Integrated Grasp and Motion Planning using Independent Contact Regions

Joan Fontanals[1], Bao-Anh Dang-Vu[2], Oliver Porges[2], Jan Rosell[1], and Máximo A. Roa[2]

***Abstract*— Traditionally, grasp and arm motion planning are considered as separate tasks. This paper presents an integrated approach that only requires the initial configuration of the robotic arm and the pose of the target object to simultaneously plan a good hand pose and arm trajectory to grasp the object. The planner exploits the concept of independent contact regions to look for the best possible grasp. The goal poses for the end effector are obtained using two different methods: one that biases a sampling approach towards favorable regions using principal component analysis, and another one that considers the capabilities of the robotic arm to decide the most promising hand poses. The proposed method is evaluated using different scenarios for the humanoid robot SpaceJustin.**

I. INTRODUCTION

Grasp planning and arm motion planning have been traditionally considered as two separated stages, solved in a sequential manner to find a valid trajectory that allows a robot to pick up an object from the environment. Depending on the way that the grasp configurations are searched, grasp planning approaches can be separated into analytical and data-driven methods [1]. Typically, analytical methods tackle the grasp search with heuristic- or optimization-based algorithms, adapting the restrictions and goal functions to the particularities of the problem. On the other hand, data-driven methods mainly rely on an off-line generated set of grasp configurations, which is later used to choose a valid grasp pose according to other restrictions of the environment. To measure the goodness of a grasp, different grasp quality measures have been proposed [2]; the most common one is the measure of the maximum perturbation wrench that a grasp can resist in any direction [3].

The problem of collision-free arm motion planning, i.e. finding a trajectory from an initial to a final configuration while avoiding obstacles in the environment, has been tackled mainly using two approaches: sampling- and optimization-based planning. Because of their simplicity, sampling-based motion planners have become very popular. They include methods based on probabilistic road-maps (PRM), where several collision-free configurations are computed beforehand and stored in the roadmap, which is later queried for the desired path; the approach is particularly useful for multiple queries in very structured and static scenarios. For single queries, rapidly-random exploring trees (RRTs) sample the configuration space while creating a tree that connects the start and goal configuration [4], which proves useful for changing environments like those encountered in manipulation tasks with movable obstacles. These randomized path planners are probabilistically complete, but often return non-optimal solutions that require a post-processing step to smooth and shorten the computed trajectories. On the other hand, optimization-based planners define path computation as an optimization problem that minimizes a suitable cost function [5]. Such cost function can be used to avoid obstacles, provide smoothness, and respect kinematic and dynamic constraints of each problem. These planning approaches suffer from limitations associated to optimization algorithms such as non-convergence or local minima, and lack the exploration nature of sampling-based planners required for difficult path planning problems. The consideration of optimality within sampling-based planners has been recently proposed [6], although these approaches are not computationally efficient for high-dimensional problems.

Finding a feasible grasp for a given scenario has been traditionally solved in a sequential manner. Given the object model, a grasp database is computed offline, and grasps are sorted according to some grasp quality measure. Later, given the current scenario, the feasibility of the grasps is evaluated, i.e. only grasps that have a corresponding inverse kinematics (IK) solution for the arm are considered in later stages. One feasible grasp defines one goal configuration for the robot. Then, given the initial and final arm configuration, a collision-free path for the arm is searched using some path planning method. If no path is found a new grasp is chosen,

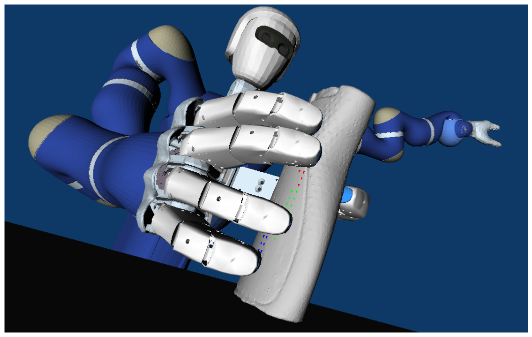
Fig. 1. SpaceJustin grasping an object. The object shows the independent contact regions for the fingers (colored points).

*This work was partially supported by the project SMErobotics, from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 287787, and by the Spanish Government through the projects DPI2011-22471 and DPI2013-40882-P.

[1]J. Fontanals and J. Rosell are with the Institute of Industrial and Control Engineering, Universitat Politècnica de Catalunya (UPC). `Joan.Fontanals@estudiant.upc.edu`, `Jan.Rosell@upc.edu`

[2]B.A. Dang-Vu, O. Porges and M.A. Roa are with the Institute of Robotics and Mechatronics at the German Aerospace Center (DLR). `Bao-Anh.Dang-Vu@dlr.de`, `Oliver.Porges@dlr.de`, `Maximo.Roa@dlr.de`

until a path is obtained or until the complete database has been explored and no solution is found [7].

