Manipulation Trajectory Optimization with Online Grasp Synthesis and Selection

Lirui Wang,1 Yu Xiang,2 Dieter Fox1,2
1University of Washington, 2NVIDIA
liruiw@cs.washington.edu, yux@nvidia.com, fox@cs.washington.edu

Abstract
In robot manipulation, planning the motion of a robot manipulator to grasp an object is a fundamental problem. A manipulation planner needs to generate a trajectory of the manipulator arm to avoid obstacles in the environment and plan an end-effector pose for grasping. While trajectory planning and grasp planning are often tackled separately, how to efficiently integrate the two planning problems remains a challenge. In this work, we present a novel method for joint motion and grasp planning. Our method integrates manipulation trajectory optimization with online grasp synthesis and selection, where we apply online learning techniques to select goal configurations for grasping, and introduce a new grasp synthesis algorithm to generate grasps online. We evaluate our planning approach and demonstrate that our method generates robust and efficient motion plans for grasping in cluttered scenes.

1 Introduction
In order to manipulate objects in cluttered scenes, a robot needs to plan its motion to reach and grasp a target object in the scene. Due to the complexity of this manipulation planning problem, it is usually decomposed into two sub-problems and tackled independently: arm motion planning and grasp planning. While arm motion planning aims at reaching the goal and avoiding obstacles in the scene, grasp planning generates feasible grasps of the target object based on the properties of the object such as 3D shape and material. However, separating motion planning and grasp planning requires some post-processing to connect the two plans, which usually involves heuristics and leads to sub-optimal solutions. How to efficiently integrate motion planning and grasp planning remains a challenge.

According to when grasps are generated during planning, integrated motion and grasp planning methods in the literature can be classified into two categories. Methods in the first category generate a fixed set of grasps offline, and formulate the problem as a grasp selection problem (Dragan, Ratliff, and Srinivasa 2011; Dragan, Gordon, and Srinivasa 2011). Generating grasps offline can take advantages of the state-of-the-art grasp planning algorithms (Miller and Allen 2004). However, a fixed set of grasps limits the number of grasp options during motion planning. Methods in the second category generate grasps online during trajectory motion planning. Online grasp planning can explore more grasp options. However, since the space of possible grasps is large and highly non-convex, approximations are usually made to simplify the grasp planning problem (Vahrenkamp et al. 2010), which limits the quality of the generated grasps.

In this work, to overcome the limitations of existing joint motion and grasp planning methods, we introduce a novel integrated Optimization-based Motion and Grasp Planner (OMG Planner) that combines online grasp synthesis and grasp selection. By leveraging existing grasp planning methods to generate an initial grasp set for online grasp selection, our method guarantees the quality of the selected grasps. By performing online grasp synthesis, our method augments the initial grasp set and provides more grasp candidates during motion planning.

Specifically, we formulate the joint planning problem in an optimization framework, where we optimize for motion trajectories and grasps jointly. To solve the optimization problem, an iterative update algorithm is introduced that alternatively updates trajectories and grasps. In each iteration of the optimization, given a grasp goal configuration, we utilize the CHOMP trajectory optimization method (Ratliff et al. 2009) with a goal set constraint (Dragan, Ratliff, and Srinivasa 2011) to update the trajectory. To update the grasp, we first assume the availability of a set of grasp candidates that can be obtained from the state-of-the-art grasp planning methods. Our online grasp selection algorithm then leverages the interplay between the grasp set and the trajectories to find the optimal goal in the grasp set efficiently during motion planning. In order to augment the grasp set during planning, we also propose a grasp synthesis procedure named Configuration Space Iterative Surface Fitting (C-Space ISF) that refines the grasp configuration quality of the selected grasp. The overall framework leverages the grasp set structure to improve motion and grasp generation.

We conduct qualitative and quantitative evaluations of our approach to demonstrate its advantages. We also perform ab-
lution studies on each component of the algorithm. Overall, our contributions in this work are:

1. We introduce an integrated planner that models joint motion and grasp planning as an optimization problem. The computation of feasible and reachable grasps and the search for collision-free motions are connected.

2. We propose an online learning algorithm and a grasp synthesis approach to select grasps and generate grasps online, which eliminates the need of a perfect grasp set and goal-selection heuristics.

3. We showcase our algorithms for manipulation planning tasks in cluttered scenes. By comparing the proposed approach with the state-of-the-art methods, we demonstrate that our integrated planner improves performance.

2 Related Work

2.1 Motion Planning

Sampling-Based Methods. Sampling-based motion planners such as Rapidly exploring Random Trees (RRTs) (LaValle 1998; Kuffner and LaValle 2000; Karaman and Frazzoli 2011) find trajectories by incrementally building space filling trees through directed sampling. RRTs offer probabilistic completeness. If there exists a solution to the problem, given sufficient time, a feasible trajectory can be found. However, sampling-based algorithms may be difficult to use in real-time applications due to computational challenges and requirements for post-processing steps such as "shortcut" heuristics (Kavraki and Latombe 1998).

