Accepted for publication in IEEE International Conference on Robotics and Automation (ICRA) June 2021

Robust Optimization-based Motion Planning for high-DOF Robots under Sensing Uncertainty

Carlos Quintero-Peña[1], Anastasios Kyrillidis[1] and Lydia E. Kavraki[1]

***Abstract*— Motion planning for high degree-of-freedom (DOF) robots is challenging, especially when acting in complex environments under sensing uncertainty. While there is significant work on how to plan under state uncertainty for low-DOF robots, existing methods cannot be easily translated into the high-DOF case, due to the complex geometry of the robot's body and its environment. In this paper, we present a method that enhances optimization-based motion planners to produce robust trajectories for high-DOF robots for convex obstacles. Our approach introduces robustness into planners that are based on sequential convex programming: We reformulate each convex subproblem as a robust optimization problem that "protects" the solution against deviations due to sensing uncertainty. The parameters of the robust problem are estimated by sampling from the distribution of noisy obstacles, and performing a first-order approximation of the signed distance function. The original merit function is updated to account for the new costs of the robust formulation at every step. The effectiveness of our approach is demonstrated on two simulated experiments that involve a full body square robot, that moves in randomly generated scenes, and a 7-DOF Fetch robot, performing tabletop operations. The results show nearly zero probability of collision for a reasonable range of the noise parameters for Gaussian and Uniform uncertainty.**

I. INTRODUCTION

As the cost of robotic platforms reduces and technology integration improves, more often than not, high degree-of-freedom (DOF) robots will make their way into non-structured, highly uncertain environments. Sampling-based [1]–[3] and optimization-based [4]–[7] planners provide efficient planning solutions for such high-dimensional systems in the presence of obstacles, but most such planners assume perfect information about the state of the robot and its environment. For motion planning, this is particularly important since uncertainty-unaware planners will likely generate unsafe motions that do not meet task specifications.

There are methodologies that incorporate uncertainty into motion planners, both sampling-based [8]–[12] and optimization-based [13]–[17]. The majority of approaches is designed only for low-DOF robots, where the robot's geometry is greatly simplified (e.g., to a point robot). Typically, Gaussian uncertainty in the state of the robot is considered.

Yet, many robotic applications involve high-DOF robots, where the uncertainty due to perception dominates that of the robots' state[2]. Take as an example a home-assistant robot faced with the task of removing cutlery and tableware from a dining table: During execution, its actions do not require complex dynamics (e.g., fast motions or lifting heavy objects), but the complex geometry of both the robot and the objects needs to be considered to avoid colliding with (or even breaking) objects on the table. *That is, there are real scenarios where both the robot and the environment are made of complex geometry, which further increase the chances of potential collisions, if uncertainty is not properly taken into account.* Due to the inherent complexity of the robot geometry, the techniques designed for low-DOF robots, more often than not, do not transfer to the high-DOF regime [15].

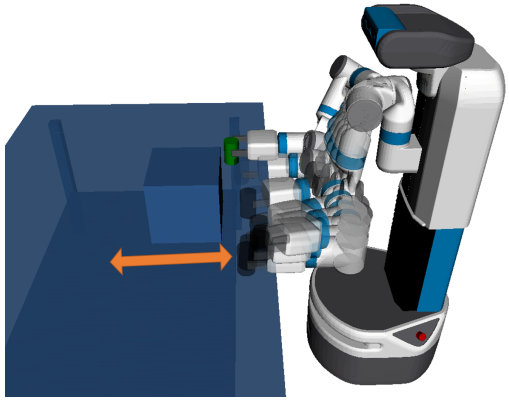

Fig. 1: Example of one trajectory generated by our approach in the *Fetch tabletop* task. The robot avoids colliding with the uncertain block. The uncertainty acts in the direction shown by the arrow.

A successful approach for robustness has been that of producing trajectories that meet certain bounds on the probability of collision, by using chance constraints [13]. This is efficiently achieved by exploiting properties of certain types of noise and geometry to transform chance constraints into deterministic ones that make the problem tractable. Optimization-based methods solve convexified forms of these constraints [13], [15]–[17], while sampling-based methods only consider samples that meet these constraints [9], [18]. *In both cases, the algorithm design is tied to the assumed noise model, making the resulting algorithms impractical to other noise models.*

We propose a method that produces robust trajectories for high-DOF robots, interacting in complex environments, when sensing uncertainty dominates. Our method is not attached to a specific noise model; it only assumes that uncertainty in the problem parameters is bounded. Instead of explicitly deriving

[1]Dept. of Computer Science, Rice University, Houston, TX 77005, USA {carlosq, anastasios, kavraki}@rice.edu. Work on this paper has been supported in part by NSF RI 2008720 and Rice University Funds

[2]Motion happens at low torques and velocities and therefore any uncertainty in the robot actuators can be corrected during execution.

an estimation to the overall risk of collision, we consider how sensing uncertainty is propagated into each convex subproblem of a planner based on sequential convex programming (SCP). Our method finds robust solutions to each subproblem by introducing artificial variables and constraints that protect the solution against *worst-case* deviations. To achieve this, we leverage results from Robust Optimization (RO) [19], [20] that allow each convex subproblem to be solved considering set-based uncertainty in the problem parameters. Under realistic assumptions, our formulation enhances SCP-based planners to produce trajectories with low probability of collision without assuming specific noise models. We test our method in motion planning problems and environments with complex geometry and for different types of noise models.