Working with a discrete set of grasps from a database has several disadvantages, such as limitations in the number of grasp possibilities for the object and low adaptability to the environment, i.e. no new grasps can be explored even if they mean only a slight change of pose with respect to one predefined grasp. An integrated approach has been recently proposed that, based only on the initial arm position and the pose of the object to be grasped, finds a suitable grasp and a path -using a bidirectional RRT- to effectively grasp the object [8]. This approach does not restrict the search of a grasp to a predefined set of hand configurations, but explores the most convenient way to grasp the object according to the actual situation. In this way, there is no need to precompute in an off-line phase a set of candidate grasps, constraints to the grasp planning problem can be added on-line, and the search of valid grasps is limited to reachable poses of the arm obtained via forward kinematics.

Inspired by this integrated planner, this paper proposes a new approach that looks for a feasible grasp on the object exploiting the concept of independent contact regions (ICRs). These regions are defined such that if each finger is positioned on its corresponding contact region, a force-closure (FC) grasp is always obtained, independently of the exact location of each finger [9]. ICRs have been successfully used as part of a shared autonomy approach for telemanipulation, where the human decides the hand pose with respect to the object based on the ICR information [10], [11]. Our work uses the ICRs as part of a fully autonomous motion planner. In order to replace the human input, and to improve the efficiency of the planning process, the search of a new grasp is restricted to areas of potentially good configurations using two tools: 1) A principal component analysis (PCA) that biases a sampling-based search of hand poses towards promising regions, and 2) an efficient representation of the reachability of the robot in a capability map [12], which restricts the search to feasible manipulation areas for the robot. A great exploration capacity is required to simultaneously combine grasp and motion planning, therefore bidirectional RRTs are used for the planning strategy.

After this introduction the paper is structured as follows. Section II reviews related work on integrated approaches for grasp and motion planning. Section III explains the concepts and methods used for searching valid grasps, including ICRs and the two tools above-mentioned to bias the search towards promising regions for grasping. Section IV describes the algorithm that integrates the grasp and motion planner. In Section V the algorithm is tested in different environments, its performance is compared against other integrated planner, and a discussion on the method concludes the paper.

II. Related work on integrated approaches

While traditional motion planning for grasping connects an initial to a desired arm/hand configuration, an integrated approach simultaneously looks for the hand configuration and arm motion to grasp a particular object given the initial arm/hand configuration and object pose. In most of the cases it is not important how the object is grasped, as long as some minimum quality conditions are met.

An initial step towards an integrated approach was taken by defining online a set of candidate grasp poses, obtained by solving an optimization problem that takes into account the location of obstacles in the scene and the likelihood of not being in collision and leading to FC grasps [13]. Candidate grasps obtained in this way are then tried out by moving the wrist and closing the fingers until some grasp is detected. This idea evolved towards the definition of Task Space Regions [14], [15], which manually define areas (subsets of $SE(3)$) of predefined good grasp poses around the object for an underactuated hand. The regions are later used to plan arm paths with a bidirectional RRT.

The previous work closest to our approach is [8], where the grasp search is performed while the motion planning loop is running. From the initial arm configuration, an RRT is built and starts growing. Occasionally, an approach movement is tried from some existing node in the tree towards the object, stopping as close as possible to the object while still there are no collisions detected. Once the hand is at this position, a grasp is evaluated by simply closing the fingers around the object. During the whole planning process the approach trajectories towards the object are tracked, to try to uniformly cover different approach directions.

Our current work also proposes a way to integrate the grasp and motion planning procedures without using any precomputed grasp database, thus making it suitable for arbitrary objects and complex end effectors. We use the idea of a goal region around the object that contains promising hand poses, but without explicitly constraining the potential grasps on the object. The method also provides robustness against uncertainty in the positioning of the fingers by computing reachable independent contact regions, which also leads to more candidate grasps than the closing and testing policy adopted in some common grasp planning approaches. To be able to perform better in cluttered environments, bidirectional RRTs are used to connect the initial configuration with the multiple grasps discovered during the search process. The planning algorithm tries to bias the connection of the trees to the ones leading to better grasps, measured according to a suitable quality metric. Details of the method are provided in the next two sections.

III. Grasp planning

The integrated grasp and motion planning algorithm presented in this paper uses the concept of Independent Contact Regions; the size of the ICRs is used as a quality metric for the planning process. Two different approaches are used to effectively search for grasp poses. The first one makes an adaptive search of grasp configurations using a Principal Component Analysis on previously successful samples. The second approach uses the information of the robot's capability map to test candidate grasp poses in the workspace.

