Sigma Hulls for Gaussian Belief Space Planning for Imprecise Articulated Robots amid Obstacles

Alex Lee Yan Duan Sachin Patil John Schulman Zoe McCarthy
Jur van den Berg Ken Goldberg Pieter Abbeel

***Abstract*— In many home and service applications, an emerging class of articulated robots such as the Raven and Baxter trade off precision in actuation and sensing to reduce costs and to reduce the potential for injury to humans in their workspaces. For planning and control of such robots, planning in *belief space*, i.e., modeling such problems as POMDPs, has shown great promise but existing belief space planning methods have primarily been applied to cases where robots can be approximated as points or spheres. In this paper, we extend the belief space framework to treat articulated robots where the linkage can be decomposed into convex components. To allow planning and collision avoidance in Gaussian belief spaces, we introduce the concept of *sigma hulls*: convex hulls of robot links transformed according to the sigma standard deviation boundary points generated by the Unscented Kalman filter (UKF). We characterize the signed distances between sigma hulls and obstacles in the workspace to formulate efficient collision avoidance constraints compatible with the Gilbert-Johnson-Keerthi (GKJ) and Expanding Polytope Algorithms (EPA) within an optimization-based planning framework. We report results in simulation for planning motions for a 4-DOF planar robot and a 7-DOF articulated robot with imprecise actuation and inaccurate sensors. These experiments suggest that the sigma hull framework can significantly reduce the probability of collision and is computationally efficient enough to permit iterative re-planning for model predictive control.**

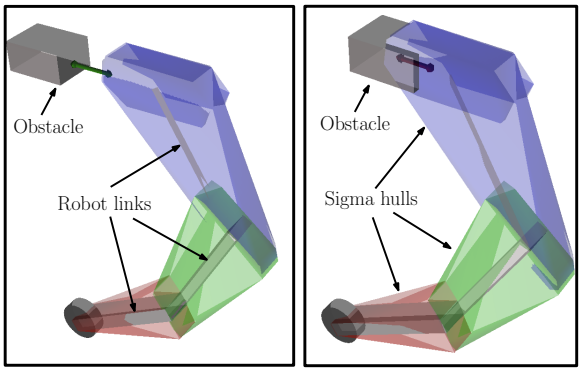
I. Introduction

Our work is motivated by the desire to facilitate robust operation of cost-effective robots such as the Raven surgical robot [23], Baxter manufacturing robot [22], and low-cost manipulators [21]. These robots use inexpensive actuation methods such as cable-driven mechanisms and serial elastic actuators that are less precise than stiff geared actuators. They also rely on inexpensive, inaccurate encoders and other sensors such as accelerometers to sense the robot state. For such robots to robustly complete navigation and manipulation tasks under motion and sensing uncertainty, it is important for the robot to explicitly perform information gathering actions to minimize the effects of uncertainty.

Alex Lee, Yan Duan, Sachin Patil, John Schulman, Zoe McCarthy, Ken Goldberg, and Pieter Abbeel are with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, CA, USA. {alexlee_gk, dementrock, sachinpatil, joschu, zmccarthy, goldberg, pabbeel}@berkeley.edu

Jur van den Berg is with the School of Computing, University of Utah, Salt Lake City, UT, USA. berg@cs.utah.edu

This research was supported in part by the National Science Foundation (NSF) under award # IIS-1227536, by Air Force Office of Scientific Research (AFOSR) under Young Investigator Program (YIP) award # FA9550-12-1-0345, by a Sloan Fellowship, and by the Intel Science and Technology Center on Embedded Computing.



(a) Obstacle outside sigma hulls (b) Obstacle overlaps sigma hulls

Fig. 1. Sigma hulls of the individual links of a 3-DOF robot (red, green, and blue) and an obstacle in the workspace (gray). The sigma hulls are visualized here only for the sake of illustration and they are never explicitly constructed in our approach. We use sigma hulls to formulate collision avoidance constraints in terms of the signed distance to obstacles. (a) A positive signed distance between a sigma hull and obstacle indicates the the obstacle is outside the sigma hull and corresponds to the length of the smallest translation that puts the two shapes into contact [12]. (b) A negative signed distance indicates that the obstacle overlaps with a sigma hull and corresponds to the minimum translation length that separates the two shapes, given by the penetration depth [1].

The problem of maximizing information gain under uncertainty is often formalized as a Partially Observable Markov Decision Process (POMDP) [15] and is defined over the space of probability distributions of the state space, also known as the belief space. Since finding globally optimal solutions to the POMDP problem is computationally intractable [6], prior work has focused on solving the POMDP problem for Gaussian belief spaces using sampling-based motion planners [20], [5] or computing locally-optimal solutions using optimization-based methods [11], [19], [29], [27].

Collision detection is an integral component of motion planning. Motion planning algorithms perform collision detection in the workspace since computing configuration space obstacles is not practical [8]. However, for planning in belief spaces, collision detection needs to be performed in a probabilistic sense by considering collisions with respect to all possible states that the robot could be in. Prior Gaussian belief space planning methods only consider point robots or spherical approximations of the robot geometry, for which probabilistic collision detection can be performed efficiently in the workspace [5], [29], [27]. In the case of articulated robots that are not well approximated as spheres, one would have to apply these methods directly in the configuration space. An alternative approach would be to use Monte Carlo sampling [17], which would be very computationally expensive for belief space planning.

There are two main contributions of our work. First, we introduce the notion of *sigma hulls* for collision avoidance in Gaussian belief spaces. We consider sigma points on the boundary of the standard deviation contour of the Gaussian distribution as computed by the Unscented Kalman filter (UKF). Sigma hulls are convex hulls of the geometry of individual robot links transformed according to the sigma points in joint space. Second, we use the signed distances between the sigma hulls and workspace obstacles to formulate collision avoidance constraints within an optimization-based framework based on sequential convex optimization [4]. We rely on advances in collision detection [1], [12] to formulate these constraints without explicitly computing the sigma hulls. We show that the collision avoidance constraints can be locally approximated by convex constraints, and we derive how to compute this convex approximation analytically.

We adopt the approach of Platt et al. [19] to use trajectory optimization [3] for computing a locally optimal trajectory in belief space. The optimization formulation incorporates constraints on the state and control inputs and is well-suited for highly under-actuated belief space planning problems. We build on recent advances in handling collisions in state space trajectory optimization [25] to plan motions for articulated robots in Gaussian belief spaces. As is standard in nonlinear optimization for control, we follow the model predictive control (MPC) paradigm [7] of re-planning after every time step. This has been demonstrated to be a very effective way of performing feedback control to remain robust to large perturbations, provided one can re-plan sufficiently fast.

We present results in simulation for planning motions for a 4-DOF planar articulated robot and a 7-DOF articulated robot with imprecise actuation and sensing. Our experiments indicate that our approach significantly reduces the probability of collision and the error of arrival at the desired target during execution. These initial results are encouraging and we posit that advances in optimization techniques and computational hardware will eventually make our approach computationally efficient to enable robots to plan at a high frequency in belief spaces. This would be an important step towards enabling low-cost robots to safely and robustly complete navigation and manipulation tasks in unstructured environments while operating under considerable uncertainty.

