Int. Conf. Adv. Robot. (ICAR)*, Jul. 2015, pp. 510–517, doi: 10.1109/ICAR.2015.7251504.

[2] N. Correll et al., "Analysis and observations from the first amazon picking challenge," *IEEE Trans. Autom. Sci. Eng. (from July 2004)*, vol. 15, no. 1, pp. 172–188, Jan. 2018, doi: 10.1109/TASE.2016.2600527.

[3] J. Stuckler, D. Holz, and S. Behnke, "RoboCup@Home: Demonstrating everyday manipulation skills in RoboCup@Home," *IEEE Robot. Autom. Mag.*, vol. 19, no. 2, pp. 34–42, Jun. 2012, doi: 10.1109/MRA.2012.2191993.

[4] Z. Liu et al., "Ocrtoc: A cloud-based competition and benchmark for robotic grasping and manipulation," *IEEE Robot. Autom. Lett.*, vol. 7, no. 1, pp. 486–493, Jan. 2022, doi: 10.1109/LRA.2021.3129136.

[5] Y. Bekiroglu et al., "Benchmarking protocol for grasp planning algorithms," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 315–322, Apr. 2020, doi: 10.1109/LRA.2019.2956411.

[6] J. Falco et al., "Benchmarking protocols for evaluating grasp strength, grasp cycle time, finger strength, and finger repeatability of robot end-effectors," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 644–651, Apr. 2020, doi: 10.1109/LRA.2020.2964164.

[7] B. Yang, J. Zhang, V. Pong, S. Levine, and D. Jayaraman, "Replab: A reproducible low-cost arm benchmark platform for robotic learning," 2019, *arXiv:1905.07447.*

[8] F. Bottarel, G. Vezzani, U. Pattacini, and L. Natale, "Graspa 1.0: Graspa is a robot arm grasping performance benchmark," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 836–843, Apr. 2020, doi: 10.1109/LRA.2020.2965865.

[9] M. Rudorfer, M. Suchi, M. Sridharan, M. Vincze, and A. Leonardis, "Burg-toolkit: Robot grasping experiments in simulation and the real world," 2022, *arXiv:2205.14099.*

[10] J. Mahler et al., "Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proc. Robot., Sci. Syst. Found.*, Jul. 2017, pp. 1–10, doi: 10.15607/RSS.2017.XIII.058.

[11] A. T. Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," 2017, *arXiv:1706.09911.*

[12] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *Proc. IEEE/CVF Int. Conf. Comput. Vision (ICCV)*, 2019, pp. 2901–2910.

[13] N. Vahrenkamp et al., "Simox: A robotics toolbox for simulation, motion and grasp planning," in *Intelligent Autonomous Systems 12*, vol. 193, S. Lee, H. Cho, K.-J. Yoon, and J. Lee, Eds. Berlin, Germany: Springer-Verlag, 2013, pp. 585–594.

[14] C. Ferrari and J. F. Canny, "Planning optimal grasps," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 1992, vol. 3, p. 6.

[15] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robot. Rep.*, vol. 1, no. 4, pp. 239–249, Dec. 2020, doi: 10.1007/s43154-020-00021-6.

[16] R. Newbury et al., "Deep learning approaches to grasp synthesis: A review," 2022, *arXiV:2207.02556.*

Open Access funding provided by 'Istituto Italiano di Tecnologia' within the CRUI CARE Agreement