2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Workspace Aware Online Grasp Planning

Iretiayo Akinola, Jacob Varley, Boyuan Chen, and Peter K. Allen

***Abstract*— This work provides a framework for a workspace aware online grasp planner. This framework greatly improves the performance of standard online grasp planning algorithms by incorporating a notion of reachability into the online grasp planning process. Offline, a database of hundreds of thousands of unique end-effector poses were queried for feasibility. At runtime, our grasp planner uses this database to bias the hand towards reachable end-effector configurations. The bias keeps the grasp planner in accessible regions of the planning scene so that the resulting grasps are tailored to the situation at hand. This results in a higher percentage of reachable grasps, a higher percentage of successful grasp executions, and a reduced planning time. We also present experimental results using simulated and real environments.**

I. Introduction

Grasp planning and motion planning are two fundamental problems in the research of intelligent robotic manipulation systems. These problems are not new, and many reasonable solutions have been developed to approach them individually. Most of the research has treated these problems as distinct research areas focusing either on: 1) how to get a set of high quality candidate grasps [1][2] or 2) how to generate viable trajectories to bring the gripper to the desired hand configuration [3][4][5]. While it is often convenient to treat grasp and motion planning as distinct problems, we demonstrate that solving them jointly can improve performance and reduce computation time. Grasp planners unaware of the robot workspace and without a notion of reachability often spend significant time and resources evaluating grasps that may simply be unreachable. For example, a reachability unaware planner is equally likely to return impossible grasps that assume the robot will approach the object from the object's side furthest from the robot. Our planner avoids unreachable areas of the workspace dramatically reducing the size of the search space. This lowers online planning time, and improves the quality of the planned grasps as more time is spent refining quality grasps in reachable portions of the workspace, rather than planning grasps that may be stable but are unreachable.

In order to generate grasps which are stable and reachable, we propose a new energy function for use with simulated annealing grasp planners [1][2]. This energy function is a weighted combination of a grasp stability term, and a grasp reachability term. Our grasp reachability term is inspired by the idea of a precomputed reachability space which has demonstrated utility in other robotics tasks [6][7]. In our work, we generate a densely sampled reachability space for our robot as shown in Fig. 1(Top Row). This offline computation checks whether an Inverse Kinematic (IK) solution exists for a given pose. This database in and of itself has utility for filtering lists of precomputed-grasps and quickly removing unreachable grasps, leaving only reasonable grasps as possible results. We further postprocess our reachability space and compute a Signed Distance Field (SDF) representation in Fig. 1(Bottom Row). The SDF is used in our online grasp planning energy function to guide the hand towards reachable regions of the robot's workspace. Obstacles present in the grasping scene can be embedded into the space prior to SDF generation which results in a field that repels the grasp search away from poses that collide with the obstacles.

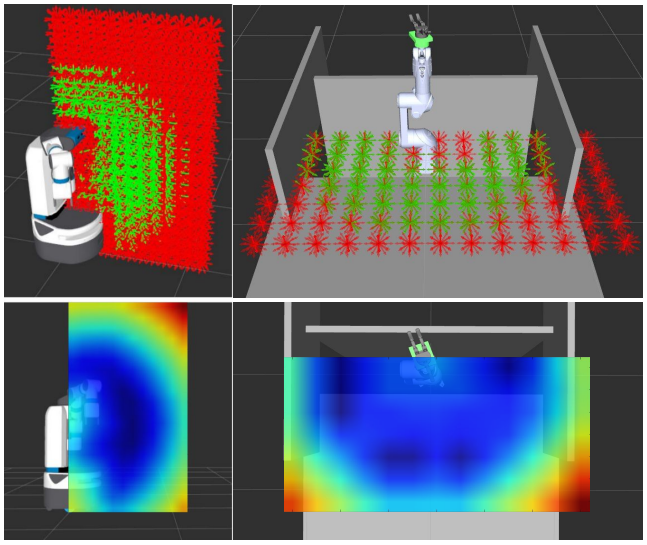


Fig. 1: Top Row: Visualization of cross sections of the precomputed reachable space for a Fetch Robot and Staubli Arm with Barrett Hand. Green arrows represent reachable poses, red arrows unreachable. This space is computed offline, once for a given robot. Bottom Row: Signed Distance Field generated from the above reachability spaces.

