Enabling Grasp Synthesis Approaches to Task-Oriented Grasping Considering the *End-State Comfort* and *Confidence* Effects

Emilio Maranci, Salvatore D'Avella, *Member, IEEE*, Paolo Tripicchio, and Carlo Alberto Avizzano, *Senior Member, IEEE*

***Abstract*—Choosing a good grasp is fundamental for accomplishing robotic grasping and manipulation tasks. Typically, the grasp synthesis is addressed separately from the planning phase, which can lead to failures during the execution of the task. In addition, most of the current grasping approaches privilege stability metrics, providing unsuitable grasps for executing subsequent tasks. The proposed work presents a framework for high-level reasoning to select the best-suited grasp depending on the task. The best grasp is chosen among a set of grasp candidates by solving an optimization problem, considering the environmental constraints, and guaranteeing the *end-state comfort* and the *confidence* effects for the task, similar to human behavior. The framework leverages *Generalized Bender Decomposition* to decouple the main non-linear optimization problem into sub-problems, thus presenting a modular structure. The method is validated with an experimental campaign using three different state-of-the-art grasping algorithms and three low-level motion planners in three different types of tasks: pick-and-place in a constrained environment, handover/tool-use, and object re-orientation. The experiments show that the proposed approach is able to find the best grasp, or at least one feasible, among the provided candidates for each task.**

***Index Terms*—Grasping, manipulation planning, task and motion planning.**

I. Introduction

SINCE the first years of life, human children learn by experience how to grasp objects of different shapes and in different scenarios. Thanks to that, for adult human beings, pick-and-place becomes a mechanical movement, and it is quite easy to understand how to grab an object never seen before thanks to their baggage of experience [1]. Psychologists have demonstrated that humans are driven primarily by safety and comfort considerations when moving in constrained environments [2]. The two principal metrics considered are the *end-state comfort* effect [3] and the clearance from obstacles during the motion [4], [5], which hereafter is referred to as *confidence* effect. The *end-state comfort* effect is a well-known phenomenon that involves choosing an initial configuration that is apparently unusual for grasping the object per se but maximizes the comfort of the final pose. For example, waiters grab inverted glasses to be filled with water holding each upside-down glass with an unusual thumb-down grasp rather than a more usual thump-up grasp to increase the comfort of the final posture. In addition, obstacles on the direct path toward the goal position influence human behavior during task execution [4]. Indeed, humans tend to maintain a safe distance from obstacles on their path, even if that means following longer routes to reach the destination (*confidence* effect) [4]. Recently, both hypotheses have been consolidated by demonstrating that humans usually choose overall comfort - manipulability and distance from obstacles - preferring longer but safer and more comfortable trajectories in the presence of obstacles [2]. In the recent literature, the problem of grasping objects is typically faced through a hierarchical approach that involves two distinct phases: the grasp synthesis and the task execution. The grasp synthesis often provides one or multiple grasp candidates ranked by some quality metrics [6], [7] or heuristics, either with classical [8] and data-driven techniques [9], [10], [11], that are sent afterward to the low-level planner to be executed. Since there is no flow of information between the high-level grasp strategy and the low-level planner, it could happen that the selected grasp is not suitable, and a different grasp should be tested, increasing the cycle time. Additionally, not all stable grasps are suitable for the subsequent manipulation tasks since they do not consider the comfort and the safeness of the task execution. Handing a screwdriver to a human collaborator implies grasping the metallic part of the

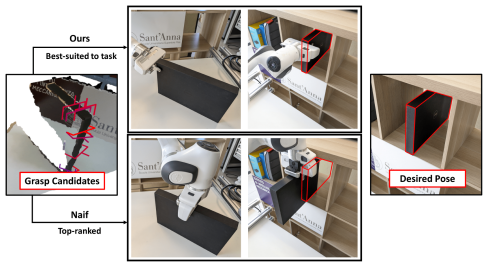

Fig. 1. Top-ranked grasps according to quality metrics are not always suited to task-oriented grasping tasks in constrained environments. The proposed approach still leveraging the same grasp synthesis method enables it to task-oriented setting with high-level reasoning.

Manuscript received 10 January 2024; accepted 16 April 2024. Date of publication 1 May 2024; date of current version 9 May 2024. This letter was recommended for publication by Associate Editor A. Sintov and Editor J. Borrás Sol upon evaluation of the reviewers' comments. *(Corresponding author: Salvatore D'Avella.)*

The authors are with the Department of Excellence in Robotics and AI, Mechanical Intelligence Institute, Scuola Superiore Sant'Anna, 56127 Pisa, Italy (e-mail: salvatore.davella@santannapisa.it).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3396102, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3396102

screwdriver close to the sharp tip, which is not a good location in terms of stability or for using the screwdriver as a tool to tighten a screw. In recent years, some novel approaches have emerged that provide suitable grasping for tool use or handover in human-robot [12], [13] collaboration scenarios. However, these methods can still be seen as grasp-synthesis techniques and are limited to a few tasks and scenarios.

The proposed work presents a framework that, starting from a set of grasp candidates provided by any off-the-shelf grasp synthesis approaches and given the desired pose of the item to be manipulated for the specific task, selects the best-suited grasp by solving an optimization problem taking into account the *end-state comfort* and *confidence* effects (fig. 1). The framework is general as it can be used with any grasping methods and low-level motion planners exploiting the novel shortcoming in the field and can also be applied to multiple tasks not limited to handover and tool use only having the desired pose of the target object, which is not uncommon in many industrial scenarios. The grasping synthesis method and the low-level motion planner can be chosen to satisfy the user's needs. In particular, the only requirement for the planner is to consider the distance between the robot and the obstacles in the scene in the computation, thus satisfying the *end-state comfort* and *confidence* effects. The proposed approach provides a general formulation and an example of implementation to embed these aspects into high-level reasoning. The method is validated with an experimental campaign using three different state-of-the-art grasping algorithms and three low-level motion planners in three different tasks: pick-and-place of objects in a constrained environment, object tool-use/handover, and object re-orientation. The experiments show that using the grasp with the highest score provided by the grasping methods is barely ever the best for accomplishing the tasks or is even feasible to be executed by the motion planner. In contrast, the proposed framework tends to find from the grasp candidate set the most suitable one for executing the task, provided that the low-level planner is able to compute a feasible path within the obstacles in the scene. The guarantees on the low-level motion planner depend on the chosen algorithm.

