Safe Motion Planning with Environment Uncertainty

Antony Thomas, Fulvio Mastrogiovanni, Marco Baglietto

Department of Informatics, Bioengineering, Robotics, and Systems Engineering, University of Genoa, Via All'Opera Pia 13, 16145 Genoa, Italy.
{antony.thomas@dibris.unige.it, fulvio.mastrogiovanni@unige.it, marco.baglietto@unige.it}

Abstract

We present an approach for safe motion planning under robot state and environment (obstacle and landmark location) uncertainties. To this end, we first develop an approach that accounts for the landmark uncertainties during robot localization. Existing planning approaches assume that the landmark locations are well known or are known with little uncertainty. However, this might not be true in practice. Noisy sensors and imperfect motions compound to the errors originating from the estimate of environment features. Moreover, possible occlusions and dynamic objects in the environment render imperfect landmark estimation. Consequently, not considering this uncertainty can wrongly localize the robot, leading to inefficient plans. Our approach thus incorporates the landmark uncertainty within the Bayes filter estimation framework. We also analyse the effect of considering this uncertainty and delineate the conditions under which it can be ignored. Second, we extend the state-of-the-art by computing an exact expression for the collision probability under Gaussian distributed robot motion, perception and obstacle location uncertainties. We formulate the collision probability process as a quadratic form in random variables. Under Gaussian distribution assumptions, an exact expression for collision probability is thus obtained which is computable in real-time. In contrast, existing approaches approximate the collision probability using upper-bounds that can lead to overly conservative estimate and thereby suboptimal plans. We demonstrate and evaluate our approach using a theoretical example and simulations. We also present a comparison of our approach to different state-of-the-art methods.
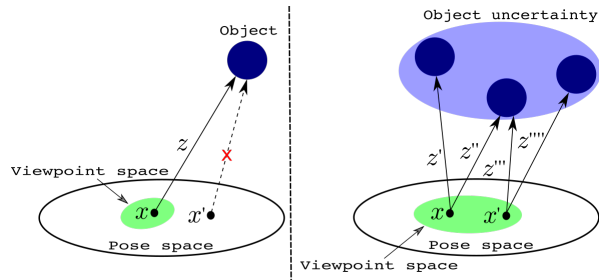
---

1. Introduction

Robots have become more pervasive and are being increasingly used in close proximity to humans and other objects (both static and dynamic) in factories, living spaces, elderly care, and robotic surgery. Planning for collision free trajectories in real-time is imperative for robots to operate safely and efficiently in such realistic conditions. However, uncertainties often arise due to insufficient knowledge about the environment, imperfect sensing or inexact robot motions. In these situations, it is indispensable to employ approaches that perform safe motion planning under motion and sensing uncertainties. Planning is therefore performed in the *belief* space, which corresponds to the set of all probability distributions over possible robot states and other variables of interest. However, at the planning time, future observations are yet to be obtained. Thus, for efficient planning and decision making, it is required to reason about future belief distributions due to possible actions and the corresponding expected future observations. The corresponding problem, known as *Belief Space Planning* (BSP), falls under the category of *Partially Observable Markov Decision Processes* (POMDPs) [1].

Uncertain environments are such that they often preclude the existence of collision free trajectories [2]. In the presence of noisy sensors, both the robot and the environment state cannot be estimated precisely and one can only reason in terms of the corresponding belief states. Moreover, in case of dynamic obstacles, their future states have to be predicted and they are not known exactly due to the lack of perfect knowledge of their motions. As such, providing safety guarantees is difficult and for safe navigation, both the robot state uncertainty and the uncertainty in obstacle estimates need to be considered while computing collision probabilities.