II. Related Work

Recently, research efforts have focused on the planning problem under uncertainty, which in generality is formalized as a POMDP, computing solutions for which is computationally intractable [6]. For problems involving discrete state, action, and/or observation spaces, algorithms have been developed that use approximate value iteration [9], [16]. However, for problems more naturally defined over continuous state, action, and observation spaces such as those arising in robot manipulation and navigation, discretizing the problem and using the aforementioned approaches leads to an exponential growth in the number of states, subjecting these problems to the curse of dimensionality [24]. The methods of [26], [13] handle continuous state and action spaces, but maintain a global (discrete) representation of the value function over the belief space, which limits their applicability to small to medium sized domains.

For problems in which it is reasonable to model beliefs as Gaussian distributions, sampling-based motion planning [20], [5] or optimization-based methods [11], [19], [29], [28] can be used for belief space planning. This concept can be extended to non-Gaussian beliefs [18] by using particle filters. These methods handle continuous spaces but only consider robots with point-like or spherical geometries to simplify probabilistic collision detection.

We adopt the approach of Platt et al. [19] to use trajectory optimization [3], [25] in belief space assuming deterministic belief dynamics. We generalize this approach to plan motions for articulated robots that are not well approximated as points or spheres. The uncertainty during execution is mitigated by re-planning after each time step [7]. In doing so, we propose a novel formulation of collision avoidance constraints in belief space using sigma hulls.

III. Preliminaries and Objective

We consider an articulated robot with K links. We are given a description of the robot geometry and a geometric description of the obstacles $\mathcal{O}$ in the workspace. Let $\mathbf{x}$ denote the robot state, which includes the degrees of freedom (DOF) of the robot such as joint angles and position of the base. We refer to the position of a given point on the robot geometry in a world coordinate frame as a function of $\mathbf{x}$ as $\mathbf{p}(\mathbf{x})$ and the pose (both position and orientation) as $\boldsymbol{\psi}(\mathbf{x})$, which is evaluated in closed form using the robot kinematics model. We assume that the state space trajectory is discretized into time intervals of equal duration $\mathcal{T} = \{0, 1, \ldots, T\}$.

At each time step $t \in \mathcal{T}$, the stochastic kinematics of the robot state $\mathbf{x}_t$ evolves according to the given model:

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{q}_t), \quad \mathbf{q}_t \sim \mathcal{N}(\mathbf{0}, I_{\dim[\mathbf{q}]}), \tag{1}$$

where $\mathbf{u}_t \in F_{\mathcal{U}}$ is a control input drawn from the set of feasible control inputs $F_{\mathcal{U}}$ and $\mathbf{q}_t$ is the motion noise. Although the state is not observed directly, noisy observations $\mathbf{z}_t$ are obtained using sensors and are related to the state $\mathbf{x}_t$ according to the given stochastic model:

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t, \mathbf{r}_t), \quad \mathbf{r}_t \sim \mathcal{N}(\mathbf{0}, I_{\dim[\mathbf{r}]}), \tag{2}$$

where $\mathbf{r}_t$ is the sensing noise. Without loss of generality, both $\mathbf{q}_t$ and $\mathbf{r}_t$ are drawn from independent Gaussian distributions with zero mean and unit variance and can be scaled appropriately to be state and control input dependent within the functions $\mathbf{f}$ and $\mathbf{h}$, respectively.

Given Gaussian models of motion and sensing uncertainty considered above, the probability distribution over the state or belief is parameterized as a Gaussian distribution. Specifically, the belief state $\mathbf{b}_t = \left[\begin{smallmatrix} \hat{\mathbf{x}}_t \\ \mathrm{vec}[\sqrt{\Sigma_t}] \end{smallmatrix}\right]$ is a vector comprised of the mean state $\hat{\mathbf{x}}_t$ and the columns of the square root $\sqrt{\Sigma_t}$ of the covariance Σ_t of a Gaussian distribution $N(\hat{\mathbf{x}}_t, \Sigma_t)$.

We assume that the initial belief $\mathbf{b}_0 = \left[\begin{smallmatrix} \hat{\mathbf{x}}_0 \\ \mathrm{vec}[\sqrt{\Sigma_0}] \end{smallmatrix}\right]$ is given. Given a current belief $\mathbf{b}_t$, a control input $\mathbf{u}_t$, and

a measurement $\mathbf{z}_{t+1}$, the evolution of the belief state can be described using a Kalman filter [19], [28]. Prior work on Gaussian belief space planning has used the extended Kalman filter (EKF) [30] but we use an unscented Kalman filter (UKF) [14], which propagates the mean and covariance more accurately for highly nonlinear dynamics and observation models as compared to the EKF.

The UKF uses the unscented transform to compute a set of samples or *sigma points* that characterize the λ-standard deviation contours of the Gaussian distribution [14] for a given value of the parameter λ. Given a Gaussian distribution $\mathcal{N}(\hat{\mathbf{x}}, \Sigma)$, $\mathcal{X} = \texttt{SigmaPoints}(\hat{\mathbf{x}}, \sqrt{\Sigma}, \lambda)$ is a set of $(2\dim[\hat{\mathbf{x}}] + 1)$ sigma points given according to:

$$\mathcal{X} = [\hat{\mathbf{x}} \dots \hat{\mathbf{x}}] + \lambda[\mathbf{0} \ \sqrt{\Sigma} \ -\sqrt{\Sigma}], \tag{3}$$

where $[\hat{\mathbf{x}} \dots \hat{\mathbf{x}}]$ denote the repeated copies of the mean $\hat{\mathbf{x}}$ concatenated as columns in a matrix. Since the UKF requires computing the square root of the covariance, we directly use the square root of the covariance included in the belief state. We also exploit the symmetry of $\sqrt{\Sigma_t}$ in our implementation to eliminate the redundancy.

We refer the reader to [14] for the details of the UKF, but at a high level the propagation is as follows, given $\hat{\mathbf{x}}_t$ and Σ_t that characterize the current belief:

$$\mathcal{X}_t = \texttt{SigmaPoints}(\hat{\mathbf{x}}_t, \sqrt{\Sigma_t}, \lambda), \tag{4}$$
$$\mathcal{Q}_t = \texttt{SigmaPoints}(\mathbf{0}, I_{\dim[\mathbf{q}]}, \lambda), \tag{5}$$
$$\mathcal{X}^-_{t+1} = \mathbf{F}(\mathcal{X}_t, \mathbf{u}_t, \mathbf{0}) \cup \mathbf{F}(\hat{\mathbf{x}}_t, \mathbf{u}_t, \mathcal{Q}_t), \tag{6}$$
$$\mathcal{R}_{t+1} = \texttt{SigmaPoints}(\mathbf{0}, I_{\dim[\mathbf{r}]}, \lambda), \tag{7}$$
$$\mathcal{Z}_{t+1} = \mathbf{H}(\mathcal{X}^-_{t+1}, \mathbf{0}) \cup \mathbf{H}(\mathrm{E}[\mathcal{X}^-_{t+1}], \mathcal{R}_{t+1}), \tag{8}$$

where $\mathrm{E}[\cdot]$ refers to the sample mean of a set of samples, and $\mathbf{F}(\cdot)$ and $\mathbf{H}(\cdot)$ refer to the sets of results of the function $\mathbf{f}$ and $\mathbf{h}$ applied to each element of the operand, respectively. The new belief is then computed according to:

$$\hat{\mathbf{x}}_{t+1} = \mathrm{E}[\mathcal{X}^-_{t+1}] + K_t(\mathbf{z}_{t+1} - \mathrm{E}[\mathcal{Z}_{t+1}]), \tag{9}$$
$$\sqrt{\Sigma_{t+1}} = \sqrt{\mathrm{Var}[\mathcal{X}^-_{t+1}] - K_t \, \mathrm{Var}[\mathcal{Z}_{t+1}] K_t^T}, \tag{10}$$

where $K_t = \mathrm{Cov}[\mathcal{X}^-_{t+1}, \mathcal{Z}_{t+1}] \, \mathrm{Var}[\mathcal{Z}_{t+1}]^{-1}$, and $\mathrm{Var}[\cdot]$ and $\mathrm{Cov}[\cdot, \cdot]$ refer to the sample variance and sample covariance respectively of the respective sets of samples. The mean is computed using $s = |\mathcal{X}^-_{t+1}| = 2(\dim[\mathbf{x}] + \dim[\mathbf{q}]) + 1$ points.

The stochastic belief space dynamics is then given by:

$$\mathbf{b}_{t+1} = \mathbf{g}(\mathbf{b}_t, \mathbf{u}_t) + W(\mathbf{b}_t, \mathbf{u}_t)\mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, I), \tag{11}$$
$$\mathbf{g}(\mathbf{b}_t, \mathbf{u}_t) = \begin{bmatrix} \mathrm{E}[\mathcal{X}^-_{t+1}] \\ \mathrm{vec}[\sqrt{\mathrm{Var}[\mathcal{X}^-_{t+1}] - K_t \, \mathrm{Var}[\mathcal{Z}_{t+1}] K_t^T}] \end{bmatrix}, \tag{12}$$
$$W(\mathbf{b}_t, \mathbf{u}_t) = \begin{bmatrix} \sqrt{K_t \, \mathrm{Var}[\mathcal{Z}_{t+1}] K_t^T} \\ 0 \end{bmatrix}. \tag{13}$$

Inspired by Platt et al. [19], we make the assumption that the maximum likelihood observation is obtained at each time step, i.e., $\mathbf{z}_{t+1} = \mathrm{E}[\mathcal{Z}_{t+1}]$, which eliminates the stochasticity from the belief dynamics. This assumption of determinism is just used for efficiently planning locally optimal trajectories in belief space. It is important to note that we consider the stochastic belief dynamics during execution of the computed trajectory. We refer to the nominal state obtained under the maximum likelihood observation assumption as $\hat{\mathbf{b}}$ and the nominal control input as $\hat{\mathbf{u}}$. The deterministic dynamics of the nominal belief $\hat{\mathbf{b}}_t$ evolves as:

$$\hat{\mathbf{b}}_{t+1} = \mathbf{g}(\hat{\mathbf{b}}_t, \hat{\mathbf{u}}_t), \tag{14}$$

where the function $\mathbf{g}$ is as defined in Eq. (12). During execution, we compute the actual belief state $\mathbf{b}_{t+1}$ using the observation $\mathbf{z}_{t+1}$ and re-plan a locally optimal nominal belief space trajectory starting from $\mathbf{b}_{t+1}$, essentially invoking model predictive feedback control (MPC) [7] in belief space.

Objective: To facilitate safe execution of motion plans in Gaussian belief space, our objective is to compute optimal trajectories in belief space that are at least λ-standard deviations safe, where λ is a user-specified parameter. When a λ-standard deviations safe optimal trajectory does not exist, we report that a solution cannot be found.

IV. Trajectory Optimization in Belief Space

We formulate belief space planning as a constrained nonlinear optimization problem which minimizes a user-defined cost function encoding the task objective while satisfying all task constraints. For notational convenience, we concatenate the belief states and control inputs for all time steps $t \in \mathcal{T}$ to form $\hat{\mathcal{B}} = [\hat{\mathbf{b}}_0 \dots \hat{\mathbf{b}}_T]^T$ and $\hat{\mathcal{U}} = [\hat{\mathbf{u}}_0 \dots \hat{\mathbf{u}}_{T-1}]^T$ that parameterize a nominal belief space trajectory such that $\hat{\mathbf{b}}_{t+1} = \mathbf{g}(\hat{\mathbf{b}}_t, \hat{\mathbf{u}}_t) \ \forall \ t \in \{0, \dots, T-1\}$. The optimization problem is then formally stated as:

$$\begin{aligned} \min_{\hat{\mathcal{B}}, \hat{\mathcal{U}}} \quad & \mathbf{C}(\hat{\mathcal{B}}, \hat{\mathcal{U}}) \\ \text{s.t.}_{\forall t \in \mathcal{T}} \quad & \hat{\mathbf{b}}_{t+1} = \mathbf{g}(\hat{\mathbf{b}}_t, \hat{\mathbf{u}}_t), \\ & \boldsymbol{\Phi}(\hat{\mathcal{B}}, \hat{\mathcal{U}}, \lambda) \geq \mathbf{0}, \\ & \boldsymbol{\psi}(\hat{\mathbf{x}}_T) = \boldsymbol{\psi}_{\text{target}}, \\ & \hat{\mathbf{u}}_t \in F_{\mathcal{U}}, \end{aligned} \tag{15}$$

where $\mathbf{C}(\hat{\mathcal{B}}, \hat{\mathcal{U}})$ is a cost function encoding the task objective, $\boldsymbol{\psi}(\hat{\mathbf{x}}_T) = \boldsymbol{\psi}_{\text{target}}$ constrains the robot end effector pose $\boldsymbol{\psi}(\hat{\mathbf{x}}_T)$ at the final time step T to be the desired end effector pose $\boldsymbol{\psi}_{\text{target}}$, and $\hat{\mathbf{u}}_t \in F_{\mathcal{U}}$ constrains the control input $\hat{\mathbf{u}}_t$ to lie in the set of feasible control inputs $F_{\mathcal{U}}$. $\boldsymbol{\Phi}(\hat{\mathcal{B}}, \hat{\mathcal{U}}, \lambda) \geq \mathbf{0}$ enforces that the trajectory is λ-standard deviations safe for probabilistic collision avoidance in belief space. The optimization is initialized with a belief trajectory $\bar{\mathcal{B}} = [\bar{\mathbf{b}}_0 \dots \bar{\mathbf{b}}_T]^T$ and $\bar{\mathcal{U}} = [\bar{\mathbf{u}}_0 \dots \bar{\mathbf{u}}_{T-1}]^T$.

The general form of the optimization problem given in Eq. (15) is challenging because of the highly nonlinear objective and constraints. It is thus difficult to find globally optimal solutions to this problem. We compute locally optimal solutions using sequential convex programming [3], by repeatedly constructing a locally convex approximation to the original problem around the initialization trajectory $(\bar{\mathcal{B}}, \bar{\mathcal{U}})$. We refer the reader to Betts [3] for a detailed exposition of trajectory optimization and Schulman et al. [25] for application of these methods to robot motion planning in state space.