The following list summarizes the contributions of the proposed work:

- a high-level reasoning framework that can enable any grasping method to task-oriented grasping;
- considering the *end-state comfort* and *confidence* effects into the computation; and
- an optimal inverse kinematic solver for redundant robots that exploits null space control actions.

The remainder of the work is organized as follows: Section II introduces the works related to the field under discussion; Section III formally describes the proposed framework; Section IV presents the experimental setups, the validation procedures, and discusses the results; Section V concludes the letter.

II. Related Works

The problem of task-oriented grasping is relevant to the industry. The new concept of flexibility and human-robot collaboration introduced by Industry 4.0 implies that robots are not meant any more to repeat the same exact pre-programmed movement but have to demonstrate the ability to adapt to the changing and unpredictability of the environment. In many industrial tasks, robots have to grasp an object to place it in a specific pose [14], use it in a subsequent task [12], grasp the target to re-orient it [15], or even to help a human operator [13]. However, such solutions are restricted to that given task and are not versatile or easily adaptable to other situations, different from the proposed method.

Humans select grasps to satisfy three main constraints: hands, object, and task-based constraints [16]. Task requirements may need a less optimal grasp in terms of stability in favor of a higher ability to manipulate the object. The two principal metrics considered by humans are the *end-state comfort* effect [3] and the clearance from obstacles during the motion [4]. Many recent approaches typically perform grasp synthesis and planning separately [17]. The trend is to approach a target and execute the grasp without considering what should be done once the object is picked. However, by knowing the manipulation trajectory, it can be exploited to impose checks on the kinematic feasibility of grasps at the start and end poses and considerably reduce the grasp space. In addition, given a desired post-grasp trajectory of the object, different choices of grasp will often determine whether or not collision-free post-grasp motions of the arm can be found, which will deliver that trajectory.

Methods developed to use task-specific constraints and regions suitable for grasping an object for a given task learn these regions from simulation trials, large numbers of labeled images [18], or as abstract functions [19] that define task-specific structures. Recently, some approaches have focused on the tasks of handover and tool use, but they can still be considered grasp synthesis techniques [20] as they do not connect the high-level grasp with the low-level planner.

More recently, a few novel solutions specifically developed for general task-oriented grasping try to exploit scene comprehension. The most promising ones propose approaches that leverage deep neural networks and affordances theory [21], sometimes even exploiting LLMs (Large Language Models) to generalize the results to unknown tasks [22], to generate the best grasp candidate according to the required task. However, even if these recent approaches take into account the environment to avoid collisions during the grasping phase, none of them consider the path and final pose as constraints, and the task itself often ends with the grasping action without planning. Additionally, they cannot handle generic pick-and-place tasks and cannot manage objects that do not present functional parts and do not appear in the dataset the network has been trained on for that task.

Therefore, the idea is to enable some less recent grasping strategies [9], [10], [11] based on geometric considerations for pick-and-place tasks to the task-oriented setting. This work demonstrates that the best grasp according to traditional quality metrics used by most grasping strategies is not always feasible to accomplish the task and is also not optimal.

III. Proposed Approach

The proposed task-oriented grasping pipeline (Fig. 2) takes as input the set of grasp candidates from an off-the-shelf grasp

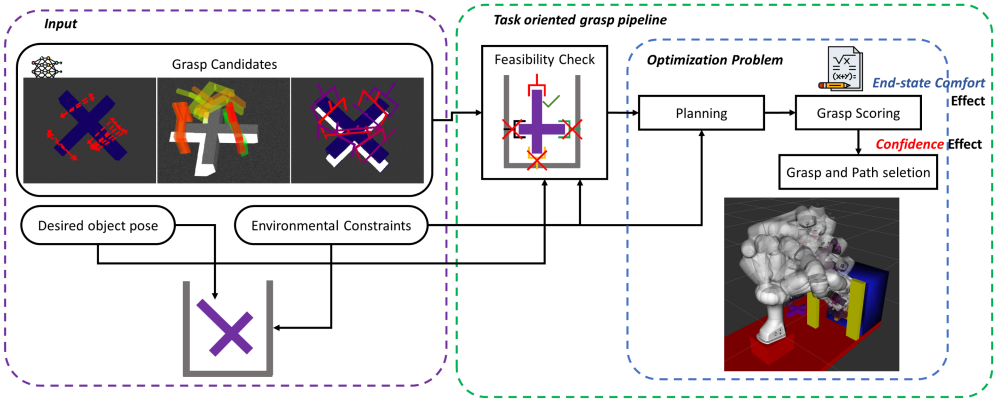

Fig. 2. Schematic overview of the proposed task-oriented grasp pipeline. The framework takes as input a set of grasp candidates provided by any grasp synthesis method, the task to be executed as the desired pose of the target object, and the environmental constraints for collision-free path planning. The set of grasp candidates is first evaluated to filter out unfeasible configurations that do not allow achieving the desired object pose by knowing the environmental constraints and the kinematics of the robot. For the grasps suitable for the task, a trajectory is computed, and those grasps for which a collision-free trajectory does not exist are discarded. The remaining grasps are scored using the optimization problem in (7), and the best grasp, along with a collision-free trajectory, is returned.

synthesis algorithm, the task to be executed as the desired pose of the target object, and the environmental constraints. Therefore, the pose of the object, the obstacles, and task constraints can be assumed to be known in the inertial frame without limiting the usability of the framework since the poses can be easily estimated by existing approaches (like [23]). Grasps are represented as a set of frames with translation and rotation expressed with respect to the inertial frame. The task has to be executed in a single movement thus, re-grasping and object re-orientation are not allowed. In addition, whether re-picking had been allowed, every grasp in the set could have been feasible as the object could have been re-oriented to achieve the desired pose, increasing execution time and control complexity. Therefore, once the object has been picked using a certain grasp, the transformation between the end effector frame and the object frame is fixed until the object is placed.

