Fast and Bounded Probabilistic Collision Detection for High-DOF Robots in Dynamic Environments

Chonhyon Park, Jae Sung Park, and Dinesh Manocha

University of North Carolina, Chapel Hill, NC 27599, USA,
{chpark,jaesungp,dm}@cs.unc.edu
http://gamma.cs.unc.edu/PCOLLISION

Abstract. We present a novel approach to performing probabilistic collision detection between a high-DOF robot and imperfect obstacle representations in dynamic, uncertain environments. These uncertainties are modeled using Gaussian distributions. We present an efficient algorithm for bounded collision probability approximation. We use our probabilistic collision algorithm for trajectory optimization in dynamic scenes for 7-DOF robots. We highlight its performance in challenging simulated and real-world environments with robot arms operating next to dynamically moving human obstacles.

1 Introduction

Robots are increasingly being used in living spaces, factories, and outdoor environments. In such environments, various elements of or parts of the robot tend to be in close proximity to humans or other moving objects. This proximity gives rise to two kinds of challenges in terms of motion planning. First, we have to predict the future actions and reactions of moving obstacles or agents in the environment to avoid collisions with the obstacles. Therefore, the collision avoidance algorithm needs to deal with uncertain and imperfect representations of obstacle motions. Second, the computed robot motion still needs to be reasonably efficient and all such collision computations have to be performed at almost realtime rates.

Various uncertainties arise from control errors, sensing errors, or environmental errors (i.e. imperfect environment representation) in the estimation and prediction of environment obstacles. Typically, these uncertainties are modeled using Gaussian distributions. In this paper, we limit ourselves to environmental errors. Current motion planning algorithms use probabilistic collision detection algorithms to compute appropriate trajectories with imperfect obstacle representations. Typically, these uncertainties are modeled using Gaussian distributions and stochastic algorithms are used to approximate the collision probability [4, 12, 16, 17]. However, it is almost impossible to guarantee a perfect collision-free trajectory as the Gaussian distributions corresponding to the obstacle positions have non-zero probabilities in the entire workspace.

Many of the stochastic algorithms used to approximate the collision probability [4, 12] tend to be computationally expensive or limited to 2D workspaces. Most prior planning approaches for high-DOF robots perform exact collision checking with scaled objects that enclose the potential object volumes [3, 5, 13, 24]. Although these planning approaches can guarantee probabilistic safety bounds, they tend to overestimate the collision probability. This overestimation can either result in less optimal trajectories or may fail to compute a feasible trajectory in the limited planning time in dynamic scenes. Therefore, it is desirable to balance the safety and efficiency in terms of the planned trajectory.

Main Results: In this paper, we present a novel approach to perform probabilistic collision detection. In particular, we present an algorithm for fast approximation of collision probability between the high-DOF robot and obstacles. Our approach computes more accurate probabilities as compared to prior approaches that perform exact collision checking with enlarged obstacle shapes. Moreover, we can guarantee that our computed probability is an upper bound on the actual probability. Moreover, we present a trajectory optimization algorithm for high-DOF robots in dynamic, uncertain environments based on our probabilistic collision detection, and a practical belief space estimation algorithm that accounts for both spatial and temporal uncertainties in the position and motion of each obstacle in dynamic environments. We have evaluated our planner using 7-DOF robot arms operating in a simulation and a real workspace environment with high-resolution point cloud data corresponding to moving human obstacles, captured using a Kinect device[1].

The paper is organized as follows. Section 2 gives a brief overview of prior work on probabilistic collision detection and motion planning. We introduce the notation and describe our probabilistic collision detection algorithm in Section 3. We describe the trajectory planning algorithm in Section 4. We highlight planning performance in challenging human environment scenarios in Section 5.

2 Related Work

In this section, we give a brief overview of prior work on probabilistic collision detection and trajectory planning in dynamic environments.

2.1 Probabilistic Collision Detection

Collision detection is an integral part of any motion planning algorithm and most prior techniques assume an exact representation of the robot and obstacles. Given some uncertainty or imperfect representation of the obstacles, certain algorithms perform probabilistic collision detection. Typically, these uncertainties are modeled using Gaussian distributions, and stochastic techniques are used to approximate the collision probability [4, 9, 12]. In stochastic algorithms, a large number of sample evaluations is required to compute an accurate collision probability.

[1] https://developer.microsoft.com/en-us/windows/kinect

If it can be assumed that the sizes of the objects are small, the collision probability between objects can be approximated using the probability at a single configuration and corresponds to the mean of the probability distribution function (PDF) [6]. This approximation is fast, but the computed probability cannot provide a bound; i.e. it can be higher or lower than the actual collision probability, and the error increases as the object becomes larger.

For high-dimensional spaces, a common approach to check collisions for imperfect or noisy objects is to perform exact collision checking with a large volume that encloses the object poses [3, 19]. Prior approaches generally enlarge an object shape, which may correspond to a robot or an obstacle, to compute the space occupied by the object for a given standard deviation. This may correspond to a sphere [5] or a sigma hull [13]. These approaches tend to compute a bounding volume for the given confidence level. However, the computed volume overestimates the probability and can be much bigger than the actual volume corresponding to the confidence level. Therefore, these approaches can result in a failure to find existing feasible trajectories for motion planning.

Other approaches have been proposed to perform probabilistic collision detection on point cloud data. Bae et al. [1] presented a closed-form expression for the positional uncertainty of point clouds. Pan et al. [16] reformulated the probabilistic collision detection problem as a classification problem and computed per point collision probability. However, these approaches assume that the environment is mostly static. Other techniques are based on broad phase data structures that handle large point clouds for realtime collision detection [17].

2.2 Planning in Dynamic and Uncertain Environments

There is considerable literature on motion planning in dynamic scenes. In many scenarios, the future positions of the obstacles are not known. As a result, the planning problem is typically solved using replanning algorithms, which interleave planning with execution. These methods include sampling-based planners [22], grid searches [15], and trajectory optimization [19].