Trajectory Optimization. Trajectory optimization starts with a possibly infeasible trajectory and then optimizes the trajectory by minimizing a cost function. CHOMP (Ratcliff et al. 2009) and related methods (Dragan, Ratliff, and Srinivasa 2011) optimize a cost functional using covariant gradient descent, while STOMP (Kalakrishnan et al. 2011) uses stochastic sampling of noisy trajectories to optimize nondifferentiable costs. More recently, TrajOpt (Schulman et al. 2014) solves a sequential quadratic program and performs convex continuous-time collision checking. GPMP2 (Mukadam et al. 2018) formulates the problem as inference on a factor graph and finds maximum a posteriori (MAP) trajectory by solving a nonlinear least squares problem. Trajectory optimization methods are very fast, but can only find locally optimal solutions. Another drawback of trajectory optimization methods is that they often require heavy hand tuning of the parameters and lose the feasibility guarantees. The local nature of previous methods can often be improved by multiple initializations and goals (Schulman et al. 2014; Dragan, Gordon, and Srinivasa 2011). Given a way to generate a diverse set of goals, we believe that the local optimum issue is partly due to the goal selection problem. Our method is based on CHOMP for trajectory optimization. Instead of optimizing trajectories on fixed endpoints, we also consider goal selection and synthesis in our planner. Our trajectory interpolations to goal set have similar flavors as random trajectory samples in STOMP and mean prior trajectory in GPMP2. While (Dragan, Gordon, and Srinivasa 2011) model goal selection problem as a regression on trajectory attributes through offline training, our method is completely online, in a sense that the goal selection strategy is learned during optimization.

2.2 Grasp Planning

Analytic Methods. Analytic methods often require full object information to define grasp qualities such as maximal disturbance resistance, force closure and antipodal grasps (Ferrari and Canny 1992; Nguyen 1988; Chen and Burdick 1993). For instance, (Ciocarlie, Goldfeder, and Allen 2007) strive to reduce the dimensionality of the search space for robust grasp and use eigengrasps in Graspit! simulator to plan with simulated annealing algorithm. Recently, (Fan et al. 2018) propose to find robust grasp pose by iterative surface fitting that optimizes error metrics of normal alignments and distances as a least square problem.

Data-driven Methods. Compared to analytic methods, data-driven approaches (Goldfeder and Allen 2011; Bohg et al. 2013) have gained popularity since they often do not require full information of objects and avoid the difficulty of modeling objective functions. For determining object grasps from images, (Redmon and Angelova 2015) assume the objects are put on a tabletop setting, and a parallel gripper grasps the object from top down. (ten Pas et al. 2017) sample 6DoF grasps and train a neural network to predict a quality score. Recent works also focus on the diversity of grasp synthesis, which can be important for motion planning with multiple goals. For example, (Mousavian, Eppner, and Fox 2019) propose to use a variational autoencoder as a generator to output grasps and use an evaluator network to score the grasp. Compared with these grasp synthesis techniques, our method also plan arm motions to reach the synthesized grasps in cluttered scenes.

2.3 Integrated Motion and Grasp Planning

The advantage of an integrated planner is that a fixed limited set of grasps is no longer required, and motion and grasp planning can impose constraints on each other online. The closest works to ours are integrated motion and grasp planners such as (Vahrenkamp et al. 2010; Fontanals et al. 2014; Hang et al. 2016), which often relies on sampling-based motion and grasp planner. The grasp search is performed while the randomized motion planning algorithm loop is running. The idea is that motion planner tries to connect growing trees to the nodes leading to better grasps while the online grasp sampling techniques synthesizes grasp poses in the trend of tree expansion. The search of valid grasps is limited to reachable poses via forward kinematics. Our approach instead is based on trajectory optimization motion planner which has a special focus on obstacle avoidance and kinematics priors. We also explicitly model a goal selection process and refine feasible grasp in configuration space.

2.4 Online Learning

Compared with statistical learning in which data is assumed to come from a distribution and the goal is to minimize an excess risk, online learning instead considers a setting where the loss function $l_t$ arrives online at time $t$ and the
model needs to dynamically adapt to newly arriving patterns (Bubeck and others 2015). In motion planning with a goal set, one often neglects the goal selection process. However, an optimal goal selection procedure can be modelled as an optimization problem in which our learner needs to choose the goal online and adapt to some measured costs. In regret-based framework, the learner can model a distribution that minimizes the regret on the loss sequence.

A specific example is the Metrical Task Systems (MTS) problem (Borodin, Linial, and Saks 1992). In MTS, there is a service cost that depends on the environment whose behavior is hard to predict. There is also a movement cost that penalizes large distribution shifts. This idea is analogous to the goal selection problem because we do not have an accurate estimate of our objective costs and we can suffer from switching goals too often. To the best of our knowledge, our framework is the first one to apply online learning techniques to the integration of motion and grasp planning.

3 Methodology

The key idea of our method is to simultaneously optimize the arm motion trajectory and the end-effector configuration for grasping. We formulate the problem as a constrained trajectory optimization problem, where we require the motion trajectory to avoid obstacles and to be smooth. Meanwhile, the last configuration of the trajectory should afford grasping. To achieve this, our method synthesizes and selects grasps online during trajectory optimization. Fig. 2 presents an overview of our optimization-based motion and grasp planner.