The proposed method assigns a cost to each grasp that indicates how suitable the grasp is to perform the task (*end-state comfort + confidence* effects), considering the object attached to the end-effector in the way it has been grasped. The costs are obtained by solving a set of Non-Linear Optimization Problems (NLOPs) in which the choice of the specific grasp represents the boundary conditions, each returning the optimal path $P^* = \{q_0^*, \dots, q_N^*\}$ to achieve the task, where q_i^* is the optimal arm configuration of the i-th waypoint of the path and N is arbitrary. Every NLOP in the set can be expressed as:

$$\begin{aligned} \min_{Q} \quad & C(Q) \\ \text{s.t.} \quad & FK(q_0) = EE_{pick} \\ & FK(q_N) = EE_{place} \end{aligned} \tag{1}$$

where $Q = \{q_0, \dots, q_N\}$ is the set of configurations to be optimized, $C(\cdot)$ is an arbitrary nonlinear convex cost function, $FK(\cdot)$ is the forward kinematic of the robotic arm, and $EE_{pick/place} \in SE(3)$ is the pick/place end-effector pose required for grasping, which is fixed.

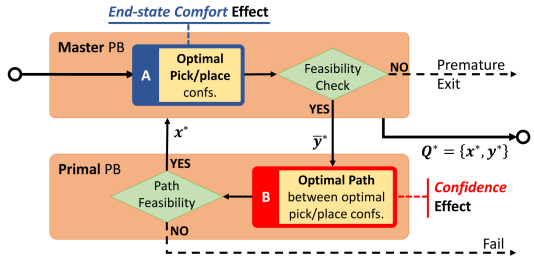

Fig. 3. Flowchart of the optimization process.

By splitting the set of optimization variables Q into two subsets $y = \{q_0, q_N\}, x = \{q_1, \dots, q_{N-1}\}$, and exploiting the *Generalized Benders Decomposition* [24], problem (1) can be written as two sub-problems:

$$\min_{y} \quad C_0(y) + v(y) \qquad [Master] \tag{2a}$$

$$v(\bar{y}) = \min_{x} \quad C_t(\bar{y}, x) \qquad [Primal]$$

$$\text{s.t.} \quad FK(\bar{y}) = [EE_{pick}, EE_{place}] \tag{2b}$$

where $\bar{y}$ is arbitrary and fixed in the *primal* sub-problem (2b), and the constraints are the same as problem (1), written in compact form. The functions C_0 and C_t are both nonlinear convex cost functions.

In the literature, this particular category of NLOPs is typically solved by an iterative procedure in which the two sub-problems are solved repeatedly until convergence is achieved: *primal* problem is solved first, starting with an initial guess of the fixed variables in the subset $\bar{y}$ and then, whether a feasible solution is obtained, the procedure switches to the *master* problem, and so on.

In contrast, this work follows a different strategy to solve the optimization problem. The approach is to evaluate the optimal pick and place configurations $\bar{y}^*$ that minimize the first term of the *Master* problem (2a). Subsequently, the *Primal* problem is solved by initializing it with the optimal value $\bar{y}^*$. If a solution for the *Primal* problem can be found, solving the *Master* problem is

straightforward, as the sub-problems are decoupled and similar convex functions are used during optimization. This way of proceeding helps to prematurely exit the optimization problem if no solution for the optimal pick and place configuration can be found, indicating that the robotic arm is unable to place the end-effector in the desired pick/place pose. The whole procedure with premature exits is displayed as a flowchart in Fig. 3.

Therefore, in this work, the decomposed problem (2) is interpreted as a one-shot two-step procedure that returns the optimal path to perform the required task using a specific grasp, which involves solving two decoupled optimization problems: first finding the best pick and place configuration (*end-state comfort* effect), and then finding the best path that connects these two configurations guaranteeing a safe distance to the obstacles in the scene (*confidence* effect). The following subsection first introduces the principal paradigm for the cost functions that the low-level planners have to embed for being applied within this framework, then gives more details on the sub-problems implementations.

A. Cost Function Formulation

An appropriate convex nonlinear cost function based on the distance between the robotic arm having the target attached to the end-effector and the obstacles in the scene needs to be employed in both the optimization sub-problems (see box A and B in Fig. 3) to respect the *end-state comfort* and the *confidence* effect. In general, the convex cost function can be expressed as:

$$C(q) = C(\, d(\mathbf{R}(q), \mathbf{O}),\, \eta(q)) \tag{3}$$

where $d(\mathbf{R}, \mathbf{O})$ is a generic term that takes into account the distance between the robot $\mathbf{R}$ and the set of obstacles $\mathbf{O}$ in the scene, and η is an optional term that could consider other aspects like the smoothness of the trajectory, joint velocities or any other, depending on the user needs. Indeed, many individuals account for secondary metrics ($\eta(q)$) with different extents in addition to the *end-state comfort* and *confidence* effect [4]. This also allows the proposed framework to be flexible to work with multiple optimizers and planners, returning the best path by considering the distance from obstacles and other relevant metrics.