II. PRELIMINARIES

We consider planning for a high-DOF robot in an environment with convex obstacles and uncertainty in its perception system. We assume that there is no uncertainty in the robot's state, i.e., the robot moves quasi-statically in the environment.

A. Trajectory Optimization through SCP

Optimization-based planners compute a trajectory by solving the following optimization problem:

$$\begin{aligned} \underset{q}{\text{minimize}} \quad & f(q) \\ \text{subject to} \quad & g_i(q) \leq 0,\ i = 1, \dots, m, \\ & h_i(q) = 0,\ i = 1, \dots, n. \end{aligned}$$

Here, q is a vector of stacked waypoints q_t that represent the discretized trajectory ($t = 1, \dots, T$); f usually encourages smooth trajectories; g_i are inequality constraints that can be used for joint limits and collision avoidance; and, h_i can be used to enforce end-effector pose or to meet dynamics.

SCP has been successfully used to solve the non-convex motion planning problem [6], [7], [21], [22]. It works by optimizing a sequence of local convex approximations of the non-convex problem, as the one above. For our discussion, we will focus on TrajOpt [6]. Collision-free trajectories can be computed by enforcing the signed distance between each obstacle and the volume swept by the robot, to be greater than certain safety distance.

Let us assume that the robot body is made of L convex links and the environment has K convex obstacles. Per iteration, TrajOpt solves the following convex subproblem:[3]

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & \widehat{f}(s) && \text{(1a)} \\ \text{subject to} \quad & a_{tkl}^\top s_t + b_{tkl} \leq 0,\ \forall t, k, l && \text{(1b)} \\ & W(s_t + q_t^0) - w \leq 0\ ,\ t = 1, \dots, T && \text{(1c)} \\ & \|s\| \leq \Delta. && \text{(1d)} \end{aligned}$$

Here, $s_t = q_t - q_t^0$ is the update that should be applied to the t-th waypoint of the current trajectory $\{q_1^0, \dots, q_T^0\}$. Constraints (1b) correspond to the convexified collision avoidance achieved through a first order approximation of the signed distance [6]: $a_{tkl}^\top = -\widehat{n}_{tkl}^\top J_{tkl}$, $b_{tkl} = d_{safe} - \text{sd}_{tkl}$. Here, $\widehat{n}_{tkl}$ is the normal vector between obstacle k and link l, J_{tkl} is the robot's Jacobian at contact point p_l, and sd_{tkl} is the signed distance between link l and obstacle k at the t-th waypoint. Constraints (1c) represent joint limits and (1d) enforces the solution to remain in the trust region Δ.

Problem (1) is solved using an ℓ_1-norm penalty, where each collision avoidance inequality is penalized using a hinge function. This leads to the convex objective (2):

$$\widehat{\phi}(s) = \widehat{f}(s) + \mu \sum_{l=1}^{L} \sum_{k=1}^{K} \sum_{t=1}^{T} \max(0, a_{tkl}^\top s_t + b_{tkl}) \qquad (2)$$

where $\mu > 0$ is a regularizer parameter. In practice, eq. (2) is solved by adding slack variables and additional constraints for the non-smooth terms to keep the problem a quadratic program. Other linear constraints (i.e., joint velocity limits or accelerations) can be seamlessly added to the formulation.

SCP algorithms, such as TrajOpt, keep track of the relationship between the original costs and their convexified versions to accept or reject the current solution. This procedure works as a line search: if the error between the model and the actual merit function remains low and there is a significant improvement in the merit function, the solution is accepted. Otherwise, it is rejected. Specific procedures for automatically adjusting the size of the trust region and for enforcing hard constraints are also key steps in SCP algorithms [6], [7].

B. Sensing Uncertainty

We represent sensing uncertainties as noise that contaminates the 3D poses of uncertain objects. This type of model is common when the shape of obstacles is known beforehand, but there is uncertainty in their pose [9], [16], [18]. Sensing systems based on computer vision and pose estimation provide this type of sensing uncertainty [23]. Under this model, we refer to the nominal obstacle O as the set of points occupied by an obstacle at its expected position, as given by the perception system. The uncertain obstacle $\mathcal{O}$ is the set of all points that result from corrupting the points in the nominal obstacle with additive random noise:

$$\mathcal{O} = \{x + \gamma \mid x \in O\} \qquad (3)$$

where $\gamma \in \mathbb{R}^3$ is some multivariate random variable. Most existing works assume that γ is Gaussian [9], [16], [18]. However, there is an increasing interest in the non-Gaussian case [18], [24]. In this work, we do not assume a specific distribution for γ, but only that it does not lead to unbounded uncertainty sets for the parameters of the approximated signed distance function (see Sec. III-A).