Formally, we define a trajectory \( \xi : [0, 1] \rightarrow \mathbb{Q} \) to be a function that maps time \([0, 1]\) to the robot configuration space \( \mathbb{Q} \), where \( \mathbb{Q} \subseteq \mathbb{R}^d \) and \( d \) is the degree of freedom of the robot manipulator. Given a scene of multiple objects, we define the feasible grasp set of a target object as \( \mathcal{G} \subseteq \mathbb{Q} \).

Then we solve the following optimization problem to find the optimal trajectory of the robot manipulator for grasping:

\[
\begin{align*}
\xi^* &= \arg\min_{\xi} f_{\text{motion}}(\xi) \\
&\text{s.t. } \xi(1) \in \mathcal{G},
\end{align*}
\]

(1)

where \( f_{\text{motion}} \) is the objective function for the trajectory, \( \xi(1) \) indicates the last configuration of the trajectory. The constraint in Eq. (1) requires the last configuration at time 1 to be a feasible grasping configuration.

3.1 Trajectory Objective Functional

Similar to CHOMP (Ratliff et al. 2009), we model the objective functional in Eq. (1) as

\[
f_{\text{motion}}(\xi) = f_{\text{obstacle}}(\xi) + \lambda f_{\text{prior}}(\xi),
\]

(2)

where the obstacle term \( f_{\text{obstacle}} \) bends the trajectory away from obstacles by penalizing parts of the robot that are close to or already in collision, the prior term \( f_{\text{prior}} \) measures the dynamical quantities across the trajectory, such as velocities and accelerations, and \( \lambda \) is a weight factor to balance the two terms. In this work, the prior term is defined as the integral over squared velocity norms

\[
f_{\text{prior}}(\xi) = \frac{1}{2} \int_0^1 \|\xi'(t)\| \, dt,
\]

(3)

where \( \xi'(t) \) indicates the velocity of the trajectory at time \( t \).

To define the obstacle term, let \( \mathcal{B} \subseteq \mathbb{R}^3 \) be a set of body points on the robot and \( x(q, u) : \mathbb{Q} \times \mathcal{B} \rightarrow \mathbb{R}^3 \) the forward kinematics mapping from a body point \( u \in \mathcal{B} \) with configuration \( q \in \mathbb{Q} \) to the workspace. Furthermore, let \( c_{\text{obstacle}} : \mathbb{R}^3 \rightarrow \mathbb{R} \) be a workspace cost function that penalizes points in the workspace inside and around the obstacles using Signed Distance Field (SDF). Since we want to drive
the body points away from collision, the obstacle term in Eq (1) is an integral that collects the cost of workspace body points on the robot as it sweeps along the trajectory:

$$f_{\text{obstacle}}(\xi) = \int_0^1 \int_B c_{\text{obstacle}}(x(\xi(t)), u) \left\| \frac{d}{dt}(x(\xi(t)), u) \right\| dtdt.$$  

(4)

Please refer to (Ratliff et al. 2009) for the definition of the obstacle cost $c_{\text{obstacle}}$ and the derivation of gradients of the two terms $f_{\text{prior}}$ and $f_{\text{obstacle}}$.

### 3.2 Iterative Update Rule

In practice, we use a discretization of the trajectory function over time: $\xi \approx (q_1^T, q_2^T, ..., q_n^T)^T \in \mathbb{R}^{n \times d}$ for resolution $n$. Under this parametrization, we can write the prior term as:

$$f_{\text{prior}}(\xi) = \frac{1}{2} K(\xi + e)^T a + c,$$  

(5)

where $K$ is a finite differencing matrix, $e$ is a vector that accounts for the contributions of the configurations that remain constant in the trajectory, i.e., the start configuration and the goal configuration, and $A = K^T K$ is the dynamic matrix. The covariant idea of CHOMP comes from the inverse of the dynamic matrix. $A^{-1}$ acts as a smoothing operator that propagates the Euclidean gradient of the objective along the trajectory and seeks to make small changes in the average acceleration. Given this parametrization, CHOMP is a variant of gradient descent that minimizes the linear approximation of $f_{\text{motion}}$ about $\xi$ within an ellipsoid trust region. The distance metric that shapes this ellipsoid is defined by $A$ and we obtain the update rule in gradient descent:

$$\xi_{i+1} = \xi_i - \eta A^{-1} v_i,$$  

(6)

where $v_i = \nabla f_{\text{motion}}(\xi_i)$ is the discretized functional gradient and $\eta$ is the step size.

### 3.3 Goal Set Constraint

The end-point of $\xi_i$ can be varied during planning for a set of feasible goal configurations $\mathcal{G}$. Following (Dragan, Ratliff, and Srinivasa 2011), we first change the $A$ matrix so that the update rule can alter the trajectory end-point.