We propose in the following an example of implementation for the cost function that has been designed to be as general as possible to meet the level of abstraction of a generic planning scene, which can be composed of multiple entities. The obstacles and the robot are modeled as volumetric shapes that can be composed of discrete parts: the robot can be composed of several links and some obstacles decomposed into simpler shapes. Given an arbitrary pair of elements in the scenario, named A and B, formed by n and m parts, respectively (Fig. 4), the matrix of distances between every part of A and B can be written as:

$$D(A,B) = \begin{bmatrix} d(a_1,b_1) & \dots & d(a_n,b_1) \\ \vdots & \ddots & \vdots \\ d(a_1,b_m) & \dots & d(a_n,b_m) \end{bmatrix} \in \mathbb{R}^{n\times m} \tag{4}$$

Minimizing (4) along the rows yields the column vector of minimum distances between each part of A and the element B,

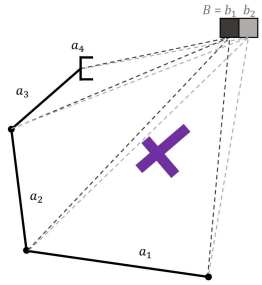

Fig. 4. Sketch example for the distance matrix computation. The dotted lines represent the distances between the robotic arm (composed of several links $a_1, \dots, a_5$) and the object B (formed by two parts b_1 and b_2, for generality).

shown in (5):

$$D_{\min}(A,B) = \begin{bmatrix} d_{\min}(a_1,B) \\ \vdots \\ d_{\min}(a_n,B) \end{bmatrix} \in \mathbb{R}^{n\times 1} \tag{5}$$

Finally, the cost component for the generic pair (A, B) can be expressed as:

$$C(A,B) = \sum_{i=1}^{n} \frac{1}{d_{\min}(a_i,B)^2 + \delta} \tag{6}$$

where the squared reciprocal of distances weighs less the large distances, not affecting much the total cost. The introduction of the machine precision δ avoids division by zero in case of contact and also preserves the convexity of the function. In addition, to prevent the optimal problem from converging on configurations in which one link of the robot goes much closer to an obstacle than all the others, a minimum safety distance is used to penalize such configurations. The radius of the safety spheres ρ can be set as a hyperparameter depending on the user requirement and has been put equal to 2.5 cm for the experimental campaign. The total cost associated with each robot configuration is obtained by summing multiple terms similar to (6), weighing the distance between the robotic arm and each obstacle in the scene:

$$\begin{aligned} C(q) &= \sum_{o\in\mathbf{O}} C(\mathbf{R}, o) \\ &= \begin{cases} \sum_{o\in\mathbf{O}} \sum_{l\in\mathbf{R}} \frac{1}{d_{\min}(l,o)^2+\delta}, & \text{if } d_{\min}(l,o) \geq \rho \\ \infty, & \text{otherwise} \end{cases} \end{aligned} \tag{7}$$

where $\mathbf{R}$ is the robot composed by a set of links l and $\mathbf{O}$ is the set of obstacles o.

B. Optimal Pick/Place Confs. Sub-Problem

The framework is meant for redundant manipulators thus, the optimal arm configuration for the pick and place pose (box A in Fig. 3) can be found by solving an optimization problem iteratively through the exploration of the null space of the end effector. Therefore, the optimization problem is tackled as shown in (8): it starts from a configuration obtained through inverse kinematics, constraining the end effector pose to be the one required for pick or place, and then the state vector is varied with control actions in the null-space of the positioning task of

the end effector, perturbing the robot configuration with a fixed amount at each iteration.

$$\begin{cases} q_0 = IK({}^{I}T_{EE,des}) \longrightarrow C(q_0) \\ q_1 = q_0 + \Delta q_0 \longrightarrow C(q_1) \\ \vdots \qquad\qquad\qquad\qquad \text{until } q_i \simeq q_0 \\ q_k = q_{k-1} + \Delta q_{k-1} \longrightarrow C(q_k) \\ \vdots \end{cases}$$

$$C(q^*) = \min(C(q_0), C(q_1), \dots, C(q_k), \dots) \tag{8}$$

where q is a configuration of the robot $\mathbf{R}$, q^* represents the configuration that minimizes the cost, and Δq is the discrete increment in the null space direction that is added to the configuration at each iteration of the process. The initial configuration is obtained by solving the inverse kinematic at the desired pose of the end effector. The procedure terminates when the configuration at a given iteration i is similar to the initial one, and the update step depends on the control action in the null space as follows:

$$\Delta q_{k-1} = (I - J^{+}_{k-1} J_{k-1}) v \tag{9}$$

where J^+ is the right pseudo-inverse of the jacobian evaluated at the corresponding step, $v \in \mathbb{R}^{r\times 1}$ is a gain vector that determines the amount of perturbation at each iteration of the process and r represents the number of joints of the manipulator. In this way, the optimization problem in (8) aims to find the best arm configuration among the solutions of the inverse kinematic of the end effector (*end-state comfort* effect).

C. Optimal Path b/w Optimal Pick/Place Confs. Sub-Problem

Once the optimal boundary configurations are generated, the *Primal* (planning) problem (box B in Fig. 3) can be solved using an optimal planner, either distance-based or embedded with the custom cost (7) to reflect the *confidence* effect. Therefore, any low-level motion planner implementing a cost function that can be expressed as (3) can be used. This modularity and flexibility of the framework derive from the main intuition of the proposed approach, which is connecting the high-level grasp synthesis method to the low-level motion planner through a distance-based cost. This connection enables including path and target position constraints into the grasp decision and represents the high-level reasoning process demanded by complex arbitrary tasks.