III. ROBUST-OPTIMIZATION MOTION PLANNING

Our proposal is not tied to a specific probability distribution of the corruption noise. *We encourage robust trajectories by robustifying each convex subproblem of a SCP planner.*

We refer to our method as Robust-Optimization Motion Planning (ROMP) (See Alg. 1). It builds off of a SCP planner, such as TrajOpt [6], with two main loops: one

[3] We show the discrete collision avoidance formulation for clarity of presentation. However, all the analysis shown here can be seamlessly transferred to the continuous case.

for every convex subproblem (line 1) and the other for the trust region iteration (line 5). First, the non-convex problem is convexified around the current solution (line 2). For the convex subproblem $(\widehat{f}, \widehat{g})$ and with access to the noise probability distribution p_γ, we estimate parameters of the robust formulation $(\widehat{a}, \widehat{b}, \delta)$ (line 3). We build the robust formulation of the current subproblem in line 4. Next, we minimize the robust merit formulation (line 6). The non-convex merit function is updated, taking into account the robustified convex problem (line 7). The trust region acceptance criteria are checked to decide whether the current solution is accepted or not (line 8). Finally, convergence is checked according to the desired criteria (line 9).

Algorithm 1: ROMP

input : Initial trajectory q^0
output : Optimal Trajectory q^*

1 **for** *SCPIteration* $\leftarrow 1, 2, \ldots$ **do**
2 $\quad (\widehat{f}, \widehat{g}) \leftarrow$ `Convexify` (f, g);
3 $\quad (\widehat{a}, \widehat{b}, \delta) \leftarrow$ `EstimateParams` $(\widehat{f}, \widehat{g}, p_\gamma)$;
4 $\quad (\widehat{f}_r, \widehat{g}_r) \leftarrow$ `RobustifyProblem` $(\widehat{f}, \widehat{g}, \widehat{a}, \hat{b})$;
5 $\quad$ **for** *TrustRegionIteration* $\leftarrow 1, 2, \ldots$ **do**
6 $\quad\quad (s^\star, r^\star) \leftarrow \min \widehat{\phi}_r(s, r)$ subject to linear constraints and trust region;
7 $\quad\quad \phi_r \leftarrow$ `MeritUpdate` $((s^\star, r^\star), \delta)$;
8 $\quad\quad q \leftarrow$ `AcceptanceCriteria` $(\phi_r, \widehat{\phi}_r)$;
9 $\quad\quad$ **if** `CheckConvergence` () **then**
10 $\quad\quad\quad$ Return q or break;

A. Uncertain Convex Subproblem

The model described in Sec. II-B will result in uncertainty in the signed distance function between the robot links and uncertain obstacles in the environment. This uncertainty is propagated through the first-order approximation of the signed distance and leads to uncertainty in parameters (a_{tkl} and b_{tkl}) of the linearized collision avoidance constraints (1b) of every convex subproblem.

In ROMP, we assume that the uncertain parameters of each convex subproblem follow an unknown, but bounded and symmetric, probability distribution around a nominal value. Let uncertain parameters a_{tkl} and b_{tkl} take values in $[\bar{a}_{tkl} - \widehat{a}_{tkl}, \bar{a}_{tkl} + \widehat{a}_{tkl}]$ and $\left[\bar{b}_{tkl} - \widehat{b}_{tkl}, \bar{b}_{tkl} + \widehat{b}_{tkl}\right]$ respectively, where $\bar{a}_{tkl}, \bar{b}_{tkl}$ are coefficients that correspond to the nominal environment, and $\widehat{a}_{tkl}, \widehat{b}_{tkl} \geq 0$ are the maximum deviations from nominal, due to the uncertain obstacle poses.

With this uncertainty model, each collision avoidance constraint (1b) can be protected against the worst deviation that each parameter may have, with respect to its nominal value. I.e., we are interested in solutions s that are feasible for *any* realization of the parameters in their corresponding uncertainty sets. This can be achieved as follows:

$$\bar{a}_{tklg} \cdot s_{tg} + \bar{b}_{tkl} + \widehat{a}_{tklg} \cdot |s_{tg}| + \widehat{b}_{tkl} \leq 0, \tag{4}$$

where s_{tg} corresponds to the g-th entry of vector s_t. The third and fourth term correspond to positive deviations from nominal values that are forced to remain in the feasible set. A reformulation of (4), without absolute values, can be incorporated into (1), leading to:

$$\underset{s,r}{\text{minimize}} \quad \widehat{f}(s) \tag{5a}$$

$$\text{subject to} \quad \bar{a}_{tkl}^\top s_t + \bar{b}_{tkl} + \widehat{a}_{tkl}^\top r_t + \widehat{b}_{tkl} \leq 0 \tag{5b}$$

$$-r_i \leq s_i \leq r_i \tag{5c}$$

$$W(s_t + q_t^0) - w \leq 0 \ , \|s\| \leq \Delta, r_i \geq 0. \tag{5d}$$

Here, $r \in \mathbb{R}^{Td}$ is a vector of artificial variables that result in $r^\star = |s^\star|$ (component-wise) at its optimal value. After convergence, formulation (5) will attain an optimal value $s^\star$ that will be feasible in (1), even in the presence of the uncertain parameters a_{tkl} and b_{tkl}.