Given a general constraints $h(\xi) = 0$, we can linearize $h$ around $\xi_i$: $h(\xi) \approx h(\xi_i) + \frac{\partial h(\xi_i)}{\partial \xi}(\xi - \xi_i) = C(\xi - \xi_i) + b$ where $C$ is the Jacobian of the $h$ evaluated at $\xi_i$ and $b = h(\xi_i)$. We use Lagrangian to solve the linearized constraint and yield the projected gradient update rule:

$$\xi_{i+1} = \xi_i - \eta A^{-1} v_i + \frac{1}{\eta} A^{-1} C^T(CA^{-1}C^T)^{-1}CA^{-1}v_i - A^{-1} C^T(CA^{-1}C^T)^{-1}b.$$  

(7)

Assume our target end configuration is $g$ at the $i$th iteration. We simply model the goal set constraint as $h(\xi_i) = \xi_i(1) - g = 0$ and denote the update rule in Eq. (7) by CHOMP-Proj($\xi, g$). With this update rule, we denote $f_g(\xi_i)$ as the objective cost if we fix $g$ from iteration $i$. As we move the end-point, CHOMP smoothly updates the remaining trajectory by curving around the obstacles.

### 3.4 Online Learning for Grasp Selection

While (Dragan, Ratliff, and Srinivasa 2011) address the problem of CHOMP planning with a goal set constraint, modelling goal selection as projection can often be simplistic. (Dragan, Gordon, and Srinivasa 2011) propose a few trajectory attributes such as goal radius, elbow room, distance, etc., for their offline regression formulation. Similarly, the optimality of a goal should be induced by the objective function and thus related to the success of motion planning. Given that we can optimize $f_g(\xi_i)$ with a target goal, we want to choose the goal that maximizes our motion generation success under an online learning scheme. An online optimization framework is suitable because every iteration, we can model an estimate of objective costs of the goal set as our suffered losses, and our strategy needs to dynamically predict the best goal for next planning iteration. Moreover, a probability distribution can be used to model our online goal selection process. In projected CHOMP, we suffer from switching to goals that can heavily perturb our current trajectory into collision. However, to allow for principled behavior that selects the next goal $g_{i+1}$, we need a strategy that balances exploitation and exploration of the costs. Let the initial distribution $p_0$ an uniform distribution on a discrete goal set $\mathcal{G}$. At the $i$th iteration, denote the goal distribution by $p_i \in \mathbb{P}$. We use online learning to solve for $p_{i+1}$. To apply optimization for goal selection, we need to define the cost of each goal in the goal set, denoted by $c_i$. Ideally, we want $c_i(g) = f_g(\xi_i)$, then the objective cost directly tells us which target goal to use. However, since the objective costs are not available without running CHOMP, we can only generate approximations to our objective costs $f(\xi_i^g) \approx f_g(\xi_i)$, by estimating the potential optimized trajectory $\hat{\xi}_i^g$ if the trajectory $\xi_i$ at time $i$ ends at $g \in \mathcal{G}$.

We propose that to compute $f(\xi_i^g)$, it is more natural to look at the tail of trajectory denoted by $\xi_i^g$ starting at $t_i = \frac{1}{\eta}$ for simplicity and $N$ is our optimization horizon. To motivate this, recall that in each CHOMP-Proj step, we hard project onto the goal constraint and propagate the changes smoothly. Therefore, as $i$ approaches $N$, our changes to the trajectory impacts more on the tail than the head of the trajectory. In other words, our estimated costs would be more accurate if we use the tail $\xi_i^g$ instead of $\xi_i^g$. Moreover, the goal cost vector needs to converge for stability of the algorithm and it should also be dynamic from learning perspective. Another way to think about this is that we are “simulating” the execution of the trajectory during planning. As shown in Fig. 3, since the prior of CHOMP is the minimum length trajectory, an intuitive trajectory tail $\xi_i^g$ can be defined as linear interpolation between $\xi_i(t_i)$ and goal $g$. The associated cost for next iteration is defined as $c_{i+1}(g) = f(\xi_i^g)$.

Given our estimated objective costs $c_{i+1} = \{f(\xi_i^g) \mid g \in \mathcal{G}\}$ and previous costs $c_1, ..., c_i$ for goal set $\mathcal{G}$, we can use online learning algorithms to update the goal distribution. Our expected loss at $(i + 1)$th iteration under this distribution is $(c_{i+1}, p_{i+1})$ and $g_{i+1}$ is simply the maximum likelihood goal of the our chosen distribution $p_{i+1}$. Overall, on-
line learning for goal selection aims to minimize the regret

\[ R_N = \sum_{i=1}^{N} \langle c_i, p_i \rangle - \min_{g \in G} \sum_{i=1}^{N} c_i(g), \]  

(8)

which compares the loss of our strategy \( p_i \) up to time \( N \) with the best cost of any fixed goal in hindsight. In the following, a few commonly used online learning methods are introduced. These methods can be used to solve \( p_{i+1} \) at the \( i \)th iteration.

Follow the Cheapest (FTC): At every time \( i \), FTC greedily moves the distribution \( p_{i+1} \) entirely to the point that is currently incurring the minimum \( c_{i+1}(g) \). Observe that the original goal-set CHOMP turns out to be a FTC algorithm with the cost vector only involving the distance between the current trajectory and the goals.