A. Reachable Independent Contact Regions (ICRs)

ICRs correspond to regions on the boundary of the object (here represented by sets of points), such that if each finger i is positioned on its corresponding ICR_i an FC grasp is always obtained. Not only the FC condition is guaranteed, but also a minimum desired grasp quality is met. The grasp quality is quantified with the largest perturbation wrench that the grasp can resist in any perturbation direction [3].

The algorithm computing ICRs has two phases [16]. The first one obtains the points on the object surface reachable by the fingers for a given hand pose; points are represented by a position vector and a corresponding normal direction. Precomputation of the workspaces for each finger speeds up this process. The second phase looks for points inside the reachable regions that guarantee an FC grasp with the desired minimum quality.

For the integrated planner, an additional quality metric for the ICRs is required. As the regions are represented as discrete sets of points, the more points in the region the more possibilities of grasping the object are provided. The metric is then defined as the number of different grasp possibilities, which is indirectly related to the area that the regions cover on the object surface. Let n be the number of fingers with contact regions on the object surface, and m_i be the number of discrete points in each region ICR_i; the quality metric for the contact regions for a particular hand pose is given by

$$Q_{ICR} = \prod_{i=1}^{n} m_i \qquad (1)$$

B. Method 1: Search of grasp poses using PCA

During the execution of the integrated planning algorithm, one requirement is to find configurations of the hand that likely lead to a valid grasp. To achieve this purpose, a goal region around the object is defined using a superellipsoid. Samples of potential hand positions are taken inside this space, and an adaptive sampling method based on PCA is applied to bias the sampling towards promising areas. This follows the idea proposed in [17], where the PCA guides the sampling process in path planning problems with narrow passages by adapting the region in the configuration space from where the next sample is taken.

The general equation of the superellipsoid is given by

$$r(\eta,\omega) = \begin{bmatrix} a_1 \cos^{\epsilon_1}\eta \cos^{\epsilon_2}\omega \\ a_2 \cos^{\epsilon_1}\eta \sin^{\epsilon_2}\omega \\ a_3 \sin^{\epsilon_1}\eta \end{bmatrix}; \begin{array}{c} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega \leq \pi \end{array} \qquad (2)$$

where a_1, a_2, a_3 are parameters that define distances along the three coordinate axis, which correspond to the size of the object plus the length of the robotic hand. The shape parameters ϵ_1 and ϵ_2 must be selected such that the goal region represents the shape of the real object while avoiding the concentration of samples in undesired places (Fig. 2).

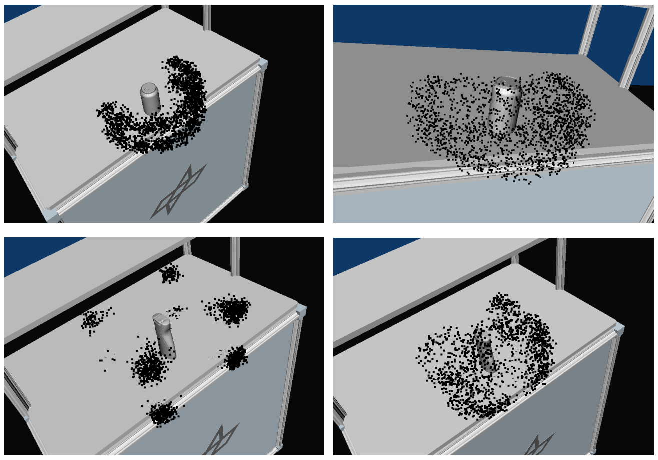

Fig. 2. Influence of the superellipsoid parameters on the distribution of samples inside the goal region for a soda can in the top row (with $\epsilon_1 = 0.1$, $\epsilon_2 = 1.0$ to the left and $\epsilon_1 = 0.5$, $\epsilon_2 = 1.0$ to the right), and for a shampoo bottle in the bottom row (with $\epsilon_1 = 0.1$, $\epsilon_2 = 0.1$ to the left and $\epsilon_1 = 0.5$, $\epsilon_2 = 0.5$ to the right).

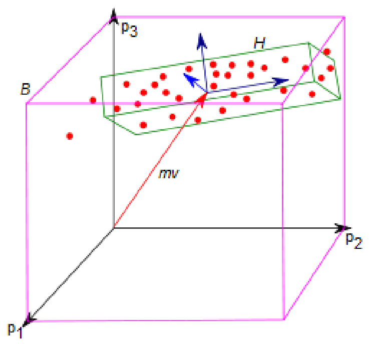

Fig. 3. PCA-based sampling applied to a space B defined by 3 parameters; mv is the mean of the set of samples and H is the new sampling region.