Constraints (5b) and (5c) can be used as penalties in the objective of (5)—using hinge functions and slack variables. The problem can still be solved as a quadratic program using off-the-shelf solvers. This step corresponds to line 4 of Alg. 1.

The convex merit function associated with (5) can be written as follows:

$$\widehat{\phi}_r(s, r) = \widehat{f}(s) + \mu \sum_{l,k,t} \Big(\max(0, \bar{a}_{tkl}^\top s_t + \bar{b}_{tkl} + \widehat{a}_{tkl}^\top r_t + \widehat{b}_{tkl}) + \sum_{g=1}^{d} \max(0, -r_{tg} - s_{tg}) + \sum_{g=1}^{d} \max(0, s_{tg} - r_{tg}) \Big)$$

where terms in color are additional costs added in ROMP, as compared to classical TrajOpt. Formulation (5) is one example of Robust Optimization (RO) [20], [25]. Our formulation assumes cardinality constrained uncertainty set. However, other shapes for the uncertainty set have been explored in the literature of RO such as ellipsoidal and polyhedral [19].

B. Estimation of Robust Parameters

At every iteration, ROMP computes the robust parameters $\widehat{a}_{tkl}, \widehat{b}_{tkl}$ (line 3 in Alg. 1). These values depend on the type of noise that corrupts the nominal obstacle's pose, and how this noise affects the signed distance, contact points and normal vectors between objects.

We estimate these values by sampling scenes based on the noise probability distribution p_γ. For each sampled scene, we compute the new values for the distances, normal vectors, contact points and robot Jacobians, and use them to linearize the signed distance function around the current trajectory. We compare each computed value against the corresponding value in the nominal scene and keep the largest.

In practice, unbounded uncertainty distributions in the obstacles' poses might produce parameter uncertainties that do not meet the boundedness condition assumed by ROMP (e.g., Gaussian additive noise in the translation component). For those cases, we constraint samples to stay within certain confidence interval of the probability distribution (e.g., an interval that contains 95% of all the realizations of the random variable). This approach has proven to work well in practice.

C. Merit Function Update

A critical step in ROMP involves updating the original merit function in a way that the model's merit function at the optimal value remains a close representation of the original problem, and the algorithm can make progress. The cost of the "robustified" convex subproblem (5) after convergence will be an upper bound of the cost of all instances of the uncertain programs [26]. For this reason, in ROMP, we increase the costs related to uncertain objects in the scene, by adding a multiple of the worst case-deviation of the signed distance for the corresponding link and obstacle:

$$\phi_r(q) = f(q) + \mu \sum_{l,kt} \max(0, d_{safe} - \text{sd}_{kl}(q_t) + \kappa \cdot \delta_{klt})$$

Here, in color, δ_{klt} is the maximum deviation of the signed distance between obstacle k and link l, when the robot is at configuration q_t due to uncertainty in the obstacle pose; κ is a parameter to be tuned.

The value of δ_{klt} represents a lower bound on the amount of cost that needs to be added to $\phi_r(q)$ to accurately represent $\widehat{\phi}_r(q)$. This parameter can be estimated when the robust parameters are computed (Sec. III-B) without additional computational effort. For specific probability distributions, it may also be analytically computed. Here, we show the case for Gaussian translational uncertainty.

Gaussian translational uncertainty: Assume that $\mathcal{K} \subset \mathbb{R}^3$ is the space occupied by the "nominal obstacle" k. Due to the (translational) sensing uncertainty, the "uncertain obstacle" k_u occupies the space $\mathcal{K}_u = \{j + \xi : j \in \mathcal{K}\}$ and $\xi \sim \mathcal{N}(0, P_\xi)$. As in [6], we assume that the signed distance can be linearized by keeping constant the contact points p_l, p_k and the normal vector $\widehat{n}$, and, therefore, can be expressed as

$$\text{sd}(q, d, \theta, \xi) \approx \widehat{n}^\top (F_l(q)p_l - F_\mathcal{K}(d, \theta, \xi)),$$

where F_l and $F_\mathcal{K}$ are transformations from local to global coordinates of the robot link and the obstacle, respectively. If we consider $R(\theta)$ the rotation component of the transformation, the signed distance can be written as:

$$\begin{aligned} \text{sd}(q, d, \theta, \xi) &= \widehat{n}^\top F_l(q)p_l - \widehat{n}^\top (R(\theta)p_k + d + \xi) \\ &= \widehat{n}^\top (F_l(q)p_l - F_\mathcal{K}(d, \theta, 0)p_k) - \widehat{n}^\top \xi \\ &= \text{sd}(q, d, \theta, 0) - \widehat{n}^\top \xi \end{aligned} \tag{6}$$