Experiments in both simulation and physical environments have been performed to assess the performance of our method on multiple objects and under various poses. We demonstrate that our reachability aware grasp planner results in a larger number of reachable grasps,

This work was supported in part by National Science Foundation grants IIS-1734557 and IIS-1527747. Authors are with the Department of Computer Science, Columbia University, New York, NY 10027, USA. (iakinola, jvarley, bchen, allen)@cs.columbia.edu,

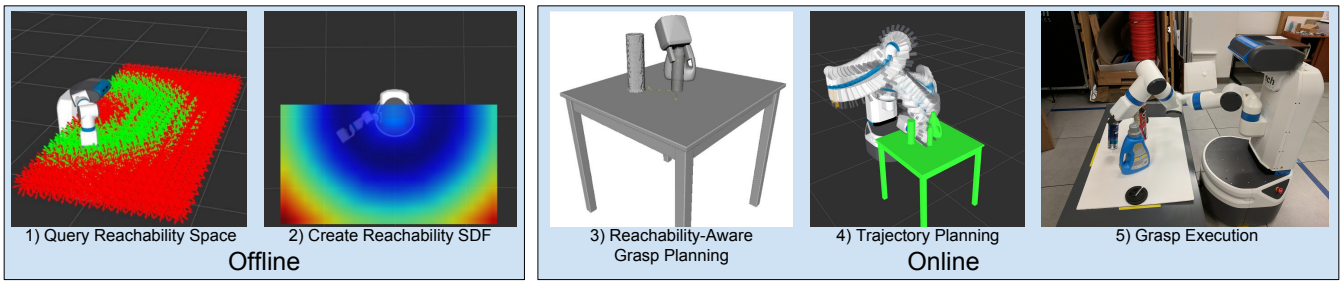


Fig. 2: Workspace Aware Online Grasping Framework - **Offline**: 1) the robot's reachability space is queried for IK solutions that are free of collisions with the robot itself and fixtures such as walls and tables. 2) An SDF is created from the reachability space. **Online**: 3) Grasp planning is quickly accomplished utilizing the reachability space SDF. 4) A motion plan for one of the planned grasps. 5) Trajectory executed by the robot for a stable grasp.

a larger number of successful grasp executions, and a reduced planning time compared to other online grasp planning methods.

Our method for workspace aware online grasp planning is overviewed in Fig. 2. This framework has a number of advantages. First, it combines two closely coupled processes into one; rather than going back and forth between grasp planning and trajectory planning until a feasible grasp is found, we solve for a fully reachable grasp at once. It increases the probability of finding feasible grasps quickly since the optimization process is guided by our energy function within reachable spaces. Also, our method is well suited for online planning which in general is more adaptable and robust than the traditional offline grasp planning approach and works in the case of novel objects for which precomputed grasps do not exist.

In summary, we take a closer look at the problem of grasp quality and reachability analysis of grasps in an integrated and structured manner. The contributions of this work include: 1) A novel representation of reachability space that uses signed-distance field (SDF) to provide a gradient field during grasp planning; 2) An online grasp planning system with an integrated notion of reachability; 3) A formulation that ranks a list of grasps (e.g. from a database) based on their combined reachability and grasp quality; 4) A method of embedding of obstacles into the reachability space during grasp planning; 5) experimental results using our framework compared to other grasp planning solutions evaluated in multiple grasping scenarios.

II. Related Work

Different works have highlighted the problem of generating non-reachable planned grasps from grasp planning systems. For example, a 2016 review of the Amazon Picking Challenge (APC) [8] explicitly highlighted the fact that many teams opted against the planning paradigm (choosing an online feedback paradigm instead). Since grasp planning metrics typically ignore the notion of reachability, a large percentage of the planned grasps end up being unreachable and not useful for the robot as illustrated in Fig. 3 . Hence, an extra computational step is required to check each of the planned grasps for reachability; a top performing APC team [9] uses this post-check solution which shows that this limited partial solution is built into even competitive systems. This post-check solution has a number of drawbacks: first, the post-processing check for reachability might not contain any reachable grasp which implies that the entire grasp planning process has to be repeated. Another closely related problem with post-reachability checking is that more time is incurred when replanning grasps that fail the reachability test stage. Also, many systems, such as those in APC, are highly tuned to specific use-cases which do not generalize well especially when the grasping environment changes in small ways. Our system presents a way to automatically incorporate the environment into grasp planning by augmenting the reachability space.

Many grasp planning platforms such as GraspIt![10], OpenRave[5], and OpenGrasp[11] only consider properties specific to the end-effector during grasp planning. Given the hand description and the object, these platforms compute grasp quality for different hand-object configurations using metrics such as force/form closure and return the best quality grasps found. This approach ignores reachability of the resulting grasp as often the arm is not even modeled in the simulator, and leaves the burden of verifying the grasp outputs to the subsequent stages of robotics applications. Our work instead incorporates the notion of reachability during planning thereby increasing the quality of the output of the grasp planning processing, from the viewpoint of the subsequent stages. The reachability space aptly captures the information about the robot geometry and kinematics; our approach incorporates this information into the grasp planning process in a way that produces feasible reachable results as input for trajectory planning. Since this involves a constant time lookup, incorporating it gives extra advantage at slight computational cost.

The reachability space presents an intuitive way to introduce soft constraints to the grasp planning process. One can restrict the allowable grasp configuration space by adjusting the reachability space to reflect such pref-

erences. Though we have not shown it in this work, this idea has been demonstrated in a previous work [12]. In their work, they use an *"Environment Clearance Score"* to guide grasps away from obstacles in the robot workspace. The same work introduced the idea of *"Grasp-scoring function"* which uses an approximate kinematics of the robot to score a grasp. This is different from our work in the sense that we build the actual reachable space using the exact kinematics of the robot. Also, while they use the scoring function on precomputed grasps, we use the reachability value directly for online grasp planning.