The goal region is then defined by the set A

$$A = \left\{ (a_1, a_2, a_3, \eta, \omega)^T \middle| \begin{array}{c} a_{i_{min}} \leq a_i \leq a_{i_{max}} \\ -\pi/2 \leq \eta \leq \pi/2 \\ \omega_{min} \leq \omega \leq \omega_{max} \end{array} \right\} \qquad (3)$$

where $a_{i_{min}}$ and $a_{i_{max}}$, $i = \{1, 2, 3\}$, define the region of interest along each coordinate axis (for instance, to avoid points below or too close to the supporting surface). ω_{min} and ω_{max} choose the angular portion of the superellipsoid that faces the robot, as shown in Fig. 2, which discards grasp poses that might look unnatural. In this paper the parameters for A were manually set, although an automatic selection process could also be implemented.

Each sample inside A provides a Cartesian position of the hand with respect to the object coordinate frame. The orientation of the hand with respect to the object is defined in such a way that the object is always "visible" for the palm, as suggested in [18] to generate more human-like paths. This still leaves one DoF, the roll orientation for the hand around the approach axis, which is chosen such that the palm is parallel to the main axis of inertia of the object [19]. Thus, the complete hand pose with respect to the object T_{hand}^{object} is obtained, which is later used for the computation of the ICRs.

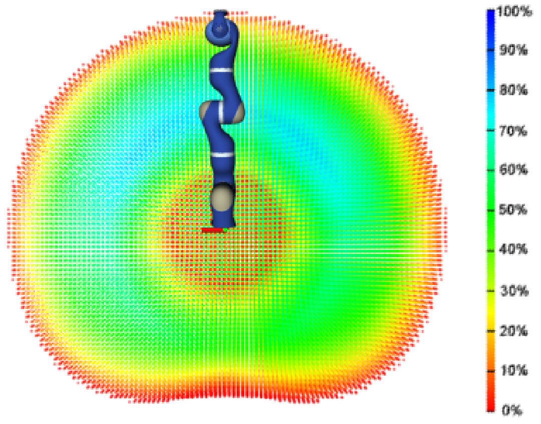

Fig. 4. Capability map for one arm of SpaceJustin. Colors indicate the reachability index for each voxel, i.e. the percentage of discrete poses within that voxel that are reachable for the arm.

During the complete planning procedure that will be presented in Section IV, different samples for robot tool frame (TCP) locations are taken inside A. These samples are checked to see whether they correspond to collision-free configurations of the arm/hand, and if at least two fingers of the hand have reachable points on the object surface. If they meet these conditions, they are introduced into the data covariance matrix used in PCA. In order to guide the selection of next samples, the main idea is to compute in the goal space a hyperbox $H \subseteq A$ aligned with the new base resulting from the PCA, centered at the mean value of the data, and with the length of each side equal to two times the deviation of the data in the corresponding axis. The expected behavior is that the probability to obtain valid samples in H (leading to valid ICRs on the object) increases at every iteration (Fig. 3).

C. Method 2: Search of grasp poses using a Capability map

In general, an offline analysis of the robot workspace is helpful to speed up the online solution of planning tasks. The representation of the regions where the TCP can be moved to is known as a reachability map [20]. It is computed as a spatial grid in the 6D space (position and orientation), where each cell has a binary value that indicates whether it is reachable or not. The cells can also have an associated quality index that measures the dexterity of the robot when located in this position, thus creating a capability map (Fig. 4). This map is computed offline using a hybrid approach that combines forward and inverse kinematics, to obtain an accurate and structured description of the robot capabilities [12].

For using the capability map, the goal region - where samples for potential hand poses are taken - is defined as a hollow box that surrounds the object. The outer dimension of the box is defined by the size of the hand, and the inner dimension corresponds to the length of the fingers minus the length of the object along that dimension. The goal region represented by such box is intersected with the capability map to provide the regions of the box that are reachable by the robotic arm. Note that this filtering is performed only

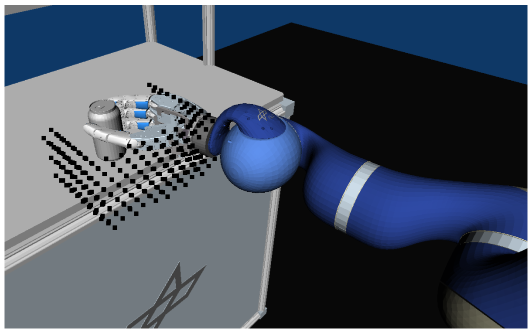

Fig. 5. Points resulting from the filtering phase of the capability map. The points define hand positions reachable by the arm.

once at the beginning of the complete planning process, as it only needs the data of the capability map and the position of the object inside the robot workspace. Fig. 5 shows the result of this filtering step for a soda can; all the represented points are possible hand positions that are inside the goal region and are also reachable by the arm.