Eq. (6) implies that the signed distance between link l and obstacle k_u is a random variable with Gaussian distribution, when the robot is at configuration q_t; i.e.,

$$sd^\xi_{klt} \sim \mathcal{N}(sd^0_{klt}, \widehat{n}^\top P_\xi \widehat{n}).$$

We know that the signed distance random variable will lie outside $sd^0_{klt} \pm \left(\sqrt{2\widehat{n}^\top P_\xi \widehat{n}}\right) \cdot \text{erf}^{-1}(2p-1)$ with probability $2(1-p)$, where erf^{-1} is the inverse error function [27]. This allows us to write the desired maximum deviation:

$$\delta_{klt} = \sqrt{2\widehat{n}^\top P_\xi \widehat{n}} \cdot \ \text{erf}^{-1}(2p-1),$$

where p is such that the random variable lies inside the interval of interest with certain given confidence.

IV. RELATED WORK

Research in safe motion planning in dynamic environments has a long history. Chance constraints have been used to represent the obstacle avoidance problem as a Disjunctive Linear Program [28] but also to optimize risk allocation using linear quadratic programming and robust model predictive control [29].

Other approaches use a similar probabilistic framework to model uncertainties, but under sampling-based methods. CC-RRT [9] is based on a modified version of RRT, where a tree of Gaussian probability distributions is grown, and each distribution in the tree is probabilistically feasible. The chance constraints are transformed into deterministic constraints, that are included in the validity checker of RRT. This approach initially allowed to model sensing and action uncertainty for linear systems, and was later extended [18] for non-Gaussian uncertainty, nonlinear systems and asymptotically optimal planners [10]. Similarly, a distributionally robust version of RRT has been proposed [11]. There, the uncertainty is assumed to come from a family of probability distributions, characterized by their first and second order moments and the optimization is performed over the worst-case distribution in the ambiguity set. In [30] the authors propose online generation of robust motion plans for nonlinear dynamics and bounded disturbances, control constraints and obstacles using feedback controllers, contraction theory and convex optimization.

The aforementioned literature is concerned with dynamical systems where the uncertainty in the robot's state dominates; they are also motivated by low-DOF robots, such as cars and aerial robots. In comparison, there is limited work on high-DOF robots, such as manipulators, where scene uncertainty has a larger impact on the safety of the planned trajectories. More recently and closer to our work, the P-Chekov planner has been proposed [15]. It relies on a deterministic planner that successfully combines sampling-based and optimization-based methods into a sparse roadmap framework. P-Chekov uses quadrature theories to estimate collision risk for every waypoint. This planner can be used for high-DOF robots, but only takes into account state uncertainty. A recently proposed a robust motion planner based on SCP [17] leverages signed distance functions and chance-constraints for nonlinear dynamical systems, under model uncertainty and disturbances. Similar to the work presented here, their work rely on SCP and signed distances. However, their theoretical guarantees and motivation hold for low-DOF robots.

V. EXPERIMENTS

We test ROMP on two motion planning tasks, namely the **Square Robot** and the **Fetch Tabletop**. The former is an omnidirectional square robot translating and rotating in the plane in an environment with random convex shapes. The latter is a 7-DOF Fetch robot arm performing tabletop operations. For both experiments, we study the behavior of ROMP for different types of uncertainty models. Overall, we highlight how ROMP can effectively incorporate uncertainty information into the planning problem and create robust

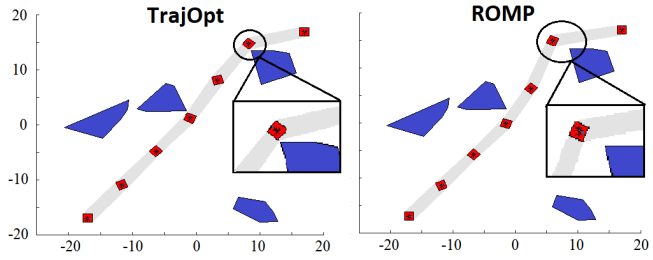

Fig. 2: Trajectory computed by TrajOpt (left) and ROMP (right) for one random scene of the *Square robot* task. Obstacles (blue) are randomly generated convex shapes. The grey shapes between consecutive waypoints are the approximated volume swept by the robot, when moving along the trajectory waypoints (shown in red).