Applications that require high responsiveness use control-based approaches [11], which can compute trajectories in realtime. They compute the robot trajectory in the workspace of the robot, according to the sensor data. However, the mapping from the Cartesian trajectory to the trajectory in the configuration space of high-DOF robots can be problematic as there can be multiple configurations for a single pose defined in the Cartesian workspace. Furthermore, control-based approaches tend to compute less optimal robot trajectories as compared to the planning approaches that incorporate the estimation of the future obstacle poses. Planning algorithms can compute better robot trajectories in applications in which a good prediction about obstacle motions in a short horizon can be obtained.

The unknown future obstacle positions are one of the source of uncertainties. POMDPs (partially-observable Markov decision processes) provide a mathematically rigorous and general approach for planning under uncertainty [10]. They handle the uncertainty by reasoning over the *belief* space. However, the POMDP formulation is regarded as computationally intractable [18] for problems that are high-dimensional or have a large number of actions. Many efficient approximations use Gaussian belief spaces, which are estimated using Bayesian filters (e.g., Kalman filters) [14, 23]. Gaussian belief spaces have also been used for the motion planning of high-DOF robots [3, 24], but most planning algorithms do not account for environment uncertainty or imperfect obstacle information. Under the conditions of dynamic environments, planning with uncertainty algorithms are mainly limited to 2D spaces [2, 7].

3 Probabilistic Collision Detection for High-DOF Robots

In this section, we first introduce the notation and terminology used in the paper; then we present our probabilistic collision detection algorithm for detecting collisions between a high-DOF robot and the dynamic environment.

3.1 Notation and Assumptions

Our goal is to compute the collision probability between a high-DOF robot configuration and a given obstacle representation of dynamic environments, where the obstacle representation is a probability distribution that accounts for uncertainties in the obstacle motion.

For an articulated robot with D one-dimensional joints, we represent a single robot configuration as $\mathbf{q}$, which is a vector composed of the joint values. The D-dimensional vector space of $\mathbf{q}$ is the configuration space $\mathcal{C}$ of the robot. We denote the collision-free subset of $\mathcal{C}$ as $\mathcal{C}_{free}$, and the other configurations corresponding to collisions as $\mathcal{C}_{obs}$.

We assume that the robot consists of J links $R_1, ..., R_J$, where $J \leq D$. Furthermore, for each robot link R_j, we use a sequence of bounding volumes $B_{j1}, ..., B_{jK}$ to tightly enclose $R_j(\mathbf{q})$, which corresponds to a robot configuration $\mathbf{q}$, i.e.,

$$\forall j : R_j(\mathbf{q}) \subset \bigcup_{k=1}^{K} B_{jk}(\mathbf{q}) \text{ for } (1 \leq j \leq J). \tag{1}$$

We denote obstacles in the environment as O_l $(1 \leq l \leq L)$. The configuration of these obstacles is specified based on their poses. As is the case for the robot, we use the bounding volumes $S_{l1}, ..., S_{lM}$ to enclose each obstacle O_l in the environment:

$$\forall l : O_l \subset \bigcup_{m=1}^{M} S_{lm} \text{ for } (1 \leq l \leq L). \tag{2}$$

For dynamic obstacles, we assume the predicted pose of a bounding volume S_{lm} at time t is estimated as a Gaussian distribution $\mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$.

3.2 Fast and Bounded Collision Probability Approximation

The collision probability between a robot configuration $\mathbf{q}_i$ with the environment at time t_i, $P(\mathbf{q}_i \in \mathcal{C}_{obs}(t_i))$ can be evaluated by checking their bounding volumes for possible overlaps, which can be formulated as

$$P(\mathbf{q}_i \in \mathcal{C}_{obs}(t_i)) = P\left(\left(\bigcup_j \bigcup_k B_{jk}(\mathbf{q}_i)\right) \bigcap \left(\bigcup_l \bigcup_m S_{lm}(t_i)\right) \neq \emptyset\right). \quad (3)$$

We assume the robot links R_j and obstacles O_l are independent of each other, as their poses depend on corresponding joint values or obstacle states. Then (3) can be computed as

$$P(\mathbf{q}_i \in \mathcal{C}_{obs}(t_i)) = 1 - \prod_j \prod_l \overline{P_{col}(i,j,l)}, \quad (4)$$

where $P_{col}(i,j,l)$ is the collision probability between $R_j(\mathbf{q}_i)$ and $O_l(t_i)$. Because poses of bounding volumes B_{jk} and S_{lm} are determined by joint values or obstacle states of the corresponding robot link or obstacle, bounding volumes for the same object are dependent on each other, and $P_{col}(i,j,l)$ can be approximated as

$$P_{col}(i,j,l) \approx \max_{k,m} P_{col}(i,j,k,l,m) \quad (5)$$

$$P_{col}(i,j,k,l,m) = P(B_{jk}(\mathbf{q}_i) \cap S_{lm}(t_i) \neq \emptyset), \quad (6)$$

where $P_{col}(i,j,k,l,m)$ denotes the collision probability between $B_{jk}(\mathbf{q}_i)$ and $S_{lm}(t_i)$.