During the planning process, samples are taken inside this reachable region. These samples provide the Cartesian position of the hand with respect to the object; the complete transformation T_{hand}^{object} is obtained with the same procedure described in Section III-B. Now, all the points obtained from the capability map in the filtering phase have reachability in at least one direction, but it does not mean that the robot can reach one of those points while pointing towards the object in the desired way. Therefore, the computed transformation T_{hand}^{object} will be checked for reachability using the capability map, before computing the more expensive inverse kinematics.

Usage of the capability map provides two main advantages. First, it defines from the beginning a portion of the goal region reachable by the arm, thus discarding unfeasible areas where otherwise samples would have to be tossed out by calling an IK solver if the reachability information was not verified in advance. Second, it can easily incorporate additional restrictions, for instance, guaranteeing that the robot is always facing the object during the grasp process, which leads to more intuitive solutions of the planning problem.

IV. Integrated Grasp and Motion Planning

This section presents the algorithm that integrates the grasp planning methods, explained in Section III, with an RRT path planning approach for defining the complete grasping motion of the arm/hand system.

Given an initial pose of the object to be grasped and an initial configuration for the arm/hand system, the goal is to find a path leading to the best possible grasp, measured according to the ICR quality described in Eq. (1). The object is described as a pointshell, to facilitate the computation of the ICRs. Algorithm 1 formalizes the approach; particular details of the planner are presented in Algorithms 2 and 3. The planner has two different parts that will be explained in detail below. The first one (Lines 3 to 17) looks for a valid grasp on the object, which defines a goal configuration, while at the same time grows an RRT from the initial arm/hand pose. The second part (Lines 18 to 31) uses a bidirectional RRT to try and connect the initial configuration to the most promising grasp on the object, while still looking for higher quality grasps. The path generated by the algorithm is later smoothed using pruning techniques.

Algorithm 1 Integrated grasp and motion planner

```
1:  procedure PLANNER                                   ▷ Planner main loop
2:      $startTree \leftarrow c_{init}$
3:      $filterCapabilityMap()$ ▷ If Cap. map is used for Alg. 2
4:      while $numgoaltrees = 0$ do                  ▷ PART 1: No goal yet
5:          $p \leftarrow randomUniform(0,1)$
6:          if $p < 0.2$ then
7:              $c_{smp} \leftarrow cspaceUniformSampling()$
8:              $extendTree(startTree)$
9:          else
10:             $c_{goal} \leftarrow findGoal()$                 ▷ Algorithm 2
11:             if $c_{goal} \neq NULL$ then                   ▷ goal found
12:                 $new\ tree$
13:                 $tree \leftarrow c_{goal}$
14:                 $listGoalTrees \leftarrow updateGoalTrees(tree)$
15:             end if
16:         end if
17:     end while
18:     $p \leftarrow randomUniform(0,1)$         ▷ PART 2: Goal found
19:     if $p < 0.8$ then
20:         $indTree \leftarrow whichTreetoConnect()$
            ▷ Probability by Eq. (4)
21:         if $connectTrees(startTree, listGoalTrees[indTree])$
    then
22:             return $pathToGrasp$
23:         end if
24:     else
25:         $c_{goal} \leftarrow findGoal()$                     ▷ Algorithm 2
26:         if $c_{goal} \neq NULL$ then                   ▷ new goal found
27:             $new\ tree$
28:             $tree \leftarrow c_{goal}$
29:             $listGoalTrees \leftarrow updateGoalTrees(tree)$
30:         end if
31:     end if
32: end procedure
```

A. Part 1: Before goal configuration

In the initial part no goal has been defined, so the execution time is split between the search of valid grasps, following one of the methods from Section III, and the random growing of a forward tree $startTree$ from the initial robot configuration. In Algorithm 1 (and in the experiments) the ratio between the two tasks has been empirically set to 80% and 20%, respectively; a thorough analysis on the influence of this ratio on the algorithm performance is still to be performed. The initial exploration of the arm/hand configuration space is useful for the second part of the planner, where a tree growing backwards from the goal has to connect to $startTree$.

Let c denote a configuration in the C-space of the arm/hand system; c_{init} is the initial configuration, c_{smp} corresponds to a configuration obtained via forward kinematics (FK), and c_{goal} is a goal configuration. A goal hand pose is obtained via Algorithm 2 with one of the sampling approaches: PCA-based computation (Section III-B) or capability-based computation (Section III-C). After this, an IK solver is called to verify that the grasp pose is reachable and obtain the corresponding arm configuration.