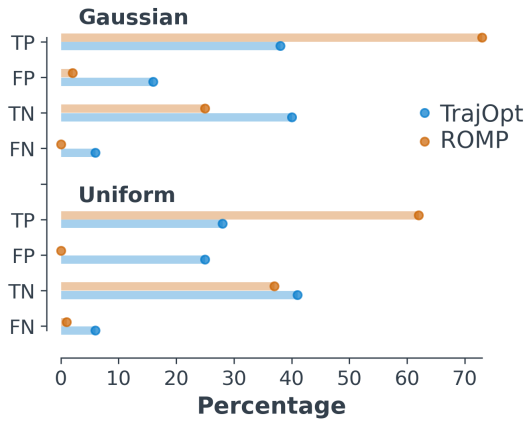

Fig. 3: Percentage of TP/FP/TN/FN for TrajOpt and ROMP, when solving 1000 different scenes in the presence of Gaussian and Uniform uncertainty.

trajectories, when compared to other uncertainty unaware planners. All the experiments were run on an Intel Xeon E-2136 CPU.

Square robot. We study the performance of TrajOpt and ROMP in 1000 noisy randomly generated scenes. The square robot starts at the lower left corner of the environment and is required to reach a configuration at the top right avoiding obstacles (See Fig. 2). Each scene consists of 4 to 6 randomly generated convex uncertain obstacles spread across the environment. We considered Gaussian and Uniform uncertainty in the position of each obstacle (in both x and y axes). For planning, we build the collision avoidance constraints using the full body of the robot instead of simplifying it as a point robot. For every scene, both TrajOpt and ROMP are initialized with a straight-line trajectory in SE(2). In the experiments we have used Box2D [31] as collision manager and Gurobi [32] as backend optimizer. Finally, we have rejected scenes where trivial (i.e., straight-line) trajectories can solve the problem. Fig. 2 shows an example of a trajectory generated by TrajOpt and ROMP on the nominal environment.

TrajOpt is unaware of the obstacle noise: it generates collision-free trajectories, up to the safety distance parameter. However, as the trajectories run close to the obstacles, they collide with the uncertain obstacles. ROMP on the other hand, pushes the trajectory further away from the obstacles.

For every scene, we have computed the percentage of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). The first two correspond to cases where the output trajectory is collision-free with respect to the nominal scene and "true" and "false" indicate whether the trajectory is also collision-free for the uncertain scene or not respectively (similarly, for the negative examples). Fig. 3 shows the results of 1000 random scenes with Uniform and Gaussian uncertainty with $\mathcal{U}(-1.0, 1.0)$ and $\mathcal{N}(0, \sigma^2 I)$, $\sigma = 0.3$ in the obstacles' position. The effect of creating robust convex sub-problems can be seen in the increased number of true positives and the significant decrease of both false positives and false negatives.

Fetch tabletop. We simulate a Fetch robot executing tabletop operations using its 7-DOF arm with the base and torso fixed (See Fig. 1). The pose of the table is accurately known, but the elements on top are subject to uncertainty in the direction normal to the robot's body. The goal is to generate safe trajectories, as the robot moves the cylinder from the left to the right. We use RRT-Connect [33] and TrajOpt as uncertainty-unaware planners to highlight how different planners are affected by the sensing uncertainty. RRT-Connect implements a post-processing heuristic to smoothen the generated trajectory [34]. TrajOpt implements continuous collision avoidance constraints (see [6]), and ROMP formulations are adapted accordingly. We use RRT-Connect implementation from OMPL [35] through MoveIt! [36]. ROMP is implemented on top of the ROS-Industrial's TrajOpt implementation [37], where the collision costs were modified to accommodate the additional terms in the robust formulation, and new artificial constraints and variables were added. All planners are benchmarked using Robowflex [38].

We let RRT-Connect, TrajOpt and ROMP solve the manipulation problem shown in Fig. 1 using the nominal scene and compute the estimated probability of collision by sampling 5000 scenes from 10 different probability distributions. We use Gaussian and Uniform distributions with increasing values of standard deviation and bounds: from 15 to 25cm and 2.5 to 5.5cm, respectively. The values of these parameters were chosen to produce similar probability of collisions in the uncertainty-unaware cases. For the Gaussian distributions, we reject samples outside of the interval, where 95% of the data lie when estimating the parameters of our robust formulation. However, we do not reject any sampled scene when estimating the probability of collision.

TrajOpt is initialized using a straight line in the manipulator's joint space, while ROMP is fed with the trajectory found by TrajOpt. RRT-Connect uses default parameter values. Fig. 1 shows an example of a trajectory generated by ROMP. The three planners generate trajectories with configurations that are close to the noisy object. However, for ROMP the trajectory is pulled closer to the robot due to the uncertainty in that direction. Fig. 4 shows the estimated probability of collision for both types of uncertainty (Left for Gaussian; Center for Uniform). ROMP finds trajectories that keep a low