Fig. 1 illustrates how $P_{col}(i,j,k,l,m)$ can be computed for $S_{lm}(t_i) \sim \mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$. We assume that the robot's bounding volume $B_{jk}(\mathbf{q}_i)$ is a sphere centered at $\mathbf{o}_{jk}(t_i)$, similar to the environment bounding volume S_{lm}, and denote the radii of B_{jk} and S_{lm} as r_1 and r_2, respectively. We assume radii r_1 and r_2 are constants. Then the exact probability of collision between them is given as:

$$P_{col}(i,j,k,l,m) = \int_{\mathbf{x}} I(\mathbf{x}, \mathbf{o}_{jk}(t_i)) p(\mathbf{x}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) d\mathbf{x}, \quad (7)$$

where the indicator function $I(\mathbf{x}, \mathbf{o})$ and the obstacle function $p(\mathbf{x}, \mathbf{p}, \mathbf{\Sigma})$ are defined as,

$$I(\mathbf{x}, \mathbf{o}) = \begin{cases} 1 \text{ if } \|\mathbf{x} - \mathbf{o}\| \leq (r_1 + r_2) \\ 0 \qquad \text{otherwise} \end{cases} \text{ and} \quad (8)$$

$$p(\mathbf{x}, \mathbf{p}, \mathbf{\Sigma}) = \frac{e^{-0.5(\mathbf{x}-\mathbf{p})^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{p})}}{\sqrt{(2\pi)^3 \|\mathbf{\Sigma}\|}}, \quad (9)$$

respectively. It is known that there is no closed form solution for the integral given in (7).

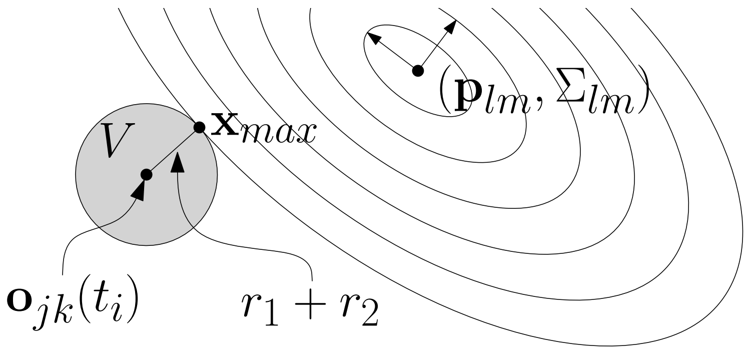


Fig. 1: Approximation of probabilistic collision detection between a sphere obstacle of radius r_2 with a probability distribution $\mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$ and a rigid sphere robot bounding volume $B_{jk}(\mathbf{q}_i)$ centered at $\mathbf{o}_{jk}(t_i)$ with radius r_1 for a robot configuration $\mathbf{q}_i$. The collision probability is approximated as $V \cdot \mathbf{x}_{max}$, where V is the volume of the sphere with the radius computed as the sum of two radii, $V = \frac{4\pi}{3}(r_1 + r_2)^3$, and $\mathbf{x}_{max}$ is the position which has the maximum probability of $\mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$.

Du Toit and Burdick approximate (7) as $V \cdot p(\mathbf{o}_{jk}(t_i), \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$, where V is the volume of the sphere, i.e., $V = \frac{4\pi}{3}(r_1 + r_2)^3$ [6]. However, this approximated probability can be either smaller or larger than the exact probability. If the covariance $\mathbf{\Sigma}_{lm}$ is small, the approximated probability can be much smaller than the exact probability. Planners using this approximation may underestimate the collision probability and may compute unsafe robot motion.

In order to avoid this problem, we compute $\mathbf{x}_{max}$, the position that has the maximum probability of $\mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$ in $\mathbf{B}_{jk}(\mathbf{q}_i)$, and compute the upper bound of $P_{col}(i, j, k, l, m)$ as

$$P_{approx}(i, j, k, l, m) = V \cdot p(\mathbf{x}_{max}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}). \tag{10}$$

Although $\mathbf{x}_{max}$ has no closed-form solution, it can be computed efficiently using numerical techniques.

Lemma 1. $\mathbf{x}_{max}$*, the position has the maximum probability of* $\mathcal{N}(\mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$ *in* $\mathbf{B}_{jk}(\mathbf{q}_i)$*, is formulated as a one-dimensional search of a parameter* λ*,*

$$\mathbf{x}_{max} = \{\mathbf{x} | \|\mathbf{x} - \mathbf{o}_{jk}(t_i)\| = (r_1 + r_2) \; and \, \mathbf{x} \in \mathbf{x}(\lambda)\}, \; where \tag{11}$$

$$\mathbf{x}(\lambda) = (\mathbf{\Sigma}_{lm}^{-1} + \lambda \mathbf{I})^{-1}(\mathbf{\Sigma}_{lm}^{-1}\mathbf{p}_{lm} + \lambda \mathbf{o}_{jk}(t_i)). \tag{12}$$

Proof. The problem of finding the position with the maximum probability in a convex region can be formulated as an optimization problem with a Lagrange multiplier λ [8],

$$\mathbf{x}_{max} = \arg\min_{\mathbf{x}} \left\{ (\mathbf{x} - \mathbf{p}_{lm})^T \mathbf{\Sigma}_{lm}^{-1} (\mathbf{x} - \mathbf{p}_{lm}) + \lambda (\mathbf{x} - \mathbf{o}_{jk})^2 \right\}. \tag{13}$$

The solution of (13) satisfies

$$\nabla \left\{ (\mathbf{x} - \mathbf{p}_{lm})^T \mathbf{\Sigma}_{lm}^{-1} (\mathbf{x} - \mathbf{p}_{lm}) + \lambda (\mathbf{x} - \mathbf{o}_{jk})^2 \right\} = 0, \tag{14}$$

and can be computed as

$$2\mathbf{\Sigma}_{lm}^{-1}(\mathbf{x} - \mathbf{p}_{lm}) + 2\lambda(\mathbf{x} - \mathbf{o}_{jk}) = 0 \tag{15}$$
$$\mathbf{x} = (\mathbf{\Sigma}_{lm}^{-1} + \lambda \mathbf{I})^{-1})(\mathbf{\Sigma}_{lm}^{-1}\mathbf{p}_{lm} + \lambda \mathbf{o}_{jk}). \tag{16}$$

The approximated probability (10) is guaranteed as an upper bound of the exact collision probability (7).