The concept of offline reachability analysis exists in the literature [7] [6]. However, many of these works use offline precomputed grasps. While our approach shares the concept of offline reachability analysis with these works, we differ in that we incorporate the reachability information directly into online grasp planning. For example, a method [13] uses capability maps and an additional inverse kinematics check to filter out sampled grasp poses. Since their grasp sampling is uniform, the chances of sampling reachable poses do not improve over time. Conversely, our method uses a gradient field to guide the grasp planning process to reachable portions. Note that our work treats grasping as a self-contained task that comes up with useful outputs for subsequent stages of robotic operations while these works give more consideration to the manipulation tasks.

Haustein etal.[14] has recently looked at combining grasp planning and motion planning. Similar to [15], their method uses a bidirectional search approach to randomly sample grasp candidates in one direction and sample arm trajectories in the opposite direction until a connection is made to form a hand grasp/arm trajectory pair. Our method is complementary to theirs in that we guide the very high-dimensional grasp goal sampling process to regions that are reachable leading to faster connection of sample grasps to the sampled arm trajectories. Our work provides a simple and effective upgrade to common standalone grasp planning approaches as well as to the inner grasp search modules of integrated grasp and motion planning systems. While our work is similar to [12] [7] in being workspace aware, ours differs in a number of ways: (1) we focus on solving online grasp planning, (2) we compute the actual reachability map, not an approximation as in [12], (3) we use a novel representation of the reachability space, and (4) we use a different optimization technique, simulated annealing.

III. Offline Reachability Space Generation

A grasp is reachable if a motion plan can be found to move the arm from its current configuration to a goal configuration that places the hand at desired grasp location. This is not always possible for a number of reasons typically because no inverse kinematics(IK) solution exists to place the hand at the desired grasp pose, self collision with other parts of the robot, or collision with obstacles in the planning scene.

Algorithm 1 Reachability Space Generation

```
1: procedure GenerateReachabilitySpace
2:     poses = uniformSampleWorkspace()
3:     pose2Reachable = {}
4:     for pose in poses do
5:         reachable = hasCollisionFreeIK(pose)
6:         pose2Reachable[pose] = reachable
7:     SDF = computeSDF(pose2Reachable)
8:     return SDF
```

A. Reachability Space Representation

We observe that this definition of the original reachability space is binary, either 1 (reachable), or 0 (not reachable), and has no gradient to indicating the direction from non-reachable regions to reachable portions of the workspace. In the context of Simulated Annealing for grasp planning, it is beneficial to provide an energy function that has a landscape that can guide solutions to more desired regions of the annealing space. This observation informed a novel signed-distance-field (SDF) representation of the reachability space that is amenable to optimization formulations. Our SDF maps a pose to a value representing the distance to the manifold or boundary between the unreachable poses and reachable poses and can be interpreted as follows $d_{sdf} = SDF(pose)$:

- $d_{sdf} = 0$: pose lies exactly on boundary between reachable and unreachable grasp poses.
- $d_{sdf} > 0$: pose lies within the reachable region of the workspace, and is distance d_{sdf} away from the boundary.
- $d_{sdf} < 0$: pose lies outside the reachable region of the workspace, and is distance d_{sdf} away from the boundary.

This field can then serve as criteria for classifying a grasp as either lying in a reachable space or not. And nearby hand configurations can be ordered based on d_{sdf}, their distance from this boundary.

Grid resolution and metric choices are two important parameters for the reachability space and SDF generation. The 6D hand pose of a given grasp has the 3D translational (linear) component and the 3D rotation (angular) component, two physically different quantities with different units. We had to systematically determine good resolution levels and a good ratio between unit linear measurements and unit angular measurements, to obtain a unified 6D metric space suitable for our purpose. Effectively, our metric can be defined as:

$$d_{sdf} = \pm\sqrt{||\Delta xyz/res_{lin}||^2 + r||\Delta rpy/res_{rot}||^2} \quad (1)$$

where Δxyz and res_{lin} (in centimeters) are the translational distance and resolution respectively, Δrpy and res_{rot} (in radians) are the rotational distance and resolution, and r is the relative metric ratio i.e. a distance of res_{lin}cm $\equiv r * res_{rot}$rad.