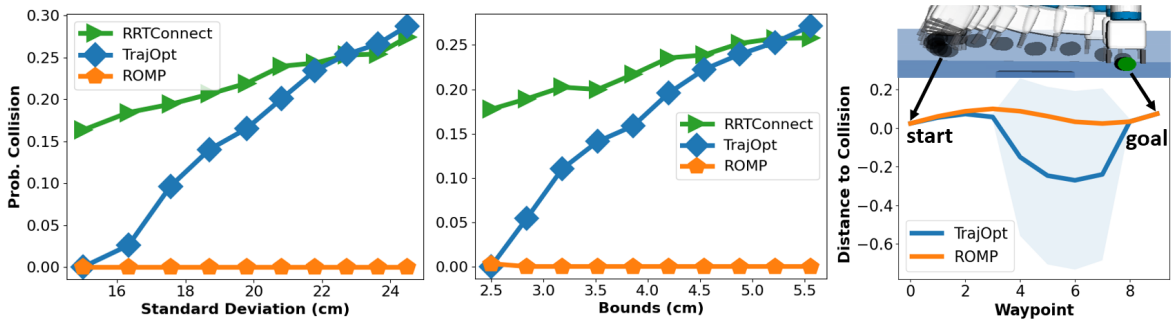

Fig. 4: Estimated probability of collision for trajectories computed by the three planners in the Fetch Tabletop task for varying levels of **(Left)** Gaussian and **(Center)** Uniform noises. **(Right)** Distance to collision between the Fetch arm and the noisy object throughout the trajectory, under Gaussian noise with $\sigma = 24$cm. The x-axis corresponds to each waypoint, while y-axis denotes "distance to collision", where negative values indicate collision. Solid lines show mean distance and shadows show 1 standard deviation over the 5000 sampled scenes. At the top we show the robot's end-effector position at every waypoint, starting where the cylinder is and finishing at the opposite side.

probability of collision for a range of noise, where the other planners generates a large number of collisions. Furthermore, the results show that ROMP is capable of handling both translational Gaussian and Uniform distributions.

In Fig. 4(Right), we plot the distance to collision at every waypoint between the robot's arm and the noisy obstacle for the Gaussian uncertainty with $\sigma = 24$cm. On top we show the pose of the end-effector for one trajectory generated by ROMP. Waypoint 0 corresponds to the grasping location at the cylinder (left) and waypoint 9 is the goal configuration at the other side of the block (right). The negative mean distance and large deviation by TrajOpt at mid-trajectory waypoints are the result of the noisy obstacle. ROMP consistently generates trajectories that are not excessively conservative, since both the mean distance and the standard deviation remain low. Table I shows probability of collision, planning time and path length for $\sigma = 24$cm (Gaussian) and bound$=$ 5.5cm (Uniform). The small difference between ROMP's trajectories in path length also supports that conservatism of the solution is not excessively large. Our method exhibits planning times $\sim$1-2 seconds, due to the incorporation of the uncertainty which is reasonable for real-time applications.

Discussion. ROMP effectively extends TrajOpt when sensing uncertainty is present during planning. This is achieved by adding artificial variables and estimating parameters that capture deviation from expected values in SCP.

While writing this work, we did not find planners in the literature that incorporate uncertainty for high-DOF robots to compare against ROMP. One work that stands out is [15], where a "Roadmap+TrajOpt" planner iteratively modifies the safety distance parameter of the collision avoidance constraints in TrajOpt. However, this method considers only uncertainty in the robot's state and not in the environment.

Recently, [16] proposes the use of convex shapes to estimate the risk of collision for sensing uncertainty. The authors integrate the risk into a SCP-based planner as chance constraints that account for the trajectory's accumulated risk. The assumption of Gaussian additive translational noise for the uncertain obstacles may limit its applicability. It would be interesting as a future work to compare this new approach to ROMP for the cases of Gaussian translational uncertainty.

TABLE I: Benchmark results (mean and standard deviation) for RRT-Connect, TrajOpt and ROMP in the Fetch tabletop task for Gaussian noise with $\sigma = 24$cm and Uniform noise with bound 5.5cm.

	Gaussian $\sigma = 24$cm		
	Prob. Collision	**Time (s)**	**Length (rad)**
RRT-Connect	0.2740	0.0715 ± 0.0005	13.03 ± 19.12
TrajOpt	0.2872	0.0663 ± 0.0000	5.424 ± 0.000
ROMP	0.0	2.2277 ± 0.0525	5.871 ± 0.000
	Uniform bound $= 5.5$cm		
	Prob. Collision	**Time (s)**	**Length (rad)**
RRT-Connect	0.2580	0.0689 ± 0.0005	12.88 ± 18.61
TrajOpt	0.2716	0.0662 ± 0.0000	5.424 ± 0.000
ROMP	0.0	1.0091 ± 0.1139	5.700 ± 0.000

VI. Conclusions

We propose ROMP, a method that extends motion planners based on SCP to incorporate sensing uncertainty into the planning process. Two important aspects make ROMP useful in realistic applications. i) ROMP uses the collision avoidance constraints, proposed in TrajOpt, allowing the use of high-DOF robots in complex environments. This is an advantage over the majority of uncertainty-aware planners that assume a simplified representation of the robot. ii) ROMP is not tied to a particular noise type, which is appealing for various tasks. Its major assumption is that the parameters of the first-order approximation for the signed distance function are bounded.