Theorem 1. *The approximated probability $P_{approx}(i,j,k,l,m)$ (10) is always greater than or equal to the exact collision probability $P_{col}(i,j,k,l,m)$ (7).*

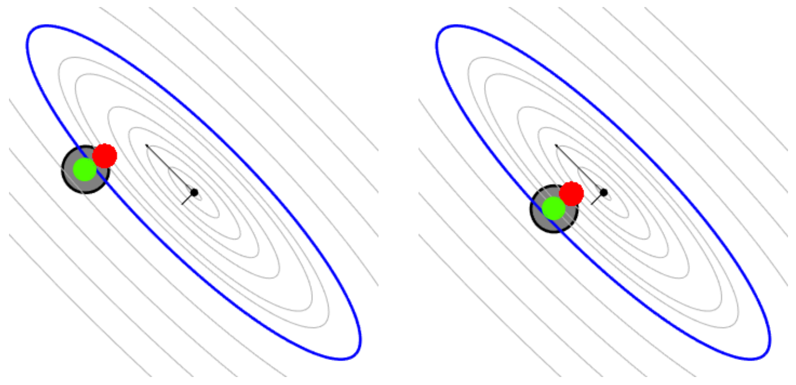
Proof. $p(\mathbf{x}_{max}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) \geq p(\mathbf{x}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm})$ for $\{\mathbf{x} | \|\mathbf{x} - \mathbf{o}_{jk}(t_i)\| \leq (r_1 + r_2)\}$ from Lemma 1. Therefore,

$$\begin{aligned}
P_{approx}(i,j,k,l,m) &= V \cdot p(\mathbf{x}_{max}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) && (17)\\
&= \int_{\mathbf{x}} I(\mathbf{x}, \mathbf{o}_{jk}(t_i)) d\mathbf{x} \cdot p(\mathbf{x}_{max}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) && (18)\\
&= \int_{\mathbf{x}} I(\mathbf{x}, \mathbf{o}_{jk}(t_i)) \cdot p(\mathbf{x}_{max}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) d\mathbf{x} && (19)\\
&\geq \int_{\mathbf{x}} I(\mathbf{x}, \mathbf{o}_{jk}(t_i)) \cdot p(\mathbf{x}, \mathbf{p}_{lm}, \mathbf{\Sigma}_{lm}) d\mathbf{x} && (20)\\
&= P_{col}(i,j,k,l,m). && (21)
\end{aligned}$$

3.3 Comparisons with Other Algorithms

In Fig. 2, we illustrate two cases of the collision probability computation between a circle B (in gray), and a point (in black) with uncertainties, $\mathbf{x} \sim (\mathbf{p}, \mathbf{\Sigma})$, in 2D. We evaluate the exact collision probabilities using the numerical integration of the PDF. The collision probability of Case I is 0.09%, which is feasible with $\delta_{CL} = 0.99$, while the probability of Case II is 1.72%, which is infeasible. The contours in Fig. 2 represent the bounds for different confidence levels. Approaches that use the exact collision checking with enlarged bounding volumes [3, 19] for a given confidence level (e.g., the blue ellipse for $\delta_{CL} = 0.99$) determine both Case I and Case II have collisions and infeasible, i.e., the collision probability is 100%, while the collision probability for Case I is only 0.09%.

Du Toit and Burdick [6] use the probability of the center point (shown in green in Fig. 2) to approximate the collision probability, as described in Section 3.2. However, their approach cannot guarantee upper bounds, and the approximated probability can be significantly smaller than the exact probability if the covariance value is small. Case II in Fig. 2 shows that the approximated probability of their approach is 0.89%, and that satisfies the safety with $\delta_{CL} = 0.99$ and determines Case II as a feasible configuration, which is not true for the exact probability 1.72%.



(a) Case I (b) Case II

Algorithms	Collision probability (O : feasible, X : infeasble)	
	Case I	Case II
Exact probability	0.09%(O)	1.72%(X)
Enlarged bounding volumes [3, 19]	100.00%(X)	100.00%(X)
Approximation using the center point PDF [6]	0.02%(O)	0.89%(O)
Our approach	0.80%(O)	8.47%(X)

Fig. 2: **Comparison of approximated collision probabilities for feasible ($P(\mathbf{x}) \leq 1 - \delta_{CL}$) and infeasible ($P(\mathbf{x}) > 1 - \delta_{CL}$) scenarios for $\delta_{CL} = 0.99$:** We compare the exact collision probability (computed using numerical integration) with approximated probabilities of 1) enlarged bounding volumes (blue contour) [3, 19], 2) approximation using object center point (in green) [6], and 3) our approach that uses the maximum probability point (in red). Our approach guarantees that we do not underestimate the probability, while our approximated probability is close to the exact probability.

Unlike [6], we approximate the probability of the entire volume using the maximum probability value of a single point (shown in red in Fig. 2), as described in Section 3.2. Our approach guarantees computation of the upper bound of collision probability, while the approximated probability is closer to the exact probability than of the enlarged bounding volume approaches.

4 Trajectory Optimization using Probabilistic Collision Detection

In this section, we present our trajectory optimization approach which uses the probabilistic collision detection to avoid collision in the dynamic environment.