We varied translational resolution $res_{lin} = 5, 10, 20$cm, angular resolution $res_{rot} = \pi/8, \pi/4, \pi/2$rad, and translational to rotational metric ratio $r = 0.1, 1, 2, 5, 10$; and checked the performance of the SDF generated for each combination. First, we randomly generated 10,000 grasp poses and checked for the existence of an IK solution i.e. the grasp's reachability. Then, for each of the resolution levels, we evaluated the quality of the resulting SDF by checking what percentage of the random grasp poses were correctly classified as reachable (or not) by the SDF manifold. Reachable grasps evaluate to positive sdf values, unreachable grasps evaluate to negative sdf values. The time to generate SDF from binary reachability space was recorded in each case. From this parameter sweep, the densest resolution level res_{lin} = 5cm, $res_{rot} = \pi/8$rad gave 0.992 & 0.973 as accuracy and precision respectively but took 177secs to generate the SDF. We observed that 10cm translational, $\pi/4$ rotational resolution levels and metric ratio $r = 1$ still gave a high accuracy (0.979) and precision (0.92) with the sdf generation time under 3secs. These choices defined the metric space for the 6D hand pose reachability space: a 10cm translation and a $\pi/4$rad rotation are equidistance in the reachability space. Note that we take into consideration the cyclical nature of the rotational degrees of freedom during SDF generation as angles wrap around at 2π.

B. Reachability Space Generation

Reachable-space generation is a one-time process for a given robot. Algorithm 1 details the process for generating the Reachable Space. We search a discretization of the 6D pose space of the robot to determine which configurations will be reachable by the hand. For the Fetch Robot, we sampled the workspace using the values in meters centered at the robot's base shown in Table I. In total 675,840 unique poses where queried for reachability. 102,692 were reachable and 573,148 were unreachable. This was computed using Moveit!'s IK solver which checks for a valid collision free IK solution for a given pose. The 6D pose space containing the binary IK query results was converted into an SDF using the fast marching method from scikit-fmm[1]. Once this computation is done, the SDF can be queried with any pose inside the space to determine both its binary reachability and distance to the boundary separating reachable and non reachable poses. Reachability space generation codes and data will be made available on the project website: `http://crlab.cs.columbia.edu/reachabilityawaregrasping/`

IV. Online Reachability-Aware Grasp Planning

Grasp planning is the process of finding "good grasps" for an object that can be executed using an articulated robotic end effector. A grasp is typically classified as "good" based on how well it can withstand external disturbances without dropping the object. The Ferrari-Canny method [16] is a common way to evaluate robot grasps; it defines how to quantitatively measure the space of disturbance wrenches that can be resisted by a given grasp. Other techniques are typically built around this metric as shown in this review [17]. Grasp planning frameworks such as GraspIt! use these grasp metrics of force and form closure as objective functions for optimization. Since this formulation has no notion of reachability, the grasp results, while stable, might require the end effector to be placed in a pose that is impossible to reach given the robot's current location.

[1]https://github.com/scikit-fmm/scikit-fmm

TABLE I: Discretization of robot workspace for SDF generation for the Fetch robot. Values are in meters (x, y, z) and radians (roll, pitch, yaw). The origin of the space is centered at the robot base. The roll was sampled less because the Fetch robot can twist its wrist to achieve any roll for a give hand pose.

	X	Y	Z	Roll	Pitch	Yaw
min	0.0	-1.1	0.0	-PI	-PI	-PI
max	1.2	1.1	2.0	PI	PI	PI
step	0.1	0.1	0.1	PI	PI/4.0	PI/4.0

A. Simulated Annealing for Grasp Planning

Grasp planning can be thought of as finding low energy configurations within the hand object space. This search is done in a multidimensional space where grasps are sampled and evaluated. The 6 dimensional (x, y, z, roll, pitch, yaw) space we generated with our SDF is a subspace of the 6 + N dimensional grasp search space. The additional N dimensions represent the EigenGrasps or eigen vectors of the end effector Degree of Freedom (DOF) space as described in [1]. For the Fetch (1 DOF), N=1, and for the Barrett Hand (4 DOF), N=2. These 6 + N dimensions describe both the pose and DOF values of the end effector. The goal of grasp planning is to find low energy points in this 6 + N dimensional space. These points represent the pose and hand configuration of "good grasps".

Simulated annealing (SA) [18] – a probabilistic technique for approximating global optimums for functions lends itself nicely for our formulation. SA presents a number of advantages in grasp planning[19]. For example, it does not need an analytical gradient which can be computationally infeasible; it handles the high nonlinearity of grasp quality functions; and it is highly adaptable to constraints. Our approach implicitly guides the annealing process ensuring that sampling is done in regions of reachable space which increases the probability of getting useful grasp solutions. While online grasping is typically avoided due to time cost of exploring a larger search space, this work makes online grasping more feasible since the majority of the space which is unreachable need not be searched. With our new energy formulation, the annealing process will quickly drive the hand towards reachable grasp locations. Section V below

shows that this new energy function increases the success probability of online grasping significantly.