It is worth noticing that some considerations about the completeness and optimality properties of the overall framework, which is a combination of two algorithms, can be made. The completeness and the optimality of the null-space iterative search procedure are guaranteed since it is an optimal brute-force approach that can fully explore the space of solutions with the norm of step Δq going to 0. Therefore, the framework inherits the properties of the embedded low-level planner.

IV. Experimental Validation and Discussion

The proposed framework has been tested on three different types of tasks: two pick-and-place tasks in constrained static environments in which a target object has to be grasped and placed in a desired pose inside a container, avoiding obstacles along the path; a handover/tool-use task where the robot has to understand where to grasp a screwdriver or a hammer depending if the tool has to be used or passed to a human operator; and an object re-orientation task where the target needs to be picked and placed with a different orientation. The framework has been developed in ROS Noetic and tested first in simulation using Gazebo and then in the real world on the Franka Emika Panda robot equipped with a Realsense D435 camera.

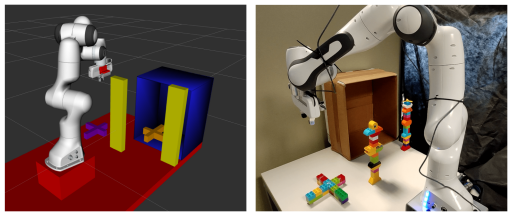

Fig. 5. Simulated scenario (left) resembling the real one (right). The object to be manipulated is the violet cross-shaped item that has to be moved inside the blue container in the orange pose, avoiding collision with the yellow obstacles and the container itself.

The procedure involves performing the required task using the best grasp selected by the proposed framework and a naif approach - just employing the top-scoring grasp outputted by the selected grasp synthesis methods, i.e., the works presented in [9], [10], [11] that are task-agnostic methods that generate grasp candidate without considering the subsequent task. The baseline algorithms are not adversarial methods but serve as proof of the effectiveness of the framework, showing that the proposed methodology can enable task-agnostic grasping algorithms into task-oriented ones where choosing a good grasp is mandatory for the execution of the task. The inverse kinematic of the manipulator is obtained using the Trac-IK algorithm [25]. After finding the optimal grasping and placing robot configuration with the problem formulated in (8), the planning is performed through three different low-level motion planners: RRTConnect* from OMPL [26], which is a sample-based planner, endowed with the proposed functional cost (7) to respect the requirement of the presented framework, CHOMP [27], and TrajOpt [28] which are two optimal planners that embed convex cost functions, and in addition, RRTConnect with CHOMP as post-processing adapter step. The experiments have been executed in a docker machine with Ubuntu 20.04 LTS running ROS-Noetic on a laptop intel-core i7 (11th gen) with 16 GB of RAM and NVIDIA MX450 graphic card with 2 GB of vRAM.

Finally, since the properties of the overall framework are inherited from the low-level planner, using optimal sample-based planners, like RRTConnect*, enables the pipeline to be probabilistically complete and optimal, ensuring that a feasible solution does not exist if the pipeline cannot find it in an unlimited amount of time. The same way of reasoning applies to CHOMP and TrajOpt.

A. Pick-and-Place in Constrained Environment

The scenario consists of a target object that has an asymmetric shape, which can be placed anywhere on the working table, two

TABLE I
RESULTS OF THE STATISTICAL ANALYSIS (SIMULATION) ON THE PICK-AND-PLACE TASK IN CONSTRAINED ENVIRONMENT

Bas.	Appr.	Planner	Succes rate	Pick/Place confs failure rate	Task exec. time [s]	Planning failure rate	Planning time [s]
GPD [9]	Ours	RRTConnect* (cost eq. (7))	100.00%	0%	92.16	0%	16.42
		CHOMP	50.00%	4.16%	62.85	45.83%	5.52
		TrajOpt	0.00%	0.00%	-	100.00%	-
		RRTConnect + CHOMP	87.50%	4.16%	90.45	8.33%	15.85
	Naif	RRTConnect*	12.50%	87.50%	16.24	0.00%	15.00
		CHOMP	4.16%	87.50%	6.48	4.16%	5.33
		TrajOpt	0.00%	100.00%	-	0.00%	-
		RRTConnect + CHOMP	8.32%	87.50%	17.39	0.00%	16.62
GQ-CNN [10]	Ours	RRTConnect* (cost eq. (7))	75.00%	25%	85.53	0%	16.27
		CHOMP	45.83%	20.83%	60.03	33%	5.63
		TrajOpt	0.00%	20.83%	-	79.16%	-
		RRTConnect + CHOMP	79.16%	20.83%	88.75	0.0%	15.74
	Naif	RRTConnect*	4.16%	95.83%	16.03	0%	15.05
		CHOMP	4.16%	95.83%	6.37	0%	5.49
		TrajOpt	0%	100%	-	0%	-
		RRTConnect + CHOMP	4.16%	95.83%	8.17	0%	7.02
Graspnet [11]	Ours	RRTConnect* (cost eq. (7))	95.83%	4.16%	100.10	0.00%	16.71
		CHOMP	70.83%	8.33%	67.49	20.83%	6.22
		TrajOpt	0.00%	8.33%	-	91.60%	-
		RRTConnect + CHOMP	95.83%	4.16%	103.07	0%	15.77
	Naif	RRTConnect* (cost eq. (7))	0.00%	100.00%	-	0.00%	-
		CHOMP	0.00%	100.00%	-	0.00%	-
		TrajOpt	0.00%	100.00%	-	0.00%	-
		RRTConnect + CHOMP	0.00%	100.00%	-	0.00%	-