We define the time-space domain $\mathcal{X}$, which adds a time dimension to the configuration space, i.e., $\mathcal{X} = \mathcal{C} \times T$. The robot's trajectory, $\mathbf{q}(t)$, is represented as a function of time from the start configuration $\mathbf{q}_s$ to the goal configuration

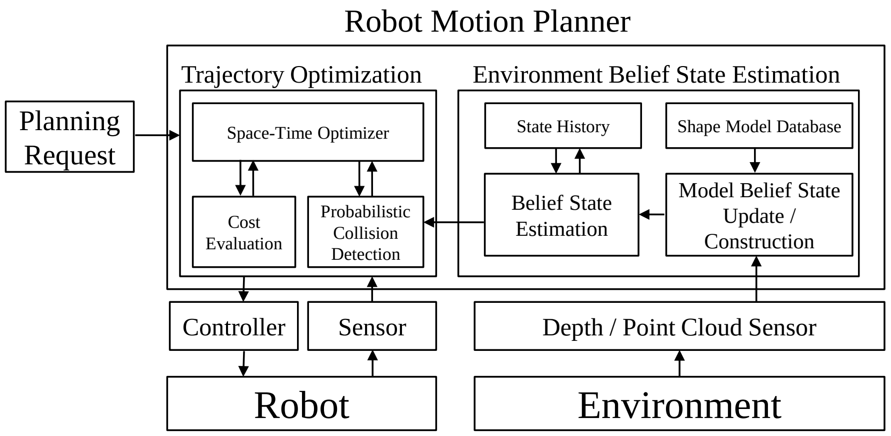


Fig. 3: **Trajectory Planning:** We highlight various components of our algorithm. These include belief space estimation of environment (described in [20]), probabilistic collision checking (described in Section 3), and trajectory optimization.

$\mathbf{q}_g$. It is represented using the matrix $\mathbf{Q}$,

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_s \ \mathbf{q}_1 \ ... \ \mathbf{q}_{n-1} \ \mathbf{q}_g \\ t_0 \ \ t_1 \ ... \ t_{n-1} \ \ t_n \end{bmatrix}, \tag{22}$$

which corresponds to $n+1$ configurations at discretized keyframes, $t_i = i\Delta_T$, which have a fixed interval Δ_T. We denote the i-th column of $\mathbf{Q}$ as $\mathbf{x}_i = \begin{bmatrix}\mathbf{q}_i^T \ t_i\end{bmatrix}^T$.

Fig. 3 highlights various components of our planning algorithm. The pseudocode description is given in Algorithm 1 for a single planning step. Our overall trajectory planning algorithm consists of two main components: environment belief state estimation and trajectory optimization.

We update the belief state of the environment $\mathbf{b} = (\mathbf{p}, \Sigma)$ using means and covariances $\mathbf{p}_{lm}$ and Σ_{lm} of the poses of the existing bounding volumes S_{lm}. That is, $\mathbf{p} = \begin{bmatrix}\mathbf{p}_{11}^T \ ... \ \mathbf{p}_{LM}^T\end{bmatrix}^T$ and $\Sigma = \mathrm{diag}(\Sigma_{11}, ..., \Sigma_{LM})$, where Σ is a block diagonal matrix of the covariances. Moreover, we use a Bayesian estimator to predict the future belief state of the environment, which is used for probabilistic collision detection.

We use incremental trajectory optimization, which repeatedly refines a motion trajectory using an optimization formulation [19]. The planner initializes the robot trajectory $\mathbf{Q}$ as a smooth trajectory of predefined length T between $\mathbf{q}_s$ and $\mathbf{q}_g$, and refines it in every planning step ΔT.

We define the collision avoidance constraint based on the following probability computation formulation:

$$\forall \mathbf{x}_i : P(\mathbf{q}_i \in \mathcal{C}_{obs}(t_i)) < 1 - \delta_{CL}. \tag{23}$$

Algorithm 1 $\mathbf{Q}^* =$PlanWithEnvUncertainty($\mathbf{Q}$, $\{\mathbf{d}_k\}, t_i$)
: Compute the optimal robot trajectory $\mathbf{Q}^*$ during the planning step $\varDelta T$ for the environment point clouds $\{\mathbf{d}\}$ at time t_i

```
Input: initial trajectory $\mathbf{Q}$, environment point clouds $\{\mathbf{d}\}$, time $t_i$
Output: Optimal robot trajectory $\mathbf{Q}^*$ for time step $\varDelta T$
1: $\mathbf{p}_i$ = EnvironmentStateComputation($\{\mathbf{d}\}$) // compute the environment state
2: for $k \in \{i, ..., i + \varDelta T\}$ do
3:     $\mathbf{B}_k$ = BeliefStateEstimation($\mathbf{B}_0, ..., \mathbf{B}_{k-1}, \mathbf{p}_i$) //estimate the current and future
       belief states
4: end for
5: while elapsed time < $\varDelta T$ do
6:     $P$=ProbCollisionChecking($\mathbf{Q}, \{\mathbf{B}_i, ..., \mathbf{B}_{i+\varDelta T}\}$) // perform probabilistic collision
       detection
7:     $\mathbf{Q}^*$=Optimize($\mathbf{Q}, P$) // compute the optimal trajectory for high-DOF robot
8: end while
```

We can compute $P(\mathbf{q}_i \in \mathcal{C}_{obs}(t_i))$ using (4) in Section 3. The computed trajectories that satisfy (23) guarantee that the probability of collision with the obstacles is bounded by the confidence level δ_{CL}, i.e. the probability that a computed trajectory has no collision is higher than δ_{CL}. Use of a higher confidence level computes safer, but more conservative trajectories. The use of a lower confidence level increases the success rate of planning, but also increases the probability of collision.

The objective function for trajectory optimization at time t_k can be expressed as the sum of trajectory smoothness cost, and collision constraint costs for dynamic uncertain obstacles and static known obstacles,

$$\begin{aligned} f(\mathbf{Q}) = \min_{Q} & \sum_{i=k+m}^{n} \left(\|\mathbf{q}_{i-1} - 2\mathbf{q}_i + \mathbf{q}_{i+1}\|^2 + C_{static}(\mathbf{Q}_i)\right) \\ & + \sum_{i=k+m}^{k+2m} \max(P(\mathbf{q}_i \in \mathcal{C}_{obs}(\mathbf{x}_i)) - (1 - \delta_{CL}), 0), \end{aligned} \tag{24}$$

where m is the number of time steps in a planning time step $\varDelta T$.