Algorithm 2 Contact And Potential Energy (SA-CP)

```
1: procedure ContactAndPotentialEnergy
2:     e_p = potentialEnergy()
3:     stable = e_p < 0
4:     if stable then
5:         return e_p
6:     else
7:         e_contact = contactEnergy()
8:         return e_contact
```

Algorithm 3 Reachability-Aware Energy (SA-OURS)

```
1: procedure ReachabilityAwareEnergy
2:     e_p = potentialEnergy()
3:     e_reach = reachabilityEnergy()
4:     stable = e_p > 0
5:     reachable = e_reach < 0
6:     if reachable && stable then
7:         return e_p + α_1 · e_reach
8:     else if reachable && !stable then
9:         e_contact = contactEnergy()
10:        return e_contact + α_2 · e_reach
11:    else //!reachable
12:        e_contact = contactEnergy()
13:        return e_contact + α_3 · e_reach
```

B. Novel Grasp Energy Formulation

In our work, we augment the *Contact and Potential* grasp energy function from [1]. The *Contact and Potential* energy function is described in Algorithm 2 and consists of a potential energy term (e_p) measuring the grasps potential to resist external forces and torques, and a contact energy term ($e_{contact}$) that defines the proximity of the hand to the object being grasped. These two terms define the grasp fitness function used by the optimization algorithm (simulated annealing) to search for good grasp configurations. Our *Reachability Aware* method augments the *Contact and Potential* energy function with a *reachability* energy term (e_{reach}) that encodes the kinematic constraints of the robot. We use the reachability SDF ($e_{reach} := d_{sdf}$) described previously as a regularizing term to obtain a new cost functions that drives the grasp planning optimization towards reachable hand configurations. As shown in [20] discretized SDFs still model and behave like a continuous function so it can be used during the simulated annealing optimization process to drive the search towards reachable hand configurations. Our reachability-aware energy function E is given as:

$$E = G + \alpha R \tag{2}$$

where G is a metric of grasp quality (e.g. force closure), R is a measure of reachability of a given grasp pose and α defines the tradeoff between the conventional grasp metric and the reachability value. To optimize the overall grasp energy, we use the simulated annealing search according to [21].

In terms of implementation, we build on the GraspIt! simulator and combine the existing energy metrics with the new reachable energy term. The energy components have forms that we exploit. The grasp energy term (as described in [1] and Algorithm 2) is negative when the grasp has force closure and positive otherwise. The reachability energy has similar meaning which gives four quadrants of possibilities. We chose the weights of each quadrants so the range of values for the reachability term and grasp term are of the same order of magnitude. This provides a gradient from bad to good quality (reachable and force-closure) grasps. Algorithm. 3 shows specifically how the energy terms are combined. α values were set to $\alpha_1 = -0.1$, $\alpha_2 = -10$, $\alpha_3 = -10$ to obtain a function where more negative energy values correspond to more reachable stable grasps, and positive high energy values correspond to unreachable and unstable grasps.

Since our reachability space is represented as a grid of discretized SDFs, we use multi-linear interpolation to get ($e_{reach} := d_{sdf}$) for a given grasp pose from the reachability SDF grid. Each query pose during the optimization is situated in the robot's reachable space to get the cell location and the reachability value is given as the weighted sum of the values for the 2^N ($N = 6$) corners of the hypervoxel where the query point falls in the pre-computed SO(6) reachable space. The 2^N corner values are looked up in the pre-computed reachability space.

$$R[p_{query}] = \sum_{i=1}^{2^N} w_i R[p_i] \tag{3}$$

where $p_{query} = [x_q, y_q, z_q, r_q, p_q, y_q]$ is the query pose, $p_i = [x_i, y_i, z_i, r_i, p_i, y_i]$ for $i = 1...2^N$ correspond to the neighbouring corners in the reachability space grid and w_i is the proportion of the grid occupied by creating a volume from connecting the query point and the corner p_i. See [22] for more details.

C. Embedding Obstacles in the Reachability Space

Obstacles whose poses are not expected to change in relation to the robot can be placed in the scene while generating the initial reachability space. This approach is applicable to modeling tables, walls and fixtures around fixed base robotic arms, or static room environments for mobile manipulators. This is useful for the Staubli Arm and attached Barrett Hand as shown in Fig. 1. This robot is fixed to the table, next to the walls, making it reasonable to incorporate these obstacles during the initial reachability space creation.

In order to incorporate other static obstacles detected at runtime into our precomputed reachability space, we mask out regions in the robot's binary reachability space that overlap with any detected obstacle prior to generating the SDF representation; this corresponds

to modifying the reachable space in the first block of Fig. 2. Given the obstacles' geometries and poses, we first collate grid locations of the reachability space that overlap with obstacle geometries. All grasp poses at these locations (including those that were otherwise reachable) are marked as unreachable. Then, we regenerate an SDF representation of the resulting binary reachable space. The SDF generation is fast, taking 2-3 seconds on average, and this step is only required when the workspace changes. While this fast obstacle-masking-procedure considers only the hand (not full arm configuration) and might miss out on some other grasp poses that become infeasible due to obstacles, we've found that it is a reasonable approximation that works well in practice, especially when the obstacles affect arm links close to the end-effector. The masked out obstacle locations imposes a negative field that pushes the SDF manifold further away from the obstacles and our experiments show that this effect results in an improved grasp planner.