Most robotic tasks require the knowledge of where the robot is with respect to the environment. Localization is therefore one of the most fundamental prob-

lem in robotics and significantly impacts planning and decision making. As such, localization is therefore a key aspect for safe and efficient navigation. However, existing approaches assume that the landmark locations are known precisely or with little uncertainty. For example, given the map of the environment, while planning for future actions the standard Markov localization does not take into account the map uncertainty (that is, landmark locations are assumed to be perfect). This means that given the map and the sensing range, there exists a region from which the landmark can be observed. This however, might not be true in practice. For example, let us consider a *Simultaneous Localization and Mapping* (SLAM) session. Wrong data association or dynamics objects preventing loop closures could lead to wrongly estimated landmark locations and thereby the corresponding map. Thus landmark estimates arising out of such a SLAM session might not be known precisely. It is noteworthy that due to this landmark location uncertainty the regions from which the landmark can be observed are also uncertain. This is visualized in Fig. 1. We define the *pose space* as the set of all possible poses the robot can assume. The blue blob denotes the object which when viewed from a pose x produces an observation z. Different observations are produced when the object is viewed from distinct poses such that the object falls within the sensing range and there are no occlusions. The set of all such poses is a subset of the pose space and is defined to be the *viewpoint space* (green region in the figure). We note that the viewpoint space is sensor-dependent and is determined by the sensing range and other aspects such as occluding objects. Intuitively, the viewpoint space is object depended and this set changes for each object. However, a pose x' that falls outside the viewpoint space does not produce an observation. Consequently, as seen on the left hand side of the figure, when the object location is known precisely, there exists a subset of the pose space from which the object can be observed. On the right hand side of the figure, the light-blue shaded region denotes the uncertainty in object location. Thus in practice the object can be anywhere within the uncertainty region. As a result, given a pose, it cannot be said with certainty that the object can be observed. Subsequently, the cardinality of the

Figure 1: The blue blob denotes an object in the environment. The green region is the viewpoint space corresponding to the set of poses from which the object can be observed. Robot pose x produces an observation z however x' does not produce an observation. On the right hand side, the light-blue shaded region denotes the uncertainty in object location. In practice the object can be anywhere within the uncertainty region. As a result, depending on where the object really is, it can be observed from either x or x' or both the poses.

subset of the pose space from which the object can be observed is increased, that is, the viewpoint space has increased (green region in the figure). As seen in the figure, depending on the object really is, it can be observed from either x or x' or both the poses. As a result, one can only reason in terms of the probability of observing the object from the considered pose or the viewpoint. Therefore, a probability distribution function for the viewpoint space is obtained where the mean viewpoint corresponds to observing the object with highest probability. Not accounting for this uncertainty can cause localization errors, leading to inefficient plans. In this paper, we will use the term *object uncertainty* to refer to this notion of uncertainty in landmark location.

1.1. Related Work

Much research activities about planning under robot state uncertainty have been carried out in the past few years, with applications spanning a variety of areas [3, 4, 5, 6, 7, 8, 9, 10]. Yet, most approaches assume that the landmarks are fairly well known or are known with little uncertainty. However, in practice, the environment is seldom known with high certainty and hence providing for-

mal guarantees for safe robotic tasks under environment uncertainty is of vital importance. In this work we explicitly consider uncertainties in the landmark locations and derive the resulting Bayes filter.

Several methods exist in the literature to compute collision probability and they differ from one another in different aspects such as (1) the formulation of the collision constraint, (2) assumptions in the shape of the robot and the obstacle, (3) modelling the associated uncertainties. The approach of truncating Gaussian distributions is leveraged to compute risk-aware and asymptotically optimal trajectories by [11]. [12] truncate [13] the estimated *a priori* Gaussian state distributions to consider only the collision free samples. Thus propagating these truncated distributions enable them to compute collision free trajectories. However, it must hold that the propagated distributions are Gaussians and therefore the truncated distributions are approximated to be Gaussian distributions. Though we model uncertainties using Gaussian distribution, we do not truncate them. Bounding volume approaches enlarge the robot by their 3-σ uncertainties. Rectangular bounding boxes for robot and the obstacles are used in [14] to compute an approximate upper bound. In [15] the future state distributions are predicted and the uncertainties are used to compute bounded collision probabilities. [16] use sigma hulls for bounding robot links and compute the signed distance of these hulls to the obstacles to formulate the collision avoidance constraints. These approaches typically overestimate collision probabilities and the computed values tend to be larger than the actual values.