Unlike the previous optimization-based planning approaches [19, 25] which maintain and cannot change the predefined trajectory duration for the computed trajectory, we adjust the duration of trajectory T to avoid collisions with the dynamic obstacles. When the trajectory planning starts from $\mathbf{t}_i$ ($\mathbf{t}_i$ can be different from $\mathbf{t}_s$ due to replanning) and if the computed trajectory $\mathbf{Q}$ violates the collision probability constraint (23) at time t_j, i.e., $P(\mathbf{q}_j \in \mathcal{C}_{obs}(t_j)) \geq \delta_{CL}$, we repeatedly add a new time step $\mathbf{x}_{new}$ before $\mathbf{x}_j$ and rescale the trajectory from $[\mathbf{t}_i, ..., \mathbf{t}_{j-1}]$ to $[\mathbf{t}_i, ..., \mathbf{t}_{j-1}, \mathbf{t}_{new}]$, until $\mathbf{x}_{new}$ is collision-free. Then, the next planning step starts from $\mathbf{x}_{new}$. It allows the planner to slow the robot down when it cannot find a safe trajectory for the previous trajectory duration due to the dynamic obstacles. If the optimization algorithm converges, our algorithm computes the optimal trajectory,

$$\mathbf{Q}^* = \arg\min_{\mathbf{Q}} f(\mathbf{Q}), \tag{25}$$

which provides a collision-free guarantee for the given confidence level δ_{CL} in dynamic environments. Further details of the integration of probabilistic collision detection with trajectory optimization can be found in [20].

5 Results

In this section, we describe our implementation and highlight the performance of our probabilistic collision checking and trajectory planning algorithm on different benchmark scenarios. We measure the performance of our planning algorithm in simulated environments with difference benchmark scenarios and robot arm models, and validate our algorithm using experiments with a real 7-DOF Fetch robot arm. In our experiments, bounding spheres are automatically generated along the medial axis of each robot link. The environments have some complex static obstacles such as tools or furnitures in a room. The dynamic obstacle is a human, and we assume that the robot operates in close proximity to the human; however, the human does not intend to interact with the robot. We use a Kinect device as the depth sensor, which can represent a human as 30-35k point clouds. We compute the state of human obstacle model which has 60 DOFs. Details of the belief state estimation of dynamic obstacles is given in [20].

5.1 Experimental Results

Robot	Robot BV	Human BV	Prob. Col BV Pairs	Prob. Col Computation Time (ms)
IIWA	40	336	13440 (40x336)	0.147
UR5	56	336	18816 (56x336)	0.282
Fetch	76	336	25536 (76x336)	0.526

Table 1: **Performance of our probabilistic collision detection:** We measure the computation time of the probabilistic collision detection per single robot configuration.

Table 1 shows the computation time of the probabilistic collision detection per single robot configuration. We evaluate (10) in Section 3 for each bounding volume pair correspond to a robot and a human obstacle, and the computation time is linear to the number of pairs.

Table 2 describes the benchmark scenarios and the performance of the planning results for simulated environments. We set $\delta_{CL} = 0.95$, except the second benchmark scenarios where the confidence levels vary.

Benchmarks		Scenarios	Planning Results		
Name	Robot		Minimum Distance (m)	Trajectory Duration (sec)	Trajectory Length (m)
Bookshelf	UR5 (6 DOFs)	Stationary obstacle	0.29	3.7	1.29
		Moving obstacle	0.35	5.4	2.14
Tool	IIWA (7 DOFs)	$\delta_{CL} = 0.95$, $\mathbf{v}_t = \mathbf{0}$	0.06	6.0	1.60
		$\delta_{CL} = 0.95$, $\mathbf{v}_t = 0.005\mathbf{I}_{3\times3}$	0.30	6.9	1.92
		$\delta_{CL} = 0.95$, $\mathbf{v}_t = 0.05\mathbf{I}_{3\times3}$	0.32	7.1	2.01
		$\delta_{CL} = 0.99$, $\mathbf{v}_t = 0.05\mathbf{I}_{3\times3}$	0.38	8.3	2.43
Comparisons using Different Prob. Collision Computations	IIWA (7 DOFs)	Our Approach	0.32	7.1	2.01
		Enlarged bounding volumes [3, 19]	0.40	8.8	2.32
		Approximation using the center point PDF [6]	-0.05	3.4	1.38

Table 2: **Planning results in our benchmarks:** We measure the planning results of the computed trajectories: the minimum distance to the human obstacle, trajectory duration, and trajectory length, for different benchmark scenarios.

In our first benchmark, the planner computes a motion for 6-DOF UR5 robot to move an object on the table to a point on the bookshelf. When a human is dashing toward the robot at a fast speed, the robot is aware of the potential collision with the predicted future human position and changes its trajectory (Fig. 4(a)). However, if a standing human only stretches out an arm toward the robot, even if the velocity of the arm is fast, the model-based prediction prevents unnecessary reactive motions, which is different from the prediction models with constant velocity or acceleration extrapolations (Fig. 4(b)).

The second benchmark shows the difference in planning results due to the different confidence and noise levels, for the same recorded human motion. Fig. 4(c)-(e) shows a robot trajectory with different confidence levels and sensor noises. If the obstacle states are assumed as exact and have no noise, the robot can follow the shortest and smoothest trajectory that is close to the obstacle (Fig. 4(c)). However, as the noise of the environment state or expected confidence level becomes higher, the computed robot trajectories become longer and less smooth to avoid potential collision with the obstacles (Fig. 4(d)-(e)).

Fig. 5 shows a 7-DOF Fetch robot arm motion which is computed using our algorithm to avoid collisions with human motion captured in run-time.