Costs and Constraints: We describe the costs and constraints in detail below:

1) $\mathbf{C}(\hat{\mathcal{B}},\hat{\mathcal{U}})$: We consider the planning objective of accomplishing the desired task while minimizing uncertainty and control effort. The cost function we use is of the form:

$$\mathbf{C}(\hat{\mathcal{B}},\hat{\mathcal{U}}) = \textstyle\sum_{t=0}^{T} \mathrm{tr}[M_t \hat{\Sigma}_t] + \sum_{t=0}^{T-1} \hat{\mathbf{u}}_t^T N_t \hat{\mathbf{u}}_t, \qquad (16)$$

where M_t and N_t are positive definite cost matrices $\forall\ t \in \mathcal{T}$ that weigh the contributions of the two cost terms. The term $\mathrm{tr}[M_t\hat{\Sigma}_t]$ penalizes the uncertainty by considering the departure from zero variance and the term $\hat{\mathbf{u}}_t^T N_t \hat{\mathbf{u}}_t$ is a quadratic cost encoding the total control effort.

2) $\hat{\mathbf{b}}_{t+1} = \mathbf{g}(\hat{\mathbf{b}}_t, \hat{\mathbf{u}}_t)\ \forall\ t \in \{0,\dots,T-1\}$: This enforces the constraint on the nominal belief dynamics given by Eq. (14). The locally convex approximation of the nonlinear equality constraint is obtained by linearizing around the initialization trajectory $(\bar{\mathcal{B}},\bar{\mathcal{U}})$ as:

$$\mathbf{g}(\hat{\mathbf{b}}_t, \hat{\mathbf{u}}_t) \approx \mathbf{g}(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t) + G_t(\hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t) + H_t(\hat{\mathbf{u}}_t - \bar{\mathbf{u}}_t) \quad (17)$$

$$G_t = \frac{\partial \mathbf{g}}{\partial \hat{\mathbf{b}}}(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t), \quad H_t = \frac{\partial \mathbf{g}}{\partial \hat{\mathbf{u}}}(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t), \qquad (18)$$

where G_t and H_t are the Jacobians of the nominal belief dynamics function evaluated (numerically) at $\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t$.

We also add inequality constraints on the belief states $\hat{\mathbf{b}}_t$ and control inputs $\hat{\mathbf{u}}_t$ of the form $\|\hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t\|_2 < \epsilon_{\mathcal{B}}$ and $\|\hat{\mathbf{u}}_t - \bar{\mathbf{u}}_t\|_2 < \epsilon_{\mathcal{U}}$ to ensure that the optimization progresses only within bounds of the region where the locally convex approximation holds, also known as the *trust region* in optimization literature. The size of the trust region is adjusted within an inner loop based on a line search procedure [25].

3) $\hat{\mathbf{u}}_t \in F_{\mathcal{U}}\ \forall\ t \in \{0,\dots,T-1\}$: This constrains the control input to lie within the set of feasible control inputs $F_{\mathcal{U}}$. For instance, control input limits might correspond to the maximum speed with which the base or links might move. We formulate the feasibility constraint as box inequality constraints $\mathbf{u}_{\min} \leq \hat{\mathbf{u}}_t \leq \mathbf{u}_{\max}$ at each time step t corresponding to the minimum $\mathbf{u}_{\min}$ and maximum $\mathbf{u}_{\max}$ control input bounds, respectively. The inequality constraints are treated as (hard) constraints in the optimization [25].

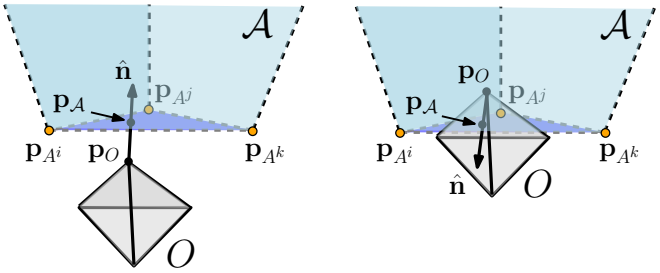
V. Sigma Hulls for Collision Avoidance

For formulating collision avoidance constraints $\mathbf{\Phi}(\hat{\mathcal{B}},\hat{\mathcal{U}},\lambda) \geq \mathbf{0}$ for Gaussian belief spaces, we introduce the notion of *sigma hulls*, which are defined as follows:

Definition: A sigma hull for a robot link is defined as the convex hull of the geometry of the link transformed in joint space according to the UKF sigma points lying on the λ-standard deviation contour of the covariance (Eq. (6)).

Fig. 1 shows the 1-standard deviation sigma hulls of each of the robot links for an articulated robot. It is important that we consider the sigma hulls of the individual robot links and not the robot geometry in its entirety.

We use the signed distance between the sigma hull and objects in the workspace $\mathcal{O}$, as shown in Fig. 1. Informally, the signed distance corresponds to the minimum translation distance required to either put two geometric shapes in contact or separate them if they are overlapping. The distance between two shapes can be calculated by the Gilbert-Johnson-Keerthi (GJK) algorithm [12] and the penetration depth is calculated by the Expanding Polytope Algorithm (EPA) [1]. One useful feature of these two algorithms is that they represent any given convex shape A by its support mapping, i.e., a function that maps a vector $\mathbf{v}$ to a point $\mathbf{p}$ on A that is furthest in direction $\mathbf{v}$:



(a) $\mathtt{sd}(\mathcal{A}, O) > 0$ (b) $\mathtt{sd}(\mathcal{A}, O) < 0$

Fig. 2. Signed distance between a 3D obstacle (gray) O in the workspace and a 3D convex hull $\mathcal{A}$ obtained by transforming the robot link geometry using the UKF sigma points. $\mathbf{p}_{\mathcal{A}}$ and $\mathbf{p}_O$ are the points of closest approach or penetration. Note that the convex hull of the geometry corresponding to the sigma points is never explicitly constructed. Instead, we rely on the concept of support mapping to formulate collision avoidance constraints.

$$\mathtt{support}_A(\mathbf{v}) = \arg\max_{\mathbf{p} \in A} \mathbf{v} \cdot \mathbf{p}. \qquad (19)$$

We consider the sigma hull of each individual link in the robot by transforming the geometry of each link according to the sigma points. [1] For a K link articulated robot and $|\mathcal{O}|$ objects in the workspace, we formulate the collision avoidance constraints for each of the $|\mathcal{O}|K$ link-object pairs at each time step. We ignore self-collisions between robot links or collisions between pairs of objects in the workspace and assume that the objects $\mathcal{O}$ in the workspace are precisely known. We also assume that all the objects $\mathcal{O}$ are convex. In case of non-convex obstacles, we use standard convex decomposition techniques to decompose non-convex geometries into convex geometries.