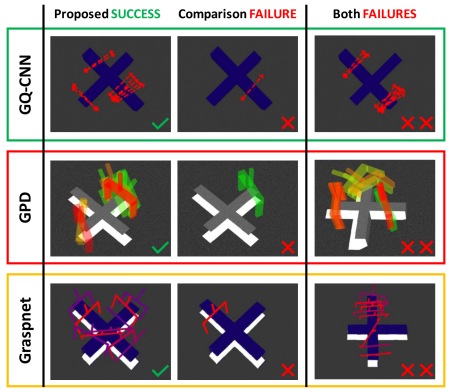

Fig. 6. Some examples of grasp candidates generated by the three baselines [9], [10], [11] during tests. The first two columns compare the whole set of grasp candidates and the best grasp, i.e., the one with the highest score obtained in a run, where the proposed procedure registers a success and the compared baseline failed. The third column presents the grasp sets obtained in cases both procedures have failed.

obstacles that hinder the trajectory from the picking configurations to the placing configuration, and a box container in which the target object has to be placed in a precise pose having wall constraints. Fig. 5 shows the scenario realized in Gazebo and in the real world. The use of the asymmetric object is meant to stress the fact that only some grasps are suitable to accomplish the task. In addition, the generalizability of the framework is not limited to objects presenting such a similar shape since the grasp set is provided by off-the-shelf grasp synthesis algorithms.

Several experiments have been done comparing the results obtained using the grasping configuration selected through the presented optimization problem in the case of the proposed framework and as the top-scoring grasp for the other three grasp synthesis methods with different low-level motion planning algorithms (RRTConnect*, CHOMP, and TrajOpt). During the experiments, the pose of the target object is varied at each run, keeping the same pose for each method to guarantee a fair comparison. In particular, the object has been placed into five different positions rotating by steps of 45 degrees in each position until a complete rotation is reached for a total of 40 trials per method. Each baseline has been configured to provide 10 grasp candidates, and each run consists of comparing the result of the best grasp candidate selected by the baseline and the best grasp chosen by the proposed framework using the set of grasps given by the baseline to vary the low-level motion planner.

It has been noticed that if the proposed method fails to complete the task it is due to the fact that the baseline does not provide any feasible grasp of the set of candidates or that the low-level motion planner does not find a feasible path (in case of TrajOpt, which is not able to plan in tight constrained environments). When this behavior occurs, i.e., if neither the proposed procedure nor the baseline can accomplish the task, the whole run is repeated, up to a maximum number of ten times equal. After the 10th attempt without any success, the next object pose is considered by registering a failure. Table I reports the results of the statistical analysis giving a comparison of the grasping success rate and then a breakdown of some detail of the procedure providing the failure rate ascribable to the absence of a feasible grasp in the set of the grasp candidate (*Pick/Place confs failure rate column*) and to the low-level planner (*Planning failure rate column*), and the time required to accomplish the whole task and computing the trajectory for connecting the pick and the place configurations. It is worth noticing that the task completion time is averaged on the number of experiments accounting for the total number of grasps in the grasp candidates set (N = 10). Instead, the planning time is averaged only on the experiments in which the planning can actually take place as there are valid pick-and-place configurations. The average time required by our framework to determine a pick or place configuration is 0.75 s, while [9], [11], and [10] require on average 4.36, 1.31, and 1.91 s, respectively. Such experiments demonstrated how the grasp with the highest score returned by the baselines is rarely the best grasp to use to execute the task in a constrained scenario, getting even no success for [11]. On the contrary, the proposed procedure manages to find a feasible grasp in most cases. Some examples of the grasp candidates generated by baseline algorithms during the tests are shown in Fig. 6.

B. Handhover/Tool-Use

The scenario consists of an object (a screwdriver or a hammer) placed on the working table without environmental constraints. The robot should understand where it is better to grasp the tool to use it or to give it to the human operator. Fig. 7 shows graphically the setup, where the green box region represents the human-robot collaboration sharing area. Exploiting the same three baselines used in the previous scenario, the proposed framework can determine the best grasp to discriminate between the tool use or the handover depending on the presence of the human operator in the scene represented as an obstacle for the pipeline. Ten runs have been executed with the three baselines, also varying the low-level motion planner, comparing each approach with the presented method mimicking the tool use or the handover. Table II summarizes the obtained results.

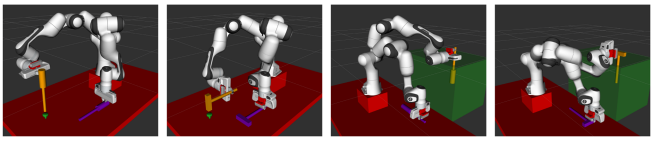

Fig. 7. First two: example of a tool use task with a screwdriver tightening a screw (left) and with a hummer humming a nail (right). Second two: example of handover in a human-robot collaboration scenario where the robot gives the tool (the screwdriver on the left and the hammer on the right) to the operator in the green area demanded to the interaction.