V. Experiments

We describe different experiments in both simulation and physical environments to assess the performance of our method on multiple objects and under various poses. All experiments (simulated and real) were run on two robot platforms: the Fetch mobile manipulator and the Barrett hand mounted on a Staubli Arm (Staubli-Barrett). The Staubli-Barrett set-up (see Fig 1) has additional walls to limit the workspace of the robot as a safety precaution to ensure the robot does not elbow objects outside of it's workspace during grasping. This safety box heavily constrains the range of grasping directions as many grasp poses will be invalidated because of collision with the walls. Workspace fixtures such as the walls were included in the scene when constructing the reachability space offline.

For each experiment, we compared a typical grasp planning method with our reachability-aware method:
Sim. Ann. Contact and Potential (SA-C&P): Simulated Annealing in GraspIt! using the Contact and Potential energy function from Algorithm 2.
Sim. Ann. Reachability-Aware (SA-Ours): Simulated Annealing in GraspIt! using the newly proposed energy function from Algorithm 3.

A. Evaluation Metrics

The metrics used for evaluating our methods include:
Percent Reachable Grasps: The number of reachable grasps divided by the total number of grasps generated from a run of a given grasp planner.
Number of Required Plan Attempts: The grasp planner returns a list of grasps ordered according to quality. We record how many of these grasps we have to go through until a valid path can be planned.
Lift Success: Here we execute the first valid grasp from the set returned by the planner. A grasp is successful if the gripper placed at the grasp pose can successful lift the object off the ground. We use the Klamp't simulator [23] to test this in simulation. For the real world experiments, we executed the grasps on the real robot and checked if the object was picked up.

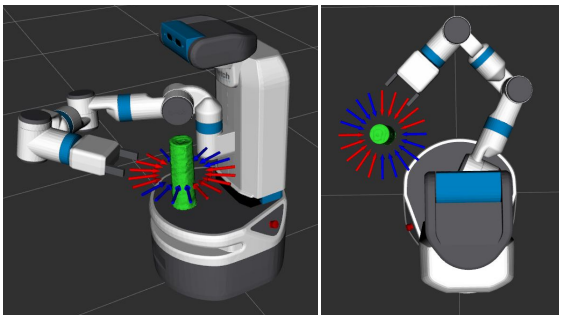
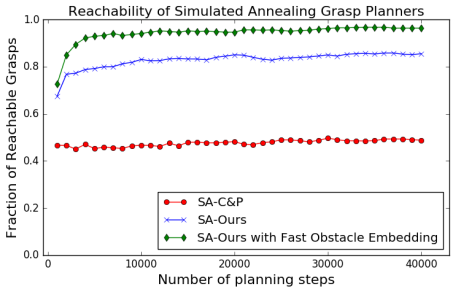


Fig. 3: Reachability results for uniformly sampled grasps (Red arrows reachable, blue unreachable). Even for objects well within the bounds of the robot workspace, there are many invalid approach directions which are not easily modeled by simple heuristics.

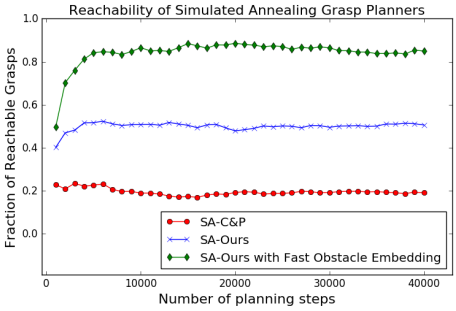
B. Grasp Planning with Runtime Obstacles Experiment

Here we set-up a typical grasp planning scenario: in a virtual scene, three objects were placed on a table. One is the target object to be grasped while the other two are "runtime detected" obstacles that the robot must not collide with during grasping (Fig 5). We used 4 different meshes as the target object. We placed the target object in 9 difficult-to-reach poses that were either at the extent of the robot's workspace, or extremely close to the robot. Both of these situations drastically limit the number of valid approach directions that the robot can use to grasp the object. For each pose, GraspIt!'s Simulated Annealing planner was run for varying number of planning steps using both the **SA-C&P** and **SA-Ours** energies. We used two versions of our reachability-aware planner (**SA-Ours**), one with *unmodified SDF* (Section IV-B) and the other with *obstacles-embedded SDF* (Section IV-C). For each planner, we checked the reachability of all planned grasps and evaluate the fraction of planned grasps that are reachable after running the planner for a given number of steps.

The results (Fig 4) show the significance of our reachability-aware grasp planning approach compared with a typical grasp planner on the same setup. Given the same amount of time, the planners each return twenty grasp solutions and we check what fraction of them are achievable by the robot and report the average over all 36 simulated object poses (4 objects, 9 poses). As expected, a typical grasp planner that has no notion of reachability (shown in red) will yield a large percentage of grasps that are not feasible for the robot. Even with additional planning time, there is no improvement in the odds of returning reachable grasps. On the other hand, our method (shown in blue) yields significantly higher fraction of reachable grasps ($>$ 25% improvement). In addition, the fraction of reachable grasps increases