Follow the Leader (FTL): At every time \( i \), FTL computes the point \( g \in G \) that has incurred the lowest total cost \( \sum_{i=1}^{i+1} c_i(g) \) and then makes the distribution \( p_{i+1} \) concentrated entirely on \( g \).

Exponential Weights (EXP): The algorithm has a parameter \( \eta_{ol} \) called the learning rate. EXP begins with an uniform distribution over actions: \( p_1(g) = \frac{1}{|G|}, \forall g \in G \). At time \( i \), we first multiply the previous probabilities according to the corresponding loss: \( p_{i+1}(g) = \exp(\eta_{ol} c_{i+1}(g))p_i(g), \forall g \in G \). Then we renormalize the vector to make it a probability distribution: \( p_{i+1}(g) = p_{i+1}(g)/\sum_{g \in G} p_{i+1}(g) \).

Mirror Descent (MD): We refer the readers to (Bubeck and others 2015) for this form of regularized optimization. In a high level, MD takes gradient steps in the dual space. Define a probability simplex \( \Delta \) and a learning rate \( \eta_{md} \). For goal selection, we want the strategy to prefer probability \( \hat{p}_i \) at iteration \( i \) with the best cost of any fixed goal in hindsight. In the following, we associate the Bregman divergence with entropy, which is strongly convex, denoted by \( \psi \). At time step \( i \), we obtain the update rule: \( p_{i+1} = \arg\min_{p \in \Delta} \eta_{md}(c_{i+1}, p) + D_{\psi}(p||p_i) \), where

\[ D_{\psi}(p||p_i) = \sum_{g \in G} (p(g) \log \frac{p(g)}{p_i(g)} + (p_i(g) - p(g))). \]

The argmin of the update rule can be solved by Lagrangian.

### 3.5 Online Grasp Synthesis

So far we have assumed that the goal set is given and fixed, which can often be a strict requirement for real world manipulation settings. Moreover, a fixed set of grasps has several disadvantages, such as limitations in the number of grasp possibilities. Even with a set of perfect grasps in workspace, it is unclear that solving inverse kinematics and filtering by collision would give a sufficient set required for a feasible motion plan. A fixed set also has low adaptability to the environment since it fails to explore new grasps.

Starting from an initial fixed goal set \( G \), we propose to synthesize more candidate grasp configurations online. Suppose after the online learning process for goal selection, we select the configuration end goal \( g_i \) at iteration \( i \). For simplicity, we drop the iteration index \( i \) in the followings. Our proposed grasp synthesis algorithm aims to improve the grasp affordance by minimizing the objective function:

\[ f_{\text{grasp}}(g) = f_{\text{std}}(g) + \gamma f_{\text{collision}}(g), \]

(9)

where \( f_{\text{std}}(g) \) measures the grasp quality using Iterative Surface Fitting (ISF). \( f_{\text{collision}}(g) \) measures collision in the scene, and \( \gamma \) is a constant to balance the two terms. We use \( f_{\text{std}}(g) \) to refine the end configuration \( g \) such that the end effector transformation \( T \in SE(3) \) makes the surface of the robot gripper \( S \) match against the surface of the target object \( O \). Compared with (Fan et al. 2018) that filters collided grasps and models the objective as a least square problem, we instead softly penalize hand and arm collisions and use gradient descent to optimize the objective. This enables efficient and stable optimization over the configuration space.

Specifically, suppose a target object \( O \) and a hand model \( S \). We generalize the forward kinematics mapping \( x(g, u) \) of a body position \( u \) in Sec. 3.1 to include directions: \( x(g, u^n) \) maps a body direction vector \( u^n \) and a robot configuration \( g \) to a direction vector in workspace. Observe that the transformation \( T \) in workspace is the hand origin transformed by \( g \). We can write all workspace costs on \( T \) as configuration space costs on \( g \). We first define a set of hand contact points \( \{h_j\} \) and a set of contact normals \( \{n_j\} \) on the robot hand and fingers, indexed by \( j \) (see Fig. 4). We denote \( u_h = \{x(g, h_j)\} \) and \( u^n_h = \{x(g, h^n_j)\} \) on the two sets in the workspace transformed by forward kinematics. We then search for object \( O \)’s nearest neighbor points \( u_o \) and normals \( u^n_o \) associated with \( u_h, u^n_h \). For \( m \) pairs of points and normals, we define the point matching loss as

\[ f_{\text{pml}}(g) = \sum_{j=1}^{m} (u_{h,j} - u_{o,j})^2, \]

(10)

and the normal alignment error as

\[ f_{\text{align}}(g) = \sum_{j=1}^{m} ((u^n_{h,j} - u^n_{o,j} + 1)^2. \]

(11)

The alignment loss aligns the surface normals, and the point
this iterative update rule for $N$ times for some horizon $N$. A pre-termination criteria can be set with the threshold on grasp quality and collision-free requirement. We note again one of the primary benefits of online grasp synthesis and selection in motion planning is that we relax the requirement of a pre-computed grasp set and eliminate the need to brute force planning for each goal. Our joint planning algorithm is shown in Algorithm 2. CHOMP-Proj by design provides a way to smoothly update the goal and avoid obstacles. ONLINE-LEARNING considers all grasps in the goal set and updates our goal distribution for the next iteration. The C-Space ISF only applies on selected goal so it is more efficient than running the refinement steps offline for all grasps.