The joint distribution between the robot and the obstacle is used to derive the collision constraint in [17] and [18] , giving bounded collision probabilities. The probabilities are obtained by marginalizing the joint distribution. However, since there is no closed form solution to this formulation, an approximate formulation is computed in [17], [18]. Assuming that the robot radius is negligible the joint distribution can be approximated as the product of the volume occupied by the robot and the conditional distribution of the obstacle evaluated at the robot location. However, as identified by [19], both the approaches produce tight bounds only when the sizes of objects are relatively very

small compared with their position uncertainties. In contrast, we do not assume any approximation and derive an exact expression for collision probability computatation. [20], an approximation is computed using *Monte Carlo Integration* (MCI), which is nonetheless computationally intensive. Another related work that uses a Monte Carlo approach and is real-time compatible is *Monte Carlo Motion Planning* (MCMP) [21]. They first solve a deterministic motion planning problem with an inflated obstacle and later adjust the inflation to compute the desired safe path.

Chance-constrained[1] approaches compute approximate upper bounds by linearizing the collision conditions. In contrast, we compute the exact collsion probability value. [22] employs chance-constraints to ensure that the probability of collision is below a specified threshold. This approach is leveraged to compute bounded collision-free trajectories with dynamic obstacles by [19], wherein the dynamic obstacles follow a constant velocity model with Gaussian noise. In [2], a *Gaussian Process* (GP) based approach is used to learn motion patterns (a mapping from states to trajectory derivatives) to identify possible future obstacles trajectories. [23] focus exclusively on obstacle uncertainty. They formalize a notion of *shadows*, which are the geometric equivalent of confidence intervals for uncertain obstacles. The shadows fundamentally give rise to loose bounds but the computational complexity of bounding the collision probability is greatly reduced. Uncertain obstacles are modelled as polytopes with Gaussian-distributed faces by [24]. Planning a collision-free path in the presence of *risk zones* is considered by [25] by penalizing the time spent in these zones. Risk contours map, which take into account the risk information (uncertainties in location, size and geometry of obstacles) in uncertain environments are used by [26] to obtain safe paths with bounded risks. A related approach for randomly moving obstacles is presented by [27]. Formal verification methods have also been used to construct

---

[1]A chance-constrained approach finds the optimal sequence of control inputs subject to the constraint that the collision probability must be below a user-specified threshold. This constraint is known as a chance constraint.

safe plans [28, 29].

Most of the approaches discussed above leverage Boole's inequality to compute the collision probability along a path by summing or multiplying the probabilities along different waypoints in the path. However, the additive approach assumes that the probabilities along the waypoints are mutually exclusive and the multiplicative approach treats them as independent. Such approaches tend to be overly conservative and rather than computing bounded collision probabilities along a path, the bound should be checked for each configuration along the path itself. In this work we perform this check for each configuration along the path. Moreover, in most approaches, the collision probability computed along each waypoint is an approximation of the true value. For example, the MCI approach of [20] approximates the resulting double summation expression for collision probability to a single summation. [19] compute an approximate upper bound for collision probability by linearizing the collision condition. [18] and [17] assume the volume occupied by the robot to be negligible. On the one hand, such approximations can overly penalize paths and could gauge all plans to be infeasible. On the other hand some approximations can be lower[2] than the true collision probability values and can lead to synthesizing unsafe plans.

1.2. Contributions

This paper makes three main theoretical contributions. First, we incorporate object uncertainties in BSP and derive the resulting Bayes filter in terms of the prediction and measurement updates of the *Extended Kalman Filter* (EKF). We also analyse the effect of incorporating object uncertainty while computing the posterior robot belief state.

The second is the computation of the collision probability under robot state uncertainty and the uncertainties in estimated obstacle locations. We formulate the collision avoidance constraint as a quadratic form in random variables under

---

[2]For example, the approach of [17] computes a lower value than the actual when the robot state covariance is small.

the assumption of spherical geometries for robot and obstacles. Unlike previous approaches that compute an upper bound or derive conservative estimates for the probability of collision, we derive an exact expression for computing it. The convergence of the obtained expression is proved and an upper bound for the truncation error is also derived. We also formalize a notion of safety in order to compute configurations that satisfy the required collision probability bounds. Moreover, we employ a Bayesian framework to predict future states of dynamic obstacles when their motion model is unknown. The current state of dynamic obstacles is estimated given the measurements, and the estimated states are used to predict future trajectories. The approach is not limited to a single obstacle and can be used to estimate the states of all obstacles detected by the robot.