5.2 Probabilistic Collision Checking and Trajectory Planning

In the next benchmark, we plan trajectories using the different probabilistic collision detection algorithms which discussed in Section 3.3. We measure the minimum distance between the robot and the human obstacle along the computed trajectory as a safety metric, and the duration and length of the end-effector trajectory as efficiency metrics. The results for the planners with three different probabilistic collision detection algorithms are shown in Table 2. The enlarged bounding volumes have the largest safety margins, but the durations and lengths

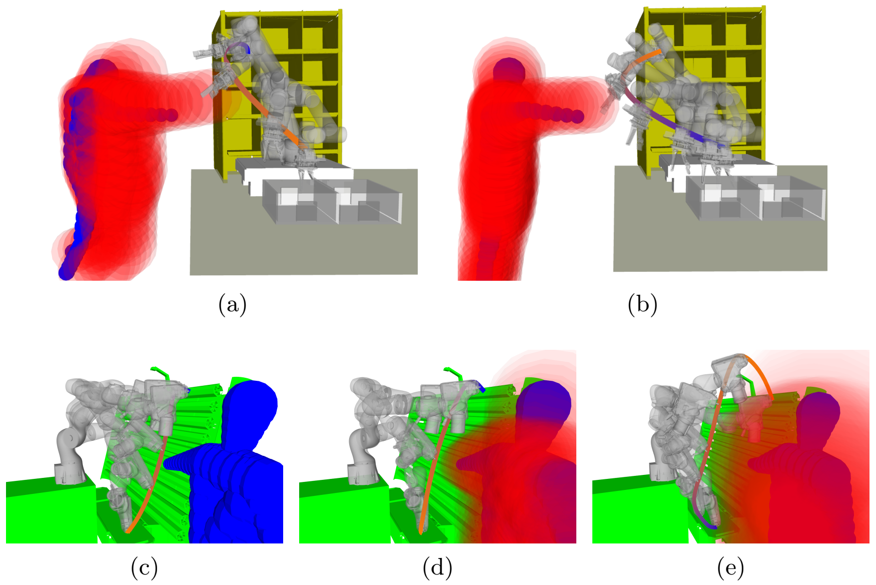


Fig. 4: **Robot Trajectory with Dynamic Human Obstacles:** Static obstacles are shown in green, the estimated current and future human bounding volumes are shown in blue and red, respectively. Our planner uses the probabilistic collision detection to compute the collision probability between the robot and the uncertain future human motion. (a) When a human is approaching the robot, our planner changes its trajectory to avoid potential future collisions. (b) When a standing human only stretches out an arm, our model-based prediction prevents unnecessary reactive motions, which results in a better robot trajectory than the prediction using simple extrapolations. (c)-(e) Robot trajectory with different confidence and noise levels: (c) A trajectory for zero-noise obstacles. (d) $\delta_{CL} = 0.95$ and $\mathbf{v}_t = 0.005 I_{3\times3}$. (e) $\delta_{CL} = 0.99$ and $\mathbf{v}_t = 0.05 I_{3\times3}$.

of the computed trajectories are longer than other approaches, since the overestimated collision probability makes the planner compute trajectories that are unnecessarily far from the obstacles. On the other hand, the approximating approach that uses the probability of the object center point underestimates the collision probability and causes several collisions in the planned trajectories, i.e., the minimum distance between the robot and human obstacle become negative. Our approach shows a similar level of safety with the approach using enlarged bounding volumes, while it also computes efficient trajectories that have shorter trajectory durations and lengths. These benchmarks demonstrate the benefits of our probabilistic collision checking on trajectory planning.

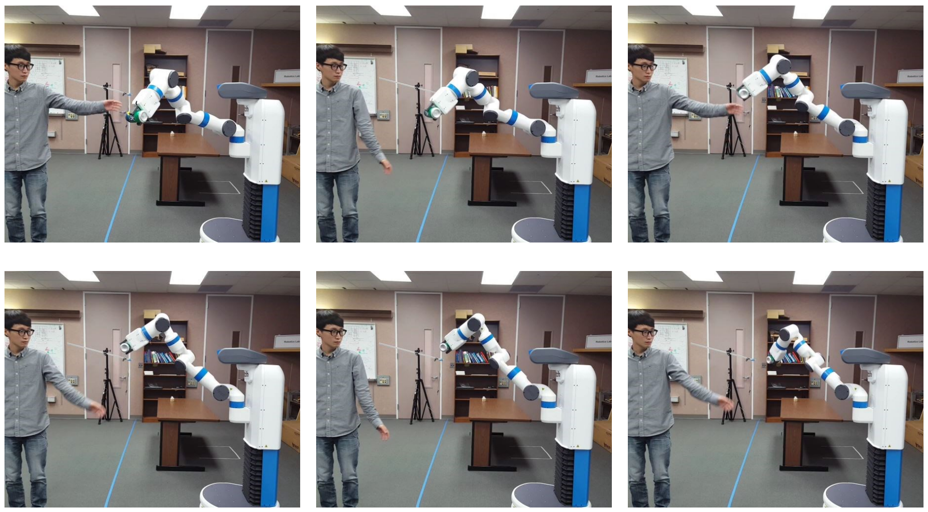


Fig. 5: **Real Robot Experiment:** We evaluate our planning algorithm on 7-DOF Fetch robot arm to compute collision-free robot motion. Our bounded probabilistic collision checking is used for computing safe trajectories.

6 Conclusions, Limitations and Future Work

We present a novel algorithm for collision probability approximation for high-DOF robots in dynamic, uncertain environments. Our approach is fast, and works well in our simulated and real robot results where it can compute efficient collision-free paths with a high confidence level. Our probabilistic collision detection computes tighter upper bounds of the collision probability as compared to prior approaches, and can be used with different planning algorithms. We highlight the performance of our planner on different benchmarks with human obstacles.