For a given link A, let $\mathcal{A} = \mathtt{convhull}\{A^0,\dots,A^s\}$ be the sigma hull obtained by transforming the link geometry according to the s UKF sigma points (Eq. (3)), as illustrated in Fig. 2. It turns out that we do not have to explicitly compute a geometrical representation of the convex hull $\mathcal{A}$ to perform the signed distance computation, since the signed distance can be calculated using the support mapping function (Eq. (19)) as:

$$\mathtt{support}_{\mathcal{A}}(\mathbf{v}) = \mathtt{support}_{A^i}(\mathbf{v}), \qquad (20)$$

$$i = \arg\max_{i \in \{0,\dots,s\}} \mathtt{support}_{A^i}(\mathbf{v}) \cdot \mathbf{v}. \qquad (21)$$

We use the above definition to compute the signed distance of $\mathcal{A}$ with respect to objects $\mathcal{O}$ in the workspace by defining

[1] For an articulated robot, we obtain the transformed geometries of all the links in a single pass by starting from the last link that contains the end effector. Since the UKF sigma points corresponding to the last link contain all the DOF of the kinematic chain, we perform all the necessary forward kinematics evaluations only once and cache the sigma shapes for all the remaining links as the computation is being performed for the last link.

the support mapping function for the transformed link geometries $\{A^0, \ldots, A^s\}$. We use efficient implementations of the GJK and EPA algorithms available in the Bullet collision checking library [10] for this purpose.

Fig. 2 shows the signed distance between an object $O \in \mathcal{O}$ and the convex hull $\mathcal{A}$. Two objects are out of collision if the signed distance $\texttt{sd}(\mathcal{A}, O)$ is positive. We enforce the following collision avoidance constraint at each time step:

$$\texttt{sd}(\mathcal{A}, O) \geq d_{\text{safe}} \quad \forall\ O \in \mathcal{O}, \tag{22}$$

where a user-specified safety margin $d_{\text{safe}} > 0$ is used to urge the robot trajectory to stay at least d_{safe} distance from other objects to avoid being in contact (corresponding to $d_{\text{safe}} = 0$).

For the purposes of optimization, the nonlinear constraint in Eq. (22) is approximated by linearized inequality constraints obtained by linearizing around the initialization trajectory $(\bar{\mathcal{B}}, \bar{\mathcal{U}})$. The signed distance $\texttt{sd}(\mathcal{A}, O)$ is a function of the belief state $\hat{\mathbf{b}}_t$ and is denoted as $\texttt{sd}_{\mathcal{A}O}(\hat{\mathbf{b}}_t)$. We denote the closest points on $\mathcal{A}$ and O that realize the signed distance as $\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)$ and $\mathbf{p}_O$, respectively, and the contact normal as the unit vector $\hat{\mathbf{n}}(\hat{\mathbf{b}}_t)$. We approximate the signed distance between $\mathcal{A}$ and O at a belief state $\hat{\mathbf{b}}_t$ in the neighborhood of the current belief $\bar{\mathbf{b}}_t$ as:

$$\texttt{sd}_{\mathcal{A}O}(\hat{\mathbf{b}}_t) \approx \hat{\mathbf{n}}(\bar{\mathbf{b}}_t) \cdot (\mathbf{p}_O - \mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)), \tag{23}$$

$$\hat{\mathbf{n}}(\bar{\mathbf{b}}_t) = \frac{(\mathbf{p}_O - \mathbf{p}_{\mathcal{A}}(\bar{\mathbf{b}}_t))}{\|\mathbf{p}_O - \mathbf{p}_{\mathcal{A}}(\bar{\mathbf{b}}_t)\|_2}, \tag{24}$$

where we assume that the contact normal does not change in the local neighborhood of $\bar{\mathbf{b}}_t$. We now linearize the expression for signed distance $\texttt{sd}_{\mathcal{A}O}(\hat{\mathbf{b}}_t)$ given by Eq. 23 at the current belief $\bar{\mathbf{b}}_t$ as:

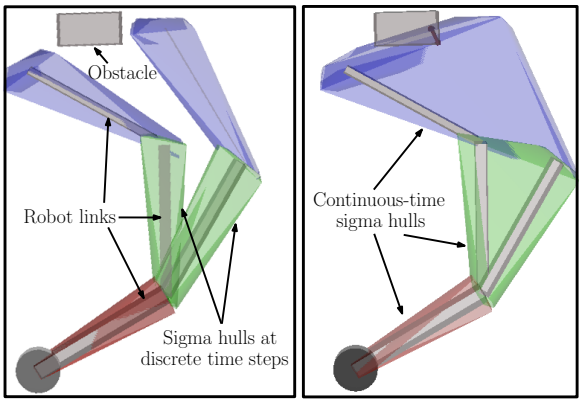
$$\texttt{sd}_{\mathcal{A}O}(\hat{\mathbf{b}}_t) \approx \texttt{sd}_{\mathcal{A}O}(\bar{\mathbf{b}}_t) + S_t(\hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t), \tag{25}$$

$$S_t = \frac{\partial \texttt{sd}_{\mathcal{A}O}}{\partial \hat{\mathbf{b}}}(\bar{\mathbf{b}}_t) \approx -\hat{\mathbf{n}}(\bar{\mathbf{b}}_t)^T \frac{\partial \mathbf{p}_{\mathcal{A}}}{\partial \hat{\mathbf{b}}}(\bar{\mathbf{b}}_t). \tag{26}$$

The closest point $\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)$ lies at a vertex, edge, or face of the convex hull $\mathcal{A}$, the vertices of which come from the transformed link geometries $\{A^0, \ldots, A^s\}$. Since collision checking is expensive even while planning in the state space [25], we compute the derivative S_t analytically instead of relying on expensive numerical derivatives as follows:

We consider the most general case where the closest point $\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)$ lies on a face of the convex hull (as shown in Fig. 2). With small notational modifications, our analysis holds for general simplices but we consider faces to be triangles since triangles meshes are the most popular representations of geometry in collision libraries [10]. Without loss of generality, we assume that face is spanned by three vertices $\mathbf{p}_{A^i}$, $\mathbf{p}_{A^j}$, and $\mathbf{p}_{A^k}$ and that the three vertices come from three distinct instances of geometry A^i, A^j, and A^k corresponding to the i, j, and k^{th} UKF sigma points, respectively. This information is made available to us from the collision library when the signed distance computation is performed. We represent $\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)$ as a barycentric combination of the vertices as:

$$\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t) = \sum_{l \in \{i,j,k\}} \alpha_l \mathbf{p}_{A^l}, \quad \sum_{l \in \{i,j,k\}} \alpha_l = 1, \tag{27}$$



(a) Obstacle does not collide with discrete-time sigma hulls (b) Obstacle overlaps with continuous-time sigma hulls

Fig. 3. (a) Sigma hulls of the robot links at discrete time steps are collision-free but the continuous trajectory between time steps might end up colliding with an obstacle (gray). (b) By considering the convex hull of the sigma hulls between consecutive time steps, we are able to find feasible solutions that avoid obstacles. This reduces the dependence of the feasibility of the solution on the discretization of the trajectory and reduces the overall computational cost of the optimization.

where $\alpha_l, l \in \{i, j, k\}$ are the barycentric coordinates of the point $\mathbf{p}_{\mathcal{A}}(\hat{\mathbf{b}}_t)$. The derivative term from Eq. (26) can now be written as:

$$\frac{\partial \mathbf{p}_{\mathcal{A}}}{\partial \hat{\mathbf{b}}}(\bar{\mathbf{b}}_t) = \sum_{l \in \{i,j,k\}} \alpha_l \frac{\partial \mathbf{p}_{A^l}}{\partial \hat{\mathbf{b}}}(\bar{\mathbf{b}}_t). \tag{28}$$

Each point $\mathbf{p}_{A^l}$ is a function of the l^{th} sigma point given by $\mathbf{p}_{A^l}(\mathcal{X}^l)$. Each $\mathcal{X}^l$ is defined as a linear combination of $\hat{\mathbf{x}}_t$ and a corresponding column of $\sqrt{\hat{\Sigma}_t}$ as given in Eq. (3), which constitute the belief state $\hat{\mathbf{b}}_t$. The derivatives in Eq. (28) can now be evaluated analytically in terms of the position Jacobian matrices $J_{\mathbf{p}_{A^l}}$ that relate the point $\mathbf{p}_{A^l}$ to the current state $\bar{\mathbf{x}}_t$.