Next, the validity of the hand pose must be evaluated to verify if there exist reachable ICRs. Reachable ICRs provide contact regions on the object surface, but do not guarantee that the hand configuration (finger positions) leading to them is free of self-collisions between the fingers. Therefore, after the ICRs are computed a hand configuration for grasping the object must be found; this is iteratively done by exploring the potential FC grasps that the regions provide.

Algorithm 2 Find goal configurations

```
1:  procedure FINDGOAL()
2:      $(x, y, z) \leftarrow sampleHandPosition()$
        ▷ Using PCA (Section III-B) or Cap. map (Section III-C)
3:      $T_{hand}^{object} \leftarrow computeTransformation(x, y, z)$
4:      $c_{goal} \leftarrow IK(T_{hand}^{object})$
5:      if $c_{goal}$ then
6:          if $validGoal(T_{hand}^{object})$ then                ▷ Algorithm 3
7:              return $c_{goal}$
8:          else
9:              return $NULL$
10:         end if
11:     else
12:         return $NULL$
13:     end if
14: end procedure
```

Algorithm 3 Verification of goal validity

```
1:  procedure VALIDGOAL($T_{hand}^{object}$)
2:      $regions \leftarrow ICR(T_{hand}^{object})$
3:      if $regions.size() \geq 2$ then
4:          $ICRquality \leftarrow computeICRqual()$            ▷ Eq.(1)
5:          if $ICRquality \geq minICRquality$ then
6:              $findCollfreeHandConfig(T_{hand}^{object}, regions)$
7:              return $true$
8:          else
9:              return $false$
10:         end if
11:     else
12:         return $false$
13:     end if
14: end procedure
```

B. Part 2: Goal configuration found

When the first valid grasp configuration is found, the second part of the algorithm starts. During this phase, the algorithm tries to connect $startTree$ with one of the valid grasp configurations. For a valid goal (grasp) configuration, a new tree $goalTree$ starts to grow backwards. At this point, most of the efforts will be focused on connecting $startTree$ with $goalTree$. However, the search for new valid grasp configurations still continues, although with a lower priority. This continuous search of new configurations looks for better quality grasps or for goals that can be easier to connect to $startTree$.

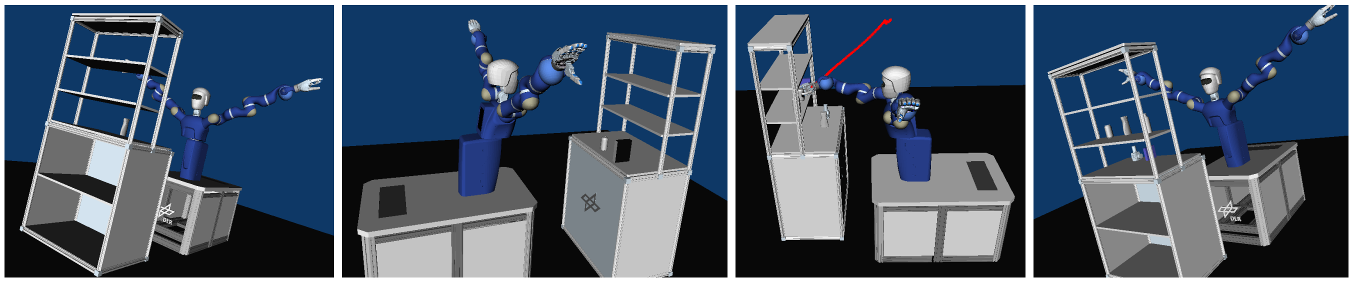
Fig. 6. Tabletop and cupboard scenarios, with and without obstacles, for testing the planning approaches.

The connection process is biased towards the grasps with higher Q_{ICR}. With n goal configurations found, each $goalTree[i]$ has an associated quality Q_{ICR_i}, and its probability to be connected is given by

$$probabilities_i = \frac{Q_{ICR_i}}{\sum_{j=1}^{n} Q_{ICR_j}} \quad (4)$$

The planning procedure comes to an end when the connection between the forward and one of the backwards growing trees is successful, and returns a path in the C-space that allows the robot to grasp the object.

V. Results and Discussion

The proposed approach has been implemented as a specialized RRT planner inside the path planning framework $The\,Kautham\,Project$ [21]. The RRT planner is based on the implementation of RRT-Connect from the Open Motion Planning Library (OMPL) [22]. The computation of the reachable ICRs follows the algorithm proposed in [23], which uses a modified Voxmap-Pointshell (VPS) algorithm for an efficient detection of hand-object collisions [24].