### 4 Experiments

#### 4.1 Implementation Details

CHOMP. We set the discretization number $n = 30$ and smooth weight $\lambda = 0.1$. The learning rate is $\eta_{\text{motion}} = 0.01$. The obstacle padding is 0.2m within which the cost increases quadratically (Katliif et al. 2009). A collision-free trajectory is assumed to have a minimum distance of 5cm from obstacles. A difference from the original CHOMP is that instead of modeling the arm with swept spheres that upper bound the distance to the center of each arm component, we uniformly sample body points on the surface of the robot models to compute the obstacle costs (see Fig. 4). Observe that this modification would make the integral over $B$ more general and more precise. We sample 15 points from each body part in our implementation. Lastly, since a grasping task requires collision-free hand configuration, we include sampled points from hand and fingers.

Online Learning. The EXP and MD algorithms take the learning rate over experts of $\eta_{\text{ol}} = 2^k \log(N)$ and $k$ is selected from $\{-11, 7, 0, 1\}$.

C-Space ISF. We set the weight for normal penalty $\alpha = 0.01$, and the arm collision coefficient $\beta = 0.01$. The constant for collision weight is $\gamma = 5$. The step size is fixed to be $\eta_{\text{collision}} = 0.01$. The nearest neighbors are found with KDTree on 1000 sampled points for each object, where repeated neighbors are filtered out to increase robustness.

#### 4.2 Planning Results

We investigate the following questions: 1) Does online learning help select goals in the grasp set to improve motion generation? 2) Can C-Space ISF refine grasp success? 3) Is there
### Table 1: Online learning algorithm evaluation

| Alg. | Baseline | Proj | FTC | FTL | EXP | MD |
|------|----------|------|-----|-----|-----|----|
| Success | 66% | 72% | 93% | 92% | 95% | 97% |
| Smooth | 9.67 | **8.41** | 11.30 | 11.75 | 12.47 | 12.05 |
| Collision | 1.86 | 1.68 | 1.56 | 1.63 | 1.54 | **1.50** |
| Time | **0.53** | 0.64 | 1.12 | 1.35 | 1.31 | 1.38 |

Table 1: Online learning algorithm evaluation. Smoothness cost and collision cost are the discrete versions of the CHOMP objectives. Planning success is defined as collision-free and a threshold on the smoothness cost (path length) of the optimized trajectory. Planning time is measured in seconds. All quantities are averaged over 100 scenes.

### Table 2: Grasp costs before and after refinement on six YCB Objects.

| Object Name | S | S (Ours) | G | G (Ours) | A | A (Ours) |
|-------------|---|---------|---|---------|---|---------|
| Cracker Box | 2.47 | **2.32** | 4.27 | 3.31 | 6.58 | 4.36 |
| Mug | 5.11 | **3.93** | 7.24 | 5.55 | 9.05 | 5.18 |
| Tomato Soup Can | 2.38 | **2.27** | 3.15 | 2.83 | 7.06 | 6.27 |
| Mustard Bottle | 4.40 | **3.19** | 4.91 | 3.79 | 8.15 | 5.73 |
| Potted Meat Can | 2.96 | **1.95** | 3.08 | 2.54 | 7.47 | 4.91 |
| Pitcher Base | 4.41 | **2.97** | 5.68 | 3.76 | 8.69 | 6.14 |
| Mean | **3.62** | 2.77 | 4.72 | 3.63 | 7.83 | 5.43 |

Table 2: Grasp costs before and after refinement on six YCB Objects. “S” denotes the simulated grasps from (Eppner, Mousavian, and Fox 2019), “G” denotes the grasps from Graspit! (Miller and Allen 2004), and “A” denotes our sampled grasps by approaching from random directions.

We test the grasp refinement performance on YCB Objects by running the C-Space ISF algorithm for 30 iterations. The initial collision-free configurations in our set $G$ are solved by IK with random seeds. Table 2 shows the averaged costs over 60 different runs with various initializations. The optimization procedure minimizes the surface fitting error while avoiding the in-collision configurations. We evaluate three different sets of initial grasps as shown in Fig. 5. The first set “Simulated” is sampled from a physics-based simulator (Eppner, Mousavian, and Fox 2019). The second set is sampled from the Graspit! planner (Ciocarlie, Goldfeder, and Allen 2007), and the third set “Approach” is collision free grasp naively sampled by random approach directions.