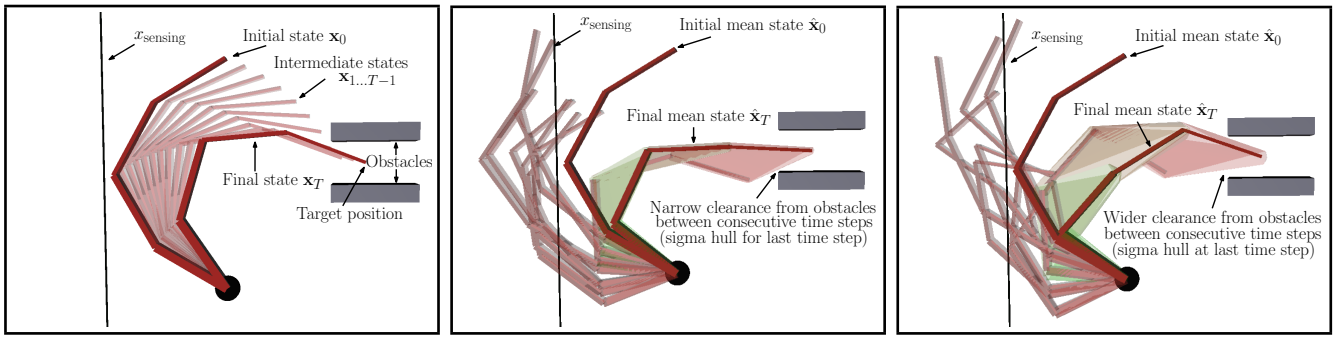
Using Eqs. (22), (25), (26), and (28), we include the collision avoidance constraint in the optimization formulation (Eq. (15)) in terms of the belief state $\hat{\mathbf{b}}_t$ as:

$$\texttt{sd}_{\mathcal{A}O}(\bar{\mathbf{b}}_t) + S_t(\hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t) - d_{\text{safe}} \geq 0,\ \forall\ O \in \mathcal{O}. \tag{29}$$

Continuous-time collision avoidance constraint: The preceding discussion describes how to formulate collision avoidance constraints at a given time step t. We can use this constraint to avoid collisions at each time step of a discretely-sampled trajectory. However, neglecting collisions between time steps can lead to trajectories that collide with obstacles in between time steps. Fig. 3(a) shows an example of sigma hulls at two consecutive time steps. Even though the individual sigma hulls are collision-free, the continuous sigma hulls are in collision (Fig. 3(b)).

Instead of relying on a fine-grained discretization of the trajectory to ensure safety, we modify the collision constraints from the preceding discussion (Eq. (22)) to formulate (probabilistic) continuous-time collision avoidance constraints by considering the convex hull of the individual sigma hulls between consecutive time steps (Fig. 3). The continuous-time collision avoidance constraint now becomes:

$$\texttt{sd}(\texttt{convhull}(\mathcal{A}_t, \mathcal{A}_{t+1}), O) \geq d_{\text{safe}} \quad \forall\ O \in \mathcal{O}. \tag{30}$$



(a) State space trajectory (b) 1-standard deviation belief space trajectory (c) 4-standard deviation belief space trajectory

Fig. 4. A 4-DOF planar robot (red) reaching into a narrow slit (gray). The robot receives noisy measurements of the end-effector position of the robot, the noise being linearly proportional to the x-coordinate of the end-effector position and is minimum at $x = x_{\text{sensing}}$. (a) A locally-optimal state space trajectory [25] produces a trajectory that avoids collisions with obstacles but is unaware of the uncertainty in the robot state. (b) A 1-standard deviation safe belief space trajectory computed using our approach. Notice how the robot moves to bring the x-coordinate of its end-effector closer to x_{sensing} to reliably localize itself, before arriving at the target with reduced uncertainty. However, there is narrow clearance from the obstacles between time steps $t = T-1$ and $t = T$, which results in a higher collision probability during execution (Fig. 5). (c) A 4-standard deviations safe trajectory reaches in by observing a wider clearance to obstacles between consecutive time steps $t = T-1$ and $t = T$, to reduce the probability of collision during execution.

It is important to note that we never explicitly compute the convex hull of the sigma hulls $\mathcal{A}_t$ and $\mathcal{A}_{t+1}$. Instead, we extend the concept of support mapping in Eq. (20) to naturally formulate the continuous-time collision avoidance constraint, where the support mapping computation considers points on both $\mathcal{A}_t$ and $\mathcal{A}_{t+1}$. Also, we have implicitly assumed here that the sigma hulls undergo only translations between time steps. However, the sigma hulls can also undergo rotations between time steps. This can be accounted for by considering an upper bound of the swept volume based on the amount of rotation, as considered in [25].

This extension is able to find feasible solutions in many cases where the discrete collision avoidance fails and converges to a solution that passes through obstacles between time steps. It also comes with a modest performance penalty since we now have to calculate the support mapping for twice as many vertices, which implies that the narrow phase collision detection takes twice as long. However, the continuous collision constraint allows us to consider lesser number of time steps in the initial discretization of the trajectory, thus reducing the overall computational cost of the optimization.

VI. Results

We evaluated our approach in simulation for planning motions for a 4-DOF planar robot and a 7-DOF robot with imprecise actuation and equipped with inaccurate sensing mechanisms. For each of the two robots, the state $\mathbf{x} = (\theta_1, \ldots, \theta_n)$ is a n-D vector (n = 4 or 7 depending on the robot) consisting of the joint angles and the control input $\mathbf{u} = (\omega_1, \ldots, \omega_n)$ is a n-D vector consisting of the angular speeds at each of the joints. The motion noise $\mathbf{q}_t \sim \mathcal{N}(\mathbf{0}, I_{\dim[\mathbf{q}]})$ is scaled by a constant matrix Q. This results in the following stochastic kinematic model:

$$\mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{q}_t) = \begin{bmatrix} \theta_1 + \tau\omega_1 \\ \vdots \\ \theta_n + \tau\omega_n \end{bmatrix} + Q\mathbf{q}_t, \tag{31}$$

where τ is the duration of the time step.