TABLE II
RESULTS OF THE STATISTICAL ANALYSIS (SIMULATION) ON THE TOOL-USE/HANDOVER WITH SCREWDRIVER AND HAMMER

Bas.	Appr.	Planner	Tool-use		Handover	
			Screwdriver	*Hammer*	*Screwdriver*	*Hammer*
GPD [9]	Ours	RRTConnect* (cost eq. (7))	100%	100%	80%	100%
		CHOMP	100%	100%	80%	100%
		TrajOpt	100%	90%	80%	100%
		RRTConnect + CHOMP	100%	100%	80%	100%
	Naif	RRTConnect*	90%	60%	0%	60%
		CHOMP	90%	60%	0%	60%
		TrajOpt	90%	50%	0%	60%
		RRTConnect + CHOMP	90%	60%	0%	60%
GQ-CNN [10]	Ours	RRTConnect* (cost eq. (7))	10%	100%	100%	70%
		CHOMP	10%	100%	100%	70%
		TrajOpt	10%	80%	100%	70%
		RRTConnect + CHOMP	10%	100%	100%	70%
	Naif	RRTConnect*	10%	70%	80%	30%
		CHOMP	10%	70%	80%	30%
		TrajOpt	10%	60%	80%	30%
		RRTConnect + CHOMP	10%	70%	80%	30%
Graspnet [11]	Ours	RRTConnect* (cost eq. (7))	100%	100%	60%	100%
		CHOMP	100%	100%	60%	100%
		TrajOpt	100%	100%	60%	100%
		RRTConnect + CHOMP	100%	100%	60%	100%
	Naif	RRTConnect* (cost eq. (7))	100%	10%	0%	70%
		CHOMP	100%	10%	0%	70%
		TrajOpt	100%	10%	0%	70%
		RRTConnect + CHOMP	100%	10%	0%	70%

In this case, a breakdown is not provided since the choice of the low-level motion planners does not affect much the performance of the overall task since the environment is not as complex as the previous one, and the grasp selection and planning times reflect the outcomes presented in the previous experiments.

Depending on the geometry of the object, the three baselines work well for one of the two tasks. For example, GQ-CNN [10] can quite successfully perform the tool-use with the hammer but not with the screwdriver at all, and, on the contrary, perform quite well in the handover with the screwdriver and poorly with the hammer. The same reasoning can be applied to the other two baseline methods because, in general, neural network-based approaches are biased by the dataset on which they have been trained. In addition, the grasp candidates are mostly concentrated in some regions of the object and not distributed on all its surfaces, limiting in some circumstances the proposed approach, like in the case of tool-use with a screwdriver using GPD [9]. However, the proposed solution can accomplish both tasks remarkably, highlighting the versatility of the method. For the sake of simplicity, the action of tightening a screw or hammering a nail has not to be performed since it is out of the scope of the presented work and would have required implementing a sequence of actions to complete the task that are only technical. However, the results consider if such an action could have been executed with the given grasp, no matter the required time.

C. Real World Experiments

1) Pick-and-Place Tasks: The same pick-and-place experiments run in simulation have been performed on the real robot in the world, using a setup similar to the virtual one. We selected RRTConnect* endowed with the proposed functional cost as the low-level motion planner since it provided the best performance in simulation on average.

In addition, a more daily life task, i.e., picking a book and placing it on a shelf has been performed using a box in place of the book. It is worth noticing that the box has been selected on purpose such that only grasps on one side were suitable to complete the task of inserting the box into the shelf. If the naif approaches selected the proper side randomly (depending on their scoring criteria), the proposed method could always choose the correct one.

2) Object Re-Orientation: To further highlight the flexibility of the proposed method, an object re-orientation task in which a mug has to be stored in a cupboard has been executed. The same shelf has been used in place of the cupboard for practical reasons. Similarly to the other cases, the naif approaches barely ever choose a correct grasping configuration that allows storing the mug on the shelf in the desired pose, in contrast to our approach.

The results obtained during the real experiments reflect the outcome of the statistical analysis. Figs. 1 and 8 show a few runs.

V. CONCLUSION

Grasping is a long-studied field in robotics. Many works have focused on how to grasp an object to maximize stability. However, such grasps may not be suitable to accomplish the subsequent task after the grasping, and the high-level grasping algorithm often does not communicate with the low-level path planner that takes into account the environmental constraints of the scene. Humans have this high-level reasoning ability and often prioritize *end-state comfort* and *confidence* effects over stability during manipulation tasks. Based on this insight, the proposed work presents a general framework for task-oriented grasp planning based on an optimization problem. The framework is general and can be used in different scenarios as it is agnostic to the identity of the objects exploiting the grasping candidates set provided by off-the-shelf grasp synthesis methods. The purpose of the framework is not to generate grasp candidates but to connect the high-level grasping with the low-level planner through a grasp score, taking into account the environmental constraints and the task to be executed. Compared to recent approaches, the presented framework is not limited to tool use or handover tasks but can interface with other tasks.

The framework has been validated in three different tasks and compared with naif baselines grasp algorithms using diverse low-level motion planners. The experimental campaign highlighted the inadequacy of traditional grasping algorithms that do not take into account the subsequent task to be accomplished in highly constrained environments.

The limitation of the proposed method is that the success of the task is constrained to the existence of at least a feasible grasp in the set of candidates provided by the grasp synthesis method

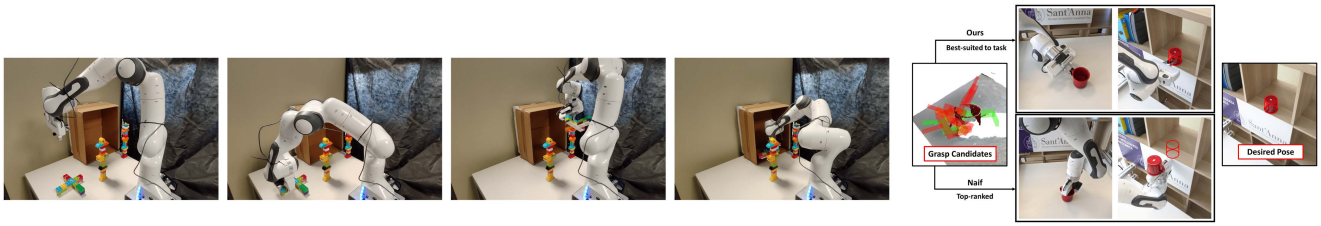

Fig. 8. Example of a run in the real world. Left: pick the crossed-shaped object and place it inside the box. Right: store a mug in a cupboard.

and to the ability of the low-level planner to find a collision-free trajectory, which is not trivial in tightly constrained environments. Future work will involve including more proper grasp synthesis algorithms and improving the planners' performance, as well as reducing the planning time. However, the pipeline is modular, and the new shortcomings in the two fields can be applied.