(a) Fetch Robot Results



(b) Staubli-Barrett Results

Fig. 4: Mean fraction of reachable grasps using different energy functions for varying planning duration. The red plot shows that using pure grasp planning with no notion of reachability gives grasp results that have a low chance of being reachable (48.2% for Fetch and 18.8% for the Staubli-Barrett). The blue plot shows that our reachability aware grasp planner results in a higher fraction (> 25% increase) of reachable grasps for both robots. The green plot shows the additional gain obtained when we embed obstacles into the SDF reachability space. This results in reachable grasps that have feasible IK results and do not collide with the obstacles hence an increase in the overall fraction of reachable grasps.

quickly with planning time as our method optimizes for both stability and reachability of grasps. While the blue plot uses the original offline SDF, the green plot shows that by embedding the runtime detected obstacles into the precomputed SDF, we are able to increase the percentage of reachable grasps (Fetch: 95.7%, Staubli-Barrett: 86.1%) compared to (84.3%, 50.2%) respectively when the precomputed SDF only avoids self collision and collision with workspace fixtures such as the walls.

After demonstrating that our method increases the feasibility of grasp planning results, we also observe that this method also leads to a significant speedup of the grasp planning process since the reachability energy term guides the annealing process quickly to reachable regions hence reducing the search space. Once GraspIt! returns a list of grasps, we iterate through the list in order of grasp quality and attempt to plan a valid path for each grasps. When a valid grasp is found, we use the Klampt! simulator to get the lift success. For a simple parallel jaw gripper like the Fetch gripper, there was no significant difference in the percentage of lift success (**SA-C&P**: **0.95** and **SA-Ours**: 0.93) but our method requires less number of motion planning attempts (**SA-C&P**: 2.75 and **SA-Ours**: **1.10**) to find a valid grasp in the list and overall returns a higher percentage of useful grasps (reachable and lift success). Our grasp energy formulation ensures that a top ranked grasp has a high chance of being reachable. The difference in grasp quality was more pronounced with the 4 DOF Barrett hand. Our method not only requires less number of motion planning attempts (**SA-C&P**: 6.2 and **SA-Ours**: **1.27**), it also achieves 20% higher lift success rate (**SA-C&P**: 0.75 and **SA-Ours**: **0.95**). This is because our planner spends most of it's time refining grasps in the much-reduced reachable regions.

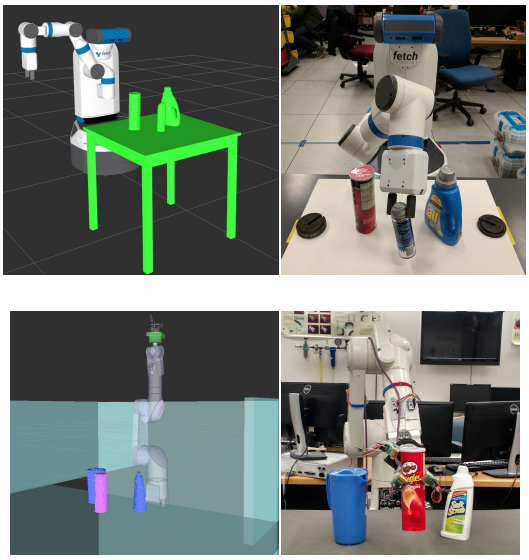


Fig. 5: Crowded scene for real world experiments. **Top** (Fetch Robot): Our Reachability-Aware planner (SA-Ours) was able to successfully grasp the shaving cream bottle 3/3 times, while annealing without the reachability space (SA-C&P) failed 2/3 times. **Bottom** (Staubli-Barrett Robot): SA-Ours successfully grasped the pringles bottle 5/5 times, while SA-C&P failed 3/5 times.

TABLE II: Grasp success results on real robot (Fetch) with a crowded scene. Each method was given 3 attempts to plan and execute a grasp on the shaving cream bottle.

Crowded Scene Grasp Planning (Fetch)			
Search Energy	# steps	Success Rate	Mean Grasp Planning Time (s)
SA-Ours	10K	**100.0%**	8.706
SA-C&P	10K	33.3%	8.360
SA-C&P	40K	66.6%	32.31

C. Real Robot Crowded Scene Experiment

To verify our planner and demonstrate that it works outside of simulation, the simulated annealing based grasp planner was run with a variable number of annealing steps and the success rate was reported in Table II and III. This experiment compares two grasp energy formulations: **SA-Ours** and **SA-C&P**.

Multiple objects were placed in the planning scene as shown in Fig 5. The additional objects reduce the range

TABLE III: Grasp success results on real Staubli-Barrett robot with a crowded scene (Fig 5). Each method was given 5 attempts to plan and execute a grasp on the pringles bottle.

Crowded Scene Grasp Planning (Barrett)			
Search Energy	# steps	Success Rate	Mean Grasp Planning Time (s)
SA-Ours	10K	**100.0%**	11.36
SA-C&P	10K	40%	9.53
SA-C&P	40K	60%	31.98

of possible grasps since the obstacles make many grasp poses infeasible. Note that both methods are aware of the obstacles during grasp planning and avoid grasps that put the hand in collision with obstacles. Table II shows that our method, **SA-Ours**, is able to achieve grasp success – with the Fetch robot picking the object three times out of three within the limited planning time. Conversely, **SA-C&P** fails to produce a reachable grasp 2/3 times when run for 10,000 steps, and fails once even when it is allowed to run for 40,000 steps. Despite being allowed to run for a long duration, the naive planner spends much of its time exploring the back half of the bottle which is completely unreachable to the Fetch robot.