Our approach has some limitations. Some of the assumptions used in belief space estimation in terms of Gaussian distribution and Kalman filter may not hold. Moreover, our approach needs pre-defined shape representations of the obstacles. The trajectory optimization may get stuck at a local minima and may not converge to a global optimal solution. Furthermore, our approach assumes that the obstacles in the scene undergo rigid motion. There are many avenues for future work. Our approach only takes into account the imperfect information about the moving obstacles. Uncertainties from control errors or sensor errors, which are rather common with the controllers and sensors, need to be integrated in our approach. Finally, we would to integrate our approach with other robots and evaluate the performance in different scenarios.

Recently, we have extended our approach to general convex polytopes and proposed specialized algorithms for bounding shapes such as AABB, OBB and k-

DOPs [21]. Furthermore, we show that by using bounding volume hierarchies, we can improve the speed and the accuracy of the collision probability computation for non-convex cases.

7 Acknowledgments

This research is supported in part by ARO Contract W911NF-14-1-0437 and NSF award 1305286.

References

1. Bae, K.H., Belton, D., Lichti, D.D.: A closed-form expression of the positional uncertainty for 3d point clouds. Pattern Analysis and Machine Intelligence, IEEE Transactions on 31(4), 577–590 (2009)
2. Bai, H., Cai, S., Ye, N., Hsu, D., Lee, W.S.: Intention-aware online pomdp planning for autonomous driving in a crowd. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on. pp. 454–460. IEEE (2015)
3. Van den Berg, J., Wilkie, D., Guy, S.J., Niethammer, M., Manocha, D.: LQG-Obstacles: Feedback control with collision avoidance for mobile robots with motion and sensing uncertainty. In: Robotics and Automation (ICRA), 2012 IEEE International Conference on. pp. 346–353. IEEE (2012)
4. Blackmore, L.: A probabilistic particle control approach to optimal, robust predictive control. In: Proceedings of the AIAA Guidance, Navigation and Control Conference. No. 10 (2006)
5. Bry, A., Roy, N.: Rapidly-exploring random belief trees for motion planning under uncertainty. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on. pp. 723–730. IEEE (2011)
6. Du Toit, N.E., Burdick, J.W.: Probabilistic collision checking with chance constraints. Robotics, IEEE Transactions on 27(4), 809–815 (2011)
7. Du Toit, N.E., Burdick, J.W.: Robot motion planning in dynamic, uncertain environments. Robotics, IEEE Transactions on 28(1), 101–115 (2012)
8. Groetsch, C.W.: The theory of Tikhonov regularization for Fredholm equations of the first kind, vol. 105. Pitman Advanced Publishing Program (1984)
9. Guibas, L.J., Hsu, D., Kurniawati, H., Rehman, E.: Bounded uncertainty roadmaps for path planning. In: Algorithmic Foundation of Robotics VIII, pp. 199–215. Springer (2010)
10. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial intelligence 101(1), 99–134 (1998)
11. Kroger, T., Wahl, F.M.: Online trajectory generation: Basic concepts for instantaneous reactions to unforeseen events. Robotics, IEEE Transactions on 26(1), 94–111 (2010)
12. Lambert, A., Gruyer, D., Pierre, G.S.: A fast monte carlo algorithm for collision probability estimation. In: Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on. pp. 406–411. IEEE (2008)
13. Lee, A., Duan, Y., Patil, S., Schulman, J., McCarthy, Z., van den Berg, J., Goldberg, K., Abbeel, P.: Sigma hulls for gaussian belief space planning for imprecise articulated robots amid obstacles. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. pp. 5660–5667. IEEE (2013)

14. Leung, C., Huang, S., Kwok, N., Dissanayake, G.: Planning under uncertainty using model predictive control for information gathering. Robotics and Autonomous Systems 54(11), 898–910 (2006)
15. Likhachev, M., Ferguson, D., Gordon, G., Stentz, A., Thrun, S.: Anytime dynamic A*: An anytime, replanning algorithm. In: Proceedings of the International Conference on Automated Planning and Scheduling (2005)
16. Pan, J., Chitta, S., Manocha, D.: Probabilistic collision detection between noisy point clouds using robust classification. In: International Symposium on Robotics Research (ISRR) (2011)
17. Pan, J., Şucan, I.A., Chitta, S., Manocha, D.: Real-time collision detection and distance computation on point cloud sensor data. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on. pp. 3593–3599. IEEE (2013)
18. Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of markov decision processes. Mathematics of operations research 12(3), 441–450 (1987)
19. Park, C., Pan, J., Manocha, D.: ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In: Proceedings of International Conference on Automated Planning and Scheduling (2012)
20. Park, C., Park, J.S., Manocha, D.: Fast and bounded probabilistic collision detection in dynamic environments for high-dof trajectory planning. CoRR abs/1607.04788 (2016), http://arxiv.org/abs/1607.04788
21. Park, J.S., Park, C., Manocha, D.: Efficient probabilistic collision detection for non-convex shapes. CoRR abs/1610.03651 (2016), http://arxiv.org/abs/1610.03651
22. Petti, S., Fraichard, T.: Safe motion planning in dynamic environments. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2210–2215 (2005)
23. Platt Jr, R., Tedrake, R., Kaelbling, L., Lozano-Perez, T.: Belief space planning assuming maximum likelihood observations. In: Proceedings of Robotics: Science and Systems (2010)
24. Sun, W., van den Berg, J., Alterovitz, R.: Stochastic extended lqr: Optimization-based motion planning under uncertainty. In: Algorithmic Foundations of Robotics XI, pp. 609–626. Springer (2015)
25. Zucker, M., Ratliff, N., Dragan, A.D., Pivtoraiko, M., Klingensmith, M., Dellin, C.M., Bagnell, J.A., Srinivasa, S.S.: CHOMP: Covariant hamiltonian optimization for motion planning. International Journal of Robotics Research (2012)