REFERENCES

[1] J. Konczak, M. Borutta, H. Topka, and J. Dichgans, "The development of goal-directed reaching in infants: Hand trajectory formation and joint torque control," *Exp. Brain Res.*, vol. 106, no. 1, pp. 156–168, 1995.

[2] F. Iwane, A. Billard, and J. d. R. Millán, "Inferring individual evaluation criteria for reaching trajectories with obstacle avoidance from EEG signals," *Sci. Rep.*, vol. 13, no. 1, 2023, Art. no. 20163.

[3] C. J. Coelho, B. E. Studenka, and D. A. Rosenbaum, "End-state comfort trumps handedness in object manipulation," *J. Exp. Psychol.: Hum. Percep. Perform.*, vol. 40, no. 2, 2014, Art. no. 718.

[4] I. T. Garzorz, A. G. Knorr, R. Gilster, and H. Deubel, "The influence of obstacles on grasp planning," *Exp. Brain Res.*, vol. 236, pp. 2639–2648, 2018.

[5] M. Mon-Williams, J. R. Tresilian, V. L. Coppard, and R. G. Carson, "The effect of obstacle position on reach-to-grasp movements," *Exp. Brain Res.*, vol. 137, pp. 497–501, 2001.

[6] C. Ferrari and J. F. Canny, "Planning optimal grasps," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1992, vol. 3, pp. 2290–2295.

[7] Z. Li and S. S. Sastry, "Task-oriented optimal grasping by multifingered robot hands," *IEEE J. Robot. Autom.*, vol. 4, no. 1, pp. 32–44, Feb. 1988.

[8] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—A survey," *IEEE Trans. Robot.*, vol. 30, no. 2, pp. 289–309, Apr. 2014, doi: 10.1109/TRO.2013.2289018.

[9] A. Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *Int. J. Robot. Res.*, vol. 36, no. 13/14, pp. 1455–1473, 2017.

[10] J. Mahler et al., "Learning ambidextrous robot grasping policies," *Sci. Robot.*, vol. 4, no. 26, 2019, Art. no. eaau4984.

[11] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1billion: A large-scale benchmark for general object grasping," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11444–11453.

[12] M. Sun and Y. Gao, "GATER: Learning grasp-action-target embeddings and relations for task-specific grasping," *IEEE Robot. Automat. Lett.*, vol. 7, no. 1, pp. 618–625, Jan. 2022.

[13] V. Ortenzi, A. Cosgun, T. Pardi, W. P. Chan, E. Croft, and D. Kulić, "Object handovers: A review for robotics," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1855–1873, Dec. 2021.

[14] S. D'Avella, C. A. Avizzano, and P. Tripicchio, "RoS-industrial based robotic cell for Industry 4.0: Eye-in-hand stereo camera and visual servoing for flexible, fast, and accurate picking and hooking in the production line," *Robot. Comput.- Integr. Manuf.*, vol. 80, 2023, Art. no. 102453.

[15] K. Wada, S. James, and A. J. Davison, "Reorientbot: Learning object reorientation for specific-posed placement," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 8252–8258.

[16] M. R. Cutkosky, "On grasp choice, grasp models, and the design of hands for manufacturing tasks," *IEEE Trans. Robot. Automat.*, vol. 5, no. 3, pp. 269–279, Jun. 1989.

[17] S. D'Avella, A. M. Sundaram, W. Friedl, P. Tripicchio, and M. A. Roa, "Multimodal grasp planner for hybrid grippers in cluttered scenes," *IEEE Robot. Automat. Lett.*, vol. 8, no. 4, pp. 2030–2037, Apr. 2023.

[18] J. Ortiz-Haro, V. N. Hartmann, O. S. Oguz, and M. Toussaint, "Learning efficient constraint graph sampling for robotic sequential manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 4606–4612.

[19] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kPAM: Keypoint affordances for category-level robotic manipulation," in *Proc. Int. Symp. Robot. Res.*, 2019, pp. 132–157.

[20] J. Jian, X. Liu, M. Li, R. Hu, and J. Liu, "Affordpose: A large-scale dataset of hand-object interactions with affordance-driven hand pose," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 14713–14724.

[21] A. Murali, W. Liu, K. Marino, S. Chernova, and A. Gupta, "Same object, different grasps: Data and semantic knowledge for task-oriented grasping," in *Proc. Conf. Robot Learn.*, 2021, pp. 1540–1557.

[22] C. Tang, D. Huang, W. Ge, W. Liu, and H. Zhang, "GraspGPT: Leveraging semantic knowledge from a large language model for task-oriented grasping," *IEEE Robot. Automat. Lett.*, vol. 8, no. 11, pp. 7551–7558, Nov. 2023.

[23] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16611–16621.

[24] I. Nowak, *Relaxation and Decomposition Methods for Mixed Integer Nonlinear Programming*, vol. 152. Berlin, Germany: Springer, 2005.

[25] P. Beeson and B. Ames, "Trac-ik: An open-source library for improved solving of generic inverse kinematics," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, 2015, pp. 928–935.

[26] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. ICRA. Millennium Conf.. IEEE Int. Conf. Robot. Automat. Symp.*, 2000, vol. 2, pp. 995–1001.

[27] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2009, pp. 489–494.

[28] J. Schulman et al., "Motion planning with sequential convex optimization and convex collision checking," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1251–1270, 2014.

Open Access funding provided by 'Scuola Superiore "S. Anna" di Studi Universitari e di Perfezionamento' within the CRUI CARE Agreement