Overall, there is a consistent performance improvement by running optimization for different initial grasps. The “Simulated” set with the highest quality is expected to have the minimum costs. But it is time-consuming to generate these grasps. In practice, without the availability of grasp databases such as “Simulated” and Graspit!, we can synthesize grasps as in “Approach” that can be efficiently generated. Note that these “Approach” grasps after optimization still have reasonable quality. Additionally, for some fine grasps such as those clustered around the bottle neck, it would be hard to run our refinement. Also, the contact and
Table 3: Comparison on planning methods for joint motion and grasp planning. CHOMP (Ratliff et al. 2009), CHOMP-C (Dragan, Ratliff, and Srinivasa 2011), TrajOpt (Schulman et al. 2014), RRT-Connect (Kuffner and LaValle 2000)

| Algorithm   | CHOMP | CHOMP-C | OMPL-TrajOpt | OMPL-RRTConnect | OMG-Grasp | OMG-OL | OMG-Full |
|-------------|-------|---------|--------------|-----------------|-----------|--------|---------|
| Success     | 60%   | 67%     | 63%          | 64%             | 63%       | 88%    | 90%     |
| Smooth      | 15.61 | 15.18   | 15.31        | 16.15           | 14.93     | 11.89  | 11.76   |
| Collision   | 2.43  | 2.26    | 2.35         | 2.52            | 2.38      | 1.71   | 1.64    |
| Grasp       | 3.69  | 3.93    | 3.81         | 3.64            | 2.98      | 3.45   | 3.11    |
| Time        | 0.97  | 1.04    | **0.23**     | 0.41            | 1.18      | 1.36   | 1.42    |

Figure 7: Planning scene examples. Red trajectories are from CHOMP, and green trajectories are from our OMG-Planner.

Planner Performance. We compare the performance of our OMG planner to the state-of-the-art motion planners which have public implementations available. We omit integrated planners (Hang et al. 2016) (Vahrenkamp et al. 2010) (Fontanals et al. 2014) due to the lack of available code. While grasp planning is not covered in these motion planners, we adopt the common routine used in practice to generate manipulation trajectory: (1) rank the goals in some grasp databases based on heuristics such as our objective estimates. (2) planning through the goal set until it finds a solution. We perform planning experiments for 150 scenes with 50 initial grasps from the high-quality “Simulated” goal set for each scene. The results are presented in Table 3. The CHOMP planner (Ratliff et al. 2009) with a fixed goal is denoted as “CHOMP”, and its goal set variant is denoted as “CHOMP-C” (Dragan, Ratliff, and Srinivasa 2011). We also compare our algorithm with two planners in OMPL (Sucan, Moll, and Kavraki 2012) implemented in Moveit! (Chitta, Sucan, and Cousins 2012): “TrajOpt” (Schulman et al. 2014) and “RRT-Connect” (Kuffner and LaValle 2000).

We also investigate the contributions of different components in our planner. The “OMG-OL” planner only adds online learning in goal selection. “OMG-Grasp” only adds grasp synthesis. “OMG-Full” is our full algorithm with joint motion and grasp planning. Comparing “OMG-OL” with other planners, we can see the improvement of our goal selection process. While these methods admit a brute force search for all goals to find a global minimum, it wastes time in many scenarios. With a strategy to terminate at the first feasible trajectory, these methods can generate suboptimal results. On the other hand, with the option to select goals in the goal set, our trajectory optimization has fundamentally more solutions and more likely to yield optimal results. Comparing “OMG-Grasp” with “CHOMP”, we see an improvement of grasp quality. Since “OMG-Grasp” consistently optimizes one grasp, it reaches the minimum grasp cost. Finally, “OMG-Full” not only selects the best goal but also refines the grasp quality and smooths the trajectory. Some qualitative planning results are shown in Fig. 7.

5 Conclusion

We have presented a joint motion and grasp planning approach to generate grasping trajectories. Our OMG-Planner does not require a perfect goal configuration, because the candidate goals are determined during the planning process. The goal set is explicitly modelled as a distribution and the best goal is selected online. Moreover, our method synthesizes new grasps online to augment the goal set. Our experiments demonstrate that the OMG-Planner can plan grasping trajectories for cluttered scenes.

Similar to other optimization-based planning methods, our approach is by nature local, and it has no guarantee to find feasible solutions. However, compared to sequential approaches, where a grasp is chosen from a set of pre-computed grasps, our method considers the whole set of feasible goal configurations in an online scheme, which is more efficient and principled. While our work is based on CHOMP, we believe the online goal selection procedure can also benefit other planning methods. Moreover, we make few assumptions about the structure of the grasping tasks, which implies that our framework could easily be extended to tasks that has a set of goal region. Finally, humans intuitively do not have a determined feasible grasp in mind before they attempt reaching, which allows re-planning in dynamic situations. Our formulation indeed has a close relationship with dynamic planning and control, so we believe integrating reactive control is a feasible future direction.
References

Bohg, J.; Morales, A.; Asfour, T.; and Kragic, D. 2013. Data-driven grasp synthesis survey. IEEE Transactions on Robotics 30(2):289–309.

Borodin, A.; Linial, N.; and Saks, M. E. 1992. An optimal on-line algorithm for metrical task system. Journal of the ACM (JACM) 39(4):745–763.

Bubeck, S., et al. 2015. Convex optimization: Algorithms and complexity. Foundations and Trends® in Machine Learning 9(3-4):231–357.

Calli, B.; Walsman, A.; Singh, A.; Srinivasa, S.; Abbeel, P.; and Dollar, A. M. 2015. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. arXiv preprint arXiv:1502.03143.