A. 4-DOF planar robot

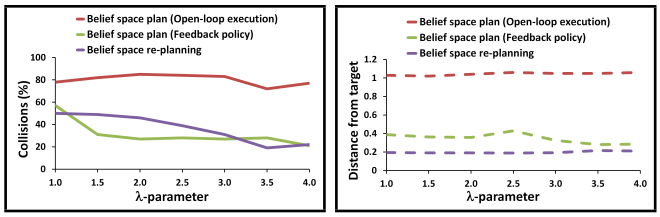
We consider a scenario where the robot obtains noisy measurements about its end-effector position, the noise being linearly proportional to the x-coordinate of the end-effector position and the noise is minimum at $x = x_{\text{sensing}}$, as shown in Fig. 4. This is similar in spirit to the light-dark example considered by Platt et al. [19] where the robot obtains reliable sensing information in a certain region of the environment but the sensor measurements get noisier as the robot moves away from the region. Let $\mathbf{p}_e(\mathbf{x}) = [x_e, y_e]^T$ be the 2-D position of the robot end-effector in the world coordinate frame. The sensing noise $\mathbf{r}_t$ is linearly proportional to $|x_e - x_{\text{sensing}}|$. The observation model is given by:

$$\mathbf{h}(\mathbf{x}_t, \mathbf{r}_t) = \begin{bmatrix} x_e \\ y_e \end{bmatrix} + \mathbf{r}_t. \tag{32}$$

Evaluation: A locally-optimal state space trajectory [25] avoids obstacles (Fig. 4(a)) but is unaware of the uncertainty in the robot state and accumulates considerable uncertainty during execution. Belief space planning improves the input trajectory such that the robot moves to bring the x-coordinate of its end-effector position closer to x_{sensing} to reliably localize itself, before arriving at the target with reduced uncertainty. A lower value of $\lambda = 1$ for planning results in a trajectory that has a narrower margin of clearance from obstacles (Fig. 4(b)), while a larger value for the λ parameter yields trajectories that have a wider clearance from obstacles (Fig. 4(c)), resulting in a safer trajectory.

We evaluated our approach in simulation by executing the trajectories with artificial motion and sensing noise. We considered two metrics to quantify the execution outcome: (i) the number of runs that resulted in collisions with obstacles, which corresponds to the probability of collision during execution, and (ii) the distance between the final end-effector position and the desired target, which corresponds to the task accuracy after execution.

We considered 100 executions of the locally optimal state space trajectory (Fig. 4(a)) [25] in two scenarios: (i) with re-planning in state space starting from the estimated state given



(a) Probability of collision (b) Mean distance from target

Fig. 5. Evaluation of our approach in simulation by considering trajectories obtained by varying the λ parameter and executing with artificial motion and sensing noise. Comparison in terms of (a) the number of trajectory executions that result in a collision with obstacles, and (b) error at final time step from the desired target. Execution of trajectories with re-planning (MPC paradigm) results in the lowest mean distance from the target as compared to open-loop execution or execution with a LQG feedback policy.

by the Kalman filter, and (ii) using a linear feedback policy (linear quadratic Gaussian controller or LQG in this case [2]). With re-planning, the probability of collision is 60% with a mean distance of 0.23 units from the target, while the LQG controller results in a 63% collision probability with a mean distance of 0.3 units from the target.

In contrast, belief space planning computes trajectories that have a significantly lower probability of collision and a lower error from the desired target at the final time step. We compared execution of trajectories computed using our approach in three scenarios: (i) open-loop execution, (ii) linear feedback policy (LQG), and (iii) re-planning (MPC paradigm). Fig. 5 shows the collision probability and mean distance from the target for each of these three scenarios for 100 simulated executions. Execution of trajectories computed using our approach with re-planning has the lowest mean distance from the target and low probability of collision during execution. The probability of collision with re-planning in belief space is 21% for a 4-standard deviation safe trajectory and the mean distance is 0.2 units from the target, which is a considerable improvement over executions of locally-optimal state space trajectories and over open-loop executions of belief space trajectories.

Our unoptimized C++ implementation running on a single 3.2 GHz Intel i7 processor core took 0.65 seconds to plan a locally optimal initial trajectory with 10 time steps for a 14-D belief space, while the total execution time including re-planning was 2.9 seconds for 10 time steps.

B. 7-DOF arm

We consider a scenario where the robot is equipped with an inaccurate proximity sensor mounted at the end-effector. The signal strength measured by the sensor decays quadratically with the distance to a given object. In this scenario, the sensor measures the signal strength in terms of the distance to a wall, which is parallel to the y-z plane and has a known location $x = x_{\text{wall}}$ (Fig. 6(a)). Let $\mathbf{p}_e(\mathbf{x}) = [x_e, y_e, z_e]^T$ be the position of the robot end-effector in the world coordinate frame. The robot also measures the angle at the base using an inaccurate encoder. The sensing noise $\mathbf{r}_t \sim \mathcal{N}(\mathbf{0}, I_{\dim[\mathbf{r}]})$ is scaled by a constant matrix R. The stochastic sensing measurements are then related to the state $\mathbf{x}_t$ as:

$$\mathbf{h}(\mathbf{x}_t, \mathbf{r}_t) = \begin{bmatrix} 1/((x_e - x_{\text{wall}})^2 + 1) \\ \theta_1 \end{bmatrix} + R\mathbf{r}_t. \tag{33}$$

Evaluation: A locally-optimal state space trajectory [25] avoids obstacles (Fig. 6(a)) but is oblivious to the uncertainty in the robot state and might accumulate considerable uncertainty during execution. Belief space planning improves the input trajectory to compute a locally optimal trajectory that is able to safely guide the robot towards the wall for reliable localization before heading back to the target (Fig. 6(b)).

We considered 100 executions of the computed state space trajectory in two scenarios. With re-planning in state space, the probability of collision is 48% with a mean distance of 0.26 units from the target, while execution using a linear feedback policy (LQG) results in a probability of collision of 36% and a mean distance of 0.39 units from the target.

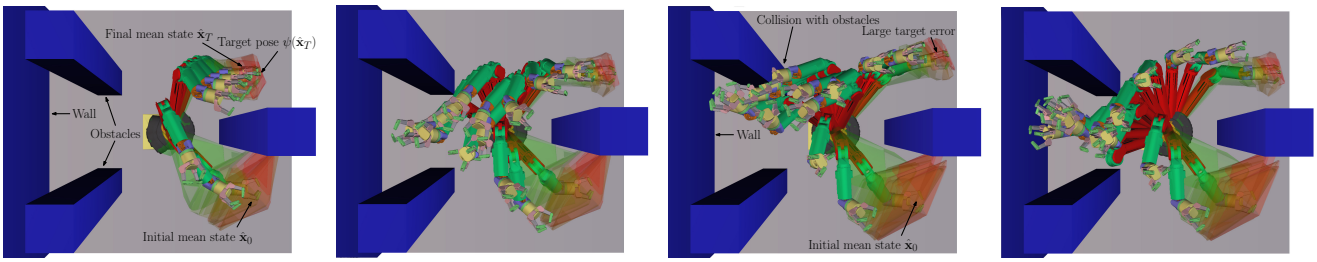
Belief space planning computes trajectories that have a significantly lower probability of collision and a lower error from the desired target at the final time step (Fig. 6(b)). We considered 100 executions of the belief space trajectory with artificial motion and sensing noise. Open-loop execution of locally-optimal belief space trajectories (Fig. 6(c)) results in a probability of collision of 59% and a mean distance of 0.34 units from the target, while execution using a linear feedback policy (LQG) results in a probability of collision of 72% and a mean distance of 0.51 units from the target. The performance of the linear feedback policy on the belief space trajectory is worse since the computed trajectory attempts to enter a narrow passage between obstacles in order to reliably localize itself, which results in a higher probability of collision during execution.