We repeated the same experiment with Staubli-Barrett robot (bottom of Fig 5). Though the objects were placed roughly at the centre of the workspace, the presence of walls and distractor objects significantly limit the range of feasible grasps for the target object. Each method had 5 trials and the results followed the same pattern: **SA-Ours** achieves grasp success all five times within the limited planning time (10,000 steps) while **SA-C&P** fails 2/5 times even when run for 40,000 steps.

VI. Conclusion and Future Work

This work provides a framework for a workspace aware grasp planner. This planner has greatly improved performance over standard online grasp planning algorithms because of its ability to incorporate a notion of reachability into the online grasp planning process. This improvement is accomplished by leveraging a large precomputed database of over 675,840 unique end-effector poses which have been tested for reachability. At runtime, our grasp planner uses this database to bias the hand towards reachable end effector configurations. This bias allows the grasp planner to generate grasps where a significantly higher percentage of grasps are reachable, a higher percentage result in successful grasp executions, and the planning time required is reduced. It has been experimentally tested in both simulated and physical environments with different arm/gripper combinations.

Several future research directions include: utilizing our computed reachability workspace to help mobile robots navigate to optimal locations for manipulating objects and improvements for speeding up the process of incorporating dynamic objects into our notion of reachability to further assist the grasp planner.

References

[1] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *Intelligent Robots Systems*. IROS, 2007, pp. 3270–3275.

[2] K. Hang, J. A. Stork, and D. Kragic, "Hierarchical fingertip space for multi-fingered precision grasping," in *Intelligent Robots and Systems (IROS)*. IEEE, 2014, pp. 1641–1648.

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

[4] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *IJRR*, vol. 33, no. 9, pp. 1251–1270, 2014.

[5] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, vol. 79, 2008.

[6] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *ROBOT2013: First Iberian Robotics Conference*, pp. 703–718.

[7] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *IROS*. IEEE, 2009.

[8] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman, "Analysis and observations from the first amazon picking challenge," *IEEE Transactions on Automation Science and Engineering*, 2016.

[9] C. Hernandez, M. Bharatheesha, W. Ko, H. Gaiser, J. Tan, K. van Deurzen, M. de Vries, B. Van Mil, J. van Egmond, R. Burger *et al.*, "Team delft's robot winner of the amazon picking challenge 2016," *preprint arXiv:1610.05514*, 2016.

[10] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[11] B. León, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moisio, J. Bohg, J. Kuffner *et al.*, "Opengrasp: a toolkit for robot grasping simulation," in *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, 2010, pp. 109–120.

[12] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Humanoid Robots, 2007 7th IEEE-RAS Internatn'l Conf. on*, pp. 42–48.

[13] F. Zacharias, C. Borst, and G. Hirzinger, "Online generation of reachable grasps for dexterous manipulation using a representation of the reachable workspace," in *Advanced Robotics. ICAR 2009. International Conference on*, pp. 1–8.

[14] J. A. Haustein, K. Hang, and D. Kragic, "Integrating motion and hierarchical fingertip grasp planning," in *Robotics and Automation (ICRA), 2017 IEEE Int'l. Conf.*, pp. 3439–3446.

[15] J. Fontanals, B.-A. Dang-Vu, O. Porges, J. Rosell, and M. A. Roa, "Integrated grasp and motion planning using independent contact regions," in *Humanoid Robots, 2014 14th IEEE-RAS International Conference on*. IEEE, 2014, pp. 887–893.

[16] C. Ferrari and J. Canny, "Planning optimal grasps," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE, 1992, pp. 2290–2295.

[17] M. A. Roa and R. Suárez, "Grasp quality measures: review and performance," *Autonomous Robots*, vol. 38.

[18] L. Ingber, "Very fast simulated re-annealing," *Mathematical and computer modelling*, vol. 12, no. 8, pp. 967–973, 1989.

[19] P. K. Allen, M. Ciocarlie, and C. Goldfeder, "Grasp planning using low dimensional subspaces," in *The Human Hand as an Inspiration for Robot Hand Development*, 2014, pp. 531–563.

[20] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed distance fields: A natural representation for both mapping and planning," in *RSS 2016 Workshop: Geometry and Beyond-Representations, Physics, and Scene Understanding for Robotics*. University of Michigan, 2016.

[21] M. T. Ciocarlie and P. K. Allen, "Hand posture subspaces for dexterous robotic grasping," *IJRR*, vol. 28, no. 7, 2009.

[22] R. Wagner, "Multi-linear interpolation," *Beach Cities Robotics*, 2008.

[23] K. Hauser, "Robust contact generation for robot simulation with unstructured meshes," in *Robotics Research*, 2016.