As future work, we plan to develop methodologies to estimate the parameters of the robust formulation efficiently. E.g., methods that learn these statistics from previous experiences would be interesting to explore. Finally, we plan to explore more sophisticated shapes of the uncertainty sets, such as polyhedral and ellipsoidal, that may provide the user with control over the conservatism of the solution and the amount of allowable risk.

REFERENCES

[1] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[2] S. M. LaValle, J. J. Kuffner, and Jr., "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[3] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[4] 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," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[5] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4569–4574.

[6] 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," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[7] R. Bonalli, A. Cauligi, A. Bylard, , and M. Pavone, "GuSTO: guaranteed sequential trajectory optimization via sequential convex programming," in *2019 IEEE Conf. on Robotics and Automation*, 2019.

[8] B. Burns and O. Brock, "Sampling-based motion planning with sensing uncertainty," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, April 2007, pp. 3313–3318.

[9] B. Luders, M. Kothari, and J. How, "Chance constrained RRT for probabilistic robustness to environmental uncertainty," in *AIAA guidance, navigation, and control conference*, 2010, p. 8160.

[10] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013.

[11] T. Summers, "Distributionally robust sampling-based motion planning under uncertainty," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 6518–6523.

[12] B. Axelrod, L. P. Kaelbling, and T. Lozano-Pérez, "Provably safe robot navigation with obstacle uncertainty," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1760–1774, 2018. [Online]. Available: https://doi.org/10.1177/0278364918778338

[13] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, Dec 2011.

[14] W. Sun, J. Van den Berg, and R. Alterovitz, "Stochastic extended LQR for optimization-based motion planning under uncertainty," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 437–447, April 2016.

[15] S. Dai, S. Schaffert, A. Jasour, A. Hofmann, and B. Williams, "Chance constrained motion planning for high-dimensional robots," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[16] C. Dawson, A. Jasour, A. Hofmann, and B. Williams, "Provably safe trajectory optimization in the presence of uncertain convex obstacles," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 6237–6244.

[17] T. Lew, R. Bonalli, and M. Pavone, "Chance-constrained sequential convex programming for robust trajectory optimization," in *European Control Conference*, St. Petersburg, Russia, May 2020, in Press. [Online]. Available: /wp-content/papercite-data/pdf/Lew.Bonalli.Pavone.ECC20.pdf

[18] B. Luders and J. How, "Probabilistic feasibility for nonlinear systems with non-gaussian uncertainty using RRT," in *Infotech@Aerospace 2011*, 2011.

[19] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Review*, vol. 53, no. 3, pp. 464–501, 2011.

[20] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton University Press, 2009.

[21] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 1917–1922.

[22] B. R, A. Bylard, A. Cauligi, T. Lew, and M. Pavone, "Trajectory optimization on manifolds: A theoretically-guaranteed embedded sequential convex programming approach," in *Robotics: Science and Systems*, 2019.

[23] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Robotics Science and Systems (RSS) XIV*, 2018.

[24] J. S. Park and D. Manocha, "Efficient probabilistic collision detection for non-gaussian noise distributions," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1024–1031, 2020.

[25] D. Bertsimas and M. Sim, "The price of robustness," *Operations Research*, vol. 52, no. 1, pp. 35–53, 2004.

[26] A. Ben-tal and A. Nemirovski, "Robust convex optimization," *Mathematics of Operations Research*, 1998.

[27] A. Prékopa, *Stochastic Programming*, 1st ed. Springer Netherlands, 1995, vol. 324.

[28] L. Blackmore, Hui Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *2006 American Control Conference*, June 2006, pp. 7 pp.–.

[29] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *47th IEEE Conference on Decision and Control (CDC*, 2008, p. 427–3432.

[30] S. Singh, A. Majumdar, J. Slotine, and M. Pavone, "Robust online motion planning via contraction theory and convex optimization," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 5883–5890.

[31] E. Catto, "Box2D," 2020. [Online]. Available: https://box2d.org/

[32] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2020. [Online]. Available: http://www.gurobi.com

[33] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, 2000, pp. 995–1001 vol.2.

[34] R. Luna, I. A. Şucan, M. Moll, and L. E. Kavraki, "Anytime solution optimization for sampling-based motion planning," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 5068–5074.

[35] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, https://ompl.kavrakilab.org.

[36] D. Coleman, I. A. Şucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a MoveIt! case study," *Journal of Software Engineering for Robotics*, 2014.

[37] R. I. Consortium, "Trajopt ROS," 2020. [Online]. Available: https://github.com/ros-industrial-consortium/trajopt_ros

[38] Z. Kingston and L. E. Kavraki, "Robowflex: Robot motion planning with *MoveIt* made easy," *IEEE Robotics and Automation Letters*, 2021, under Review. [Online]. Available: https://arxiv.org/abs/2103.12826