The application examples are solved for SpaceJustin, the upper body of a humanoid robot with 2 arms of 7 degrees of freedom (DoF) each, plus one neck with 2 DoF and one additional DoF in the waist; the robot uses two five-fingered DLR-HIT hands II with 15 DoF each. Two different environments have been tested, one tabletop and one cupboard scenario, grasping different objects with and without obstacles in the way (Fig. 6). Objects different that the target are considered as obstacles.

The same test scenarios were replicated inside the planning framework $Simox$ [25], which includes the planning approach described in [8], hereafter referred to as Grasp-RRT (following the name given by the authors). Thus, a fair comparison of the approaches can be obtained, using the same platform and same geometric models for collision detection. The approaches proposed in this paper will be referred to as PCA-RRT and CAP-RRT, depending on whether the planner searches for grasp poses using the PCA technique or the capability map, respectively.

The comparison is performed for grasping two objects, a soda can and a shampoo bottle. Tables I to IV summarize the computational times and success rate for the planning approaches. N.O. and W.O. stand for scenarios with no obstacles or with obstacles, respectively. 50 runs have been executed for every test, allowing them to run for a maximum time of 100 seconds.

TABLE I
Time to grasp the soda can (s)

Scenario		Grasp-RRT	PCA-RRT	CAP-RRT
Tabletop	N.O	15.32 ± 18.33	13.20 ± 9.68	5.58 ± 3.30
	W.O	18.86 ± 20.87	28.43 ± 17.57	12.66 ± 8.93
Cupboard	N.O	30.23 ± 25.96	15.18 ± 11.08	8.95 ± 7.49
	W.O	38.88 ± 34.83	26.34 ± 16.14	23.79 ± 20.00

TABLE II
Success rate for grasping the soda can

Scenario		Grasp-RRT	PCA-RRT	CAP-RRT
Tabletop	N.O	100 %	100 %	100 %
	W.O	100 %	100 %	100 %
Cupboard	N.O	98 %	98 %	100 %
	W.O	86 %	96 %	98 %

The results show that the integration of motion and grasp planning procedures successfully leads to finding a feasible grasp configuration with its corresponding path for the arm/hand motion, as demonstrated also in the video attachment. Note that the Grasp-RRT is a method that works exclusively based on FK computations, while the two methods proposed in this paper - based on ICRs computation - need some IK calls: to guarantee that the potential grasps are in fact feasible for the arm (PCA-RRT), or to obtain an arm configuration that leads to a given reachable hand pose (CAP-RRT).

Despite working with different philosophies, the Grasp-

TABLE III
Time to grasp the shampoo bottle (s)

Scenario		Grasp-RRT	PCA-RRT	CAP-RRT
Tabletop	N.O	32.99 ± 28.78	11.59 ± 8.09	6.37 ± 3.23
	W.O	36.3± 33.29	24.88 ± 20.80	10.93 ± 6.06
Cupboard	N.O	36.3 ± 33.29	23.59 ± 22.20	5.59 ± 2.83
	W.O	37.95 ± 31.08	34.3 ± 19.63	11.09 ± 6.10

TABLE IV
Success rate for grasping the shampoo bottle

Scenario		Grasp-RRT	PCA-RRT	CAP-RRT
Tabletop	N.O	94 %	100 %	100 %
	W.O	86 %	94 %	100 %
Cupboard	N.O	86 %	98 %	100 %
	W.O	90 %	96 %	100 %

TABLE V
TIME DISTRIBUTION FOR THE CAP-RRT PLANNING APPROACH

Total	Collision	Reachability	ICR	Rest
12.15 s	10.00 s	0.54 s	0.16 s	1.46 s
100 %	82.3 %	4.4%	1.32 %	12.02 %

RRT and the PCA-RRT approaches have a comparable performance in several cases, although the PCA-RRT seems to have less variability in the computational times and behaves better in most of the tested scenarios. However, the best approach turns out to be the CAP-RRT, i.e. the planner that uses information from the capability map to restrict the directions that the robot should use to try and grasp the object. It is faster that the Grasp-RRT (1.5 to 6.5 times, depending on the scenario) and has less variability in the time required to get a solution.

To gain some insight into the time distribution of the planning approach, the averaged times for another 50 runs of CAP-RRT in the tabletop scenario with obstacles are presented in Table V. The times are analyzed for three critical parts: collision detection, computation of reachable points for the hand, computation of reachable ICRs, and for the remaining parts (RRT generation and connectivity). The results show that most of the time spent in the complete procedure goes into collision detection, which is performed using the original meshes of the robot model. However, using simplified models can reduce the weight of the collision detection in the total planning time. Another potential gain in time can be achieved by improving the method for selecting the final finger poses for grasping the object given the ICRs, so that configurations with self-collisions between the fingers can be efficiently avoided.