Chen, I.-M., and Burdick, J. W. 1993. Finding antipodal space: A unified framework for grasp planning and in-hand manipulation planning with goal sets using constrained trajectory optimization. In 2011 IEEE International Conference on Robotics and Automation (ICRA), 995–1001.

Chitta, S.; Sucan, I.; and Cousins, S. 2012. Moveit![ros tools]. IEEE Robotics & Automation Magazine 19(1):18–19.

Ciocarlie, M.; Goldfeder, C.; and Allen, P. 2007. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In Robotics: Science and Systems Manipulation Workshop-Sensing and Adapting to the Real World (RSS).

Coumans, E., et al. 2013. Bullet physics library. Open source: bulletphysics. org 15(49):5.

Dragan, A.; Gordon, G. J.; and Srinivasa, S. 2011. Learning from experience in manipulation planning: Setting the right goals.

Dragan, A. D.; Ratliff, N. D.; and Srinivasa, S. S. 2011. Manipulation planning with goal sets using constrained trajectory optimization. In 2011 IEEE International Conference on Robotics and Automation (ICRA), 4582–4588.

Eppner, C.; Mousavian, A.; and Fox, D. 2019. A billion ways to grasp: an evaluation of grasp sampling schemes on a dense, physics-based grasp data set. in Proceedings of the International Symposium on RoboticsResearch (ISR).

Fan, Y.; Lin, H.-C.; Tang, T.; and Tomizuka, M. 2018. Grasp planning for customized grippers by iterative surface fitting. In 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), 28–34.

Ferrari, C., and Canny, J. F. 1992. Planning optimal grasps. In IEEE International Conference on Robotics and Automation (ICRA), volume 3, 2290–2295.

Fontanals, J.; Dang-Vu, B.-A.; Porges, O.; Rosell, J.; and Roa, M. A. 2014. Integrated grasp and motion planning using independent contact regions. In 2014 IEEE-RAS International Conference on Humanoid Robots, 887–893.

Goldfeder, C., and Allen, P. K. 2011. Data-driven grasping. Autonomous Robots 31(1):1–20.

Hang, K.; Li, M.; Stork, J. A.; Bekiroglu, Y.; Pokorny, F. T.; Billard, A.; and Kragic, D. 2016. Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaptation. IEEE Transactions on robotics 32(4):960–972.

Kalakrishnan, M.; Chitta, S.; Theodorou, E.; Pastor, P.; and Schaal, S. 2011. Stomp: Stochastic trajectory optimization for motion planning. In 2011 IEEE International Conference on Robotics and Automation (ICRA), 4569–4574.

Karaman, S., and Frazzoli, E. 2011. Sampling-based algorithms for optimal motion planning. The International Journal of Robotics Research (IJRR) 30(7):846–894.

Kavraki, L. E., and Latombe, J.-C. 1998. Practical motion planning in robotics: Current approaches and future directions, chapter probabilistic roadmaps for robot path planning.

Kuffner, J. J., and LaValle, S. M. 2000. Rrt-connect: An efficient approach to single-query path planning. In Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation (ICRA), volume 2, 995–1000.

LaValle, S. M. 1998. Rapidly-exploring random trees: A new tool for path planning.

Miller, A. T., and Allen, P. K. 2004. Graspit! a versatile simulator for robotic grasping.

Mousavian, A.; Eppner, C.; and Fox, D. 2019. 6-dof grasp-net: Variational grasp generation for object manipulation. arXiv preprint arXiv:1905.10520.

Mukadam, M.; Dong, J.; Yan, X.; Dellaert, F.; and Boots, B. 2018. Continuous-time gaussian process motion planning via probabilistic inference. The International Journal of Robotics Research (IJRR) 37(11):1319–1340.

Nguyen, V.-D. 1988. Constructing force-closure grasps. The International Journal of Robotics Research 7(3):3–16.

Ratliff, N.; Zucker, M.; Bagnell, J. A.; and Srinivasa, S. 2009. Chomp: Gradient optimization techniques for efficient motion planning. IEEE International Conference on Robotics and Automation (ICRA).

Redmon, J., and Angelova, A. 2015. Real-time grasp detection using convolutional neural networks. In 2015 IEEE International Conference on Robotics and Automation (ICRA), 1316–1322.

Schulman, J.; Duan, Y.; Ho, J.; Lee, A.; Awwal, I.; Bradlow, H.; Pan, J.; Patil, S.; Goldberg, K.; and Abbeel, P. 2014. Motion planning with sequential convex optimization and convex collision checking. The International Journal of Robotics Research (IJRR) 33(9):1251–1270.

Sucan, I. A.; Moll, M.; and Kavraki, L. E. 2012. The open motion planning library. IEEE Robotics & Automation Magazine 19(4):72–82.

ten Pas, A.; Gualtieri, M.; Saenko, K.; and Platt, R. 2017. Grasp pose detection in point clouds. The International Journal of Robotics Research (IJRR) 36(13-14):1455–1473.

Vahrenkamp, N.; Do, M.; Asfour, T.; and Dillmann, R. 2010. Integrated grasp and motion planning. In 2010 IEEE International Conference on Robotics and Automation (ICRA), 2883–2888.