In contrast, execution of the belief space trajectory with re-planning results in a low probability of collision of 20% and a mean distance of 0.19 units from the target, which is lower than executions of state space trajectories or belief space trajectories with a linear feedback policy.

Our implementation took 1.9 seconds to plan a locally optimal state space trajectory with 20 time steps while it took 8.2 seconds to plan a belief space trajectory with 20 time steps for a 35-D belief space. The total execution time including re-planning in belief space was 46.3 seconds. Our experiments suggest that re-planning in belief space can significantly increase the probability of successful task execution with little computational overhead.

VII. Conclusion and Future Work

We presented a novel formulation for collision avoidance for planning in Gaussian belief spaces for articulated robots that are not well approximated as points or spheres. We propose the use of sigma hulls, which are convex hulls of the robot geometry transformed according to the sigma points generated by the Unscented Kalman filter (UKF). We use efficient formulations of the support mapping of convex shapes to formulate collision avoidance constraints in terms of the signed distance between the sigma hulls and objects in the workspace within an optimization-based planning framework. Our experiments suggest that re-planning in belief space can considerably reduce the probability of collision and error from desired target during execution.



(a) State space trajectory (b) Belief space trajectory (c) Open-loop execution (d) Execution with re-planning

Fig. 6. Simulated trajectory traces for a 7-DOF articulated robot moving in a constrained environment with obstacles (blue). The robot localizes itself based solely on distance of the robot end-effector from a wall in the environment, with the signal strength decaying quadratically with the distance. We visualize the uncertainty at the initial state and target in terms of the sigma hulls (rendered translucent). (a) A locally-optimal state space trajectory [25] produces a trajectory that avoids collisions with obstacles but is unaware of the uncertainty in the robot state. (b) A locally-optimal belief space trajectory. Notice how the robot moves closer to the wall through a narrow passage to reliably localize itself before moving to the desired target. (c) Open-loop execution of the belief space trajectory leads to collisions with objects in the workspace because the optimal trajectory is computed assuming deterministic belief dynamics (Eq. (14)). The end-effector is also farther away from the desired target. (d) We initialize the optimization with the previously computed solution and the current belief state to re-plan after every time step. A resultant execution of the trajectory that successfully avoids collisions is shown. The robot moves the end-effector closer to the wall for an extended period of time for reliable localization before safely heading back to the target.

Our work opens up several avenues for future research. Instead of λ being a user-specified parameter, one potential extension could be to incorporate λ in the optimization and search over λ values for an optimal solution that is both safe and minimizes the cost function encoding the task objective. Our Gaussian parameterization of the belief state might not be applicable for some applications, for instance ones where multi-modal beliefs are expected to appear. Another extension is to consider self-collisions between the robot links and collisions with other uncertain objects in the workspace. Finally, we plan to apply this approach to plan motions for low-cost robot platforms such as the Raven surgical robot and the Baxter manufacturing robot.

REFERENCES

[1] G. V. D. Bergen, "Proximity Queries and Penetration Depth Computation on 3D Game Objects," in *Game Developers Conference (GDC)*, 2001.

[2] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 1995, vol. 1, no. 2.

[3] J. T. Betts, *Practical Methods for Optimal Control and Estimation using Nonlinear Programming*. Society for Industrial & Applied Mathematics, 2010, vol. 19.

[4] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[5] A. Bry and N. Roy, "Rapidly-exploring Random Belief Trees for Motion Planning Under Uncertainty," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2011, pp. 723–730.

[6] J. T. C. Papadimitriou, "The Complexity of Markov Decision Processes," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, 1987.

[7] E. F. Camacho and C. Bordons, *Model Predictive Control*. London, UK: Springer Verlag, 2004.

[8] J. Canny, *Complexity of Robot Motion Planning*. MIT press, 1988.

[9] A. D. Christiansen and K. Goldberg, "Comparing Two Algorithms for Automatic Planning by Robots in Stochastic Environments," *Robotica*, vol. 13, no. 6, pp. 565–574, 1995.

[10] E. Coumans, "Bullet Collision Detection and Physics Library," Available: http://bulletphysics.org, 2013.

[11] T. Erez and W. D. Smart, "A Scalable Method for Solving High-Dimensional Continuous POMDPs Using Local Approximation," in *Conf. on Uncertainty in Artificial Intelligence*, 2010, pp. 160–167.

[12] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A Fast Procedure for Computing the Distance between Complex Objects in Three-Dimensional Space," *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.

[13] K. Hauser, "Randomized Belief-Space Replanning in Partially Observable Continuous Spaces," *Algorithmic Foundations of Robotics IX*, pp. 193–209, 2011.

[14] S. J. Julier and J. K. Uhlmann, "New Extension of the Kalman Filter to Nonlinear Systems," in *AeroSense 1997*, 1997, pp. 182–193.

[15] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and Acting in Partially Observable Stochastic Domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[16] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient Point-based POMDP Planning by Approximating Optimally Reachable Belief Spaces," in *Robotics: Science and Systems (RSS)*, 2008.

[17] A. Lambert, D. Gruyer, and G. St Pierre, "A Fast Monte Carlo Algorithm for Collision Probability Estimation," in *Intl. Conf. on Control, Automation, Robotics and Vision*, 2008, pp. 406–411.

[18] R. Platt, L. Kaelbling, T. Lozano-Perez, and R. Tedrake, "Efficient Planning in Non-Gaussian Belief Spaces and its Application to Robot Grasping," in *Int. Symp. on Robotics Research (ISRR)*, 2011.

[19] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez, "Belief Space Planning assuming Maximum Likelihood Observations," in *Robotics: Science and Systems (RSS)*, 2010.

[20] S. Prentice and N. Roy, "The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance," *Int. Journal of Robotics Research*, vol. 28, no. 11–12, pp. 1448–1465, 2009.

[21] M. Quigley, R. Brewer, S. P. Soundararaj, V. Pradeep, Q. Le, and A. Y. Ng, "Low-cost Accelerometers for Robotic Manipulator Perception," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010, pp. 6168–6174.

[22] Rethink Robotics, "Baxter manufacturing robot," Available: http://www.rethinkrobotics.com/index.php/products/baxter/, 2012.

[23] J. Rosen, M. Lum, M. Sinanan, and B. Hannaford, "Raven: Developing a Surgical Robot from a Concept to a Transatlantic Teleoperation Experiment," in *Surgical Robotics: System Applications and Visions*. Springer, 2011, ch. 8, pp. 159–197.

[24] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

[25] J. Schulman, J. Ho, A. Lee, H. Bradlow, I. Awwal, and P. Abbeel, "Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization," in *Robotics: Science and Systems (RSS)*, 2013.

[26] S. Thrun, "Monte Carlo POMDPs," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1064–1070, 2000.

[27] J. van den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized Path Planning for Robots with Motion Uncertainty and Imperfect State Information," *Int. Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[28] J. van den Berg, S. Patil, and R. Alterovitz, "Motion Planning under Uncertainty using Iterative Local Optimization in Belief Space," *Int. Journal of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012.

[29] M. P. Vitus and C. J. Tomlin, "Closed-Loop Belief Space Planning for Linear, Gaussian Systems," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2011, pp. 2152–2159.

[30] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Univ. North Carolina at Chapel Hill, Tech. Rep. TR 95-041, 2006.