Finally, as the presented planners do not rely on a precomputed grasp database, they can easily adapt to different object shapes and sizes. Moreover, the approaches do not necessarily require a CAD model of the object in advance; an approximation of the object surface can be used to compute the ICRs online. This computation of contact regions provides robustness to the grasps, compared to grasp plans where only contact points on the object surface are computed. The proposed approaches have a performance that makes them suitable for path planning in realistic scenarios.

ACKNOWLEDGMENT

The authors would like to thank Nikolaus Vahrenkamp for his support on the implementation of the scenarios with the $Simox$ framework, and Katharina Hertkorn for her great support with the implementation of the method on the robot SpaceJustin.

REFERENCES

[1] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis - a survey," *IEEE Trans. Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[2] M. A. Roa and R. Suárez, "Grasp quality measures: Review and performance," *Autnonomous Robots*, 2014, DOI: 10.1007/s10514-014-9402-3.

[3] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 1992, pp. 2290–2295.

[4] J. Kuffner and S. M. Lavalle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2000, pp. 995–1001.

[5] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *Int. J. Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[6] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[7] D. Berenson, R. Diankov, K. Nishikawi, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2007, pp. 42–48.

[8] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Simultaneous grasp and motion planning," in *IEEE Robotics and Automation Magazine*, 2012, pp. 43–57.

[9] M. A. Roa and R. Suárez, "Computation of independent contact regions for grasping 3D objects," *IEEE Trans. Robotics*, vol. 25, no. 4, pp. 839–850, 2009.

[10] K. Hertkorn, M. A. Roa, M. Brucker, P. Kremer, and C. Borst, "Virtual reality support for teleoperation using online grasp planning," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2013, pp. 2074–2074.

[11] K. Hertkorn, B. Weber, P. Kremer, M. A. Roa, and C. Borst, "Assistance for telepresence using online grasp planning," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2013, pp. 507–513.

[12] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *ROBOT2013: First Iberian Robotics Conference*, ser. Advances in Intelligent Systems and Computing 253, M. Armada, A. Sanfeliu, and M. Ferre, Eds. Springer, 2014, pp. 703–718.

[13] D. Berenson and S. Srinivasa, "Grasp synthesis in cluttered environments for dexterous hands," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2008, pp. 189–196.

[14] D. Berenson, S. Srinivasa, D. Ferguson, A. Collet, and J. Kuffner, "Manipulation planning with workspace goal regions," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2009, pp. 618–624.

[15] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robotics Research*, vol. 30, no. 12, pp. 1435–1460, 2011.

[16] M. A. Roa, K. Hertkorn, C. Borst, and G. Hirzinger, "Reachable independent contact regions for precision grasps," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2011, pp. 5337–5343.

[17] J. Rosell, R. Suárez, and A. Pérez, "Path planning for grasping operations using an adaptive PCA-based sampling method," *Autonomous Robots*, vol. 35, no. 1, pp. 27–36, 2013.

[18] J. Rosell, R. Suárez, A. Pérez, and C. Rosales, "Including virtual constraints in motion planning for anthropomorphic hands," in *Proc. IEEE Int. Symp. Assembly and Manufacturing - ISAM*, 2011, pp. 1–6.

[19] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuoka, "Human-guided grasp measures improve grasp robustness on physical robot," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2010, pp. 2294–2301.

[20] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: Representing robot capabilities," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2007, pp. 3229–3236.

[21] J. Rosell, A. Pérez, A. Aliakbar, Muhayyuddin, L. Palomo, and N. García, "The Kautham Project: A teaching and research tool for robot motion planning," in *Proc. IEEE Int. Conf. on Emerging Technologies and Factory Automation, ETFA*, 2014.

[22] I. Şucan, M. Moll, and E. Kavraki, "The open motion planning library," *IEEE Robotics and Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[23] B. A. Dang-Vu, M. A. Roa, and C. Borst, "Extended independent contact regions for grasping applications," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2013, pp. 3527–3534.

[24] M. Sagardia, T. Hulin, C. Preusche, and G. Hirzinger, "Improvements of the voxmap-pointshell algorithm - fast generation of haptic data structures," in *Proc. 53rd Int. Wissenschaftliches Kolloquium*, 2008.

[25] N. Vahrenkamp, M. Kröhnert, S. Ulbrich, T. Asfour, G. Metta, R. Dillmann, and G. Sandini, "Simox: A robotics toolbox for simulation, motion and grasp planning," in *Intelligent Autonomous Systems 12*. Springer, 2013, pp. 585–594.