The third is the derivation of collision constraints for convex shaped polygonal objects. Note that we derive the collision avoidance constraint by assuming robot and obstacles to be spherical objects. This is a reasonable assumption for most practical purposes since the robot and obstacles may be enclosed by minimum volume spheres. However, in the case of 2D mobile robot collision avoidance planning, most often it is enough to consider the robot footprint. Due to the minimum volume spheres considered, the footprints are thus circular. Yet, this assumption can lead to overly conservative estimates and while deriving the collision constraint and therefore we go a step beyond previous approaches by considering the exact convex footprints of the robot and the obstacles.

This paper is an extension of the preliminary work presented by [30]. Compared with [30], a more rigorous treatment of object uncertainty is presented by introducing viewpoint and pose spaces. We also provide the derivations for the mean and covariance of the posterior belief when incorporating object uncertainty and further discuss the impact of object uncertainty under varying object location uncertainties. We further derive the collision constraint for non-circular geometries and provide a rigorous comparison of our approach with other state-of-the-art methods. Collision computation is extended to dynamic obstacles

by estimating the state of all the obstacles perceived by the robot. Further, the offline planning approach is extended to real-time online planning. Finally, we discuss the limitations of our approach and delineate suitable extensions to overcome them. We also present new results with static and dynamic obstacles in both single-robot and multi-robot settings.

1.3. Organization

The rest of the paper is organized as follows. We introduce the notations, define the considered problem and the assumptions in Section 2. In Section 3, we derive the posterior belief parameters incorporating object uncertainty. We formulate the collision constraint and derive an expression for the collision probability in Section 4. In this section, we also provide a rigorous comparison of our approach to other approaches. State estimation for dynamic obstacles is then discussed in Section 5. In Section 7 we evaluate our approach in simulation under different mobile robot scenarios. In Section 8 we outline the limitations and discuss possible extensions. Conclusion are drawn in Section 9.

2. Notations and Problem Definition

We shall denote vectors by bold lower case letters, that is $\mathbf{x}$ and its components by lower case letters. The transpose of $\mathbf{x}$ will be denoted by $\mathbf{x}^T$ and its Euclidean norm by $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$. The the expected value of a random vector[3] $\mathbf{x}$ will be denoted by $\mathbb{E}(\mathbf{x})$. Matrices will be denoted by capital letters, that is M. The trace of a square matrix M will be denoted by $tr(M)$ and its determinant by $det(M)$. The identity matrix will be denoted by I or I_n when the dimension needs to be stressed. A diagonal matrix with diagonal elements $\lambda_1, \ldots, \lambda_n$ will be denoted by $diag(\lambda_1, \ldots, \lambda_n)$. Sets will be denoted using mathcal fonts, that is $\mathcal{S}$. Unless otherwise mentioned, subscripts on vectors/matrices will be used to denote time indexes and (whenever necessary) superscripts will be used to indicate the robot or the object that it refers to. For example, $\mathbf{x}_k^i$ represents the

---

[3]By a random vector we refer to a vector random variable.

9

state of robot i at time k. The notation $P(\cdot)$ will be used to denote the probability of an event and the probability density function (pdf) will be denoted by $p(\cdot)$.

We now formally define the problem that we tackle in this paper. Let us consider a mobile robot operating in a partially-observable environment. The map of the environment is either known *a priori* or it is built using a standard *Simultaneous Localization and Mapping* (SLAM) algorithm. At any time k, we denote the robot pose (or configuration) by $\mathbf{x}_k \doteq (x_k, y_k, \theta_k)$, the acquired measurement from objects is denoted by $\mathbf{z}_k$ and the applied control action is denoted as $\mathbf{u}_k$. It is noteworthy that by *objects* we refer to both the landmarks and the obstacles in the environment. We also make the following assumptions: (1) the uncertainties are modelled using Gaussian distributions, (2) the robot and obstacles are assumed to be non-deformable objects.

To describe the dynamics of the robot, we consider a standard motion model with Gaussian noise

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) + w_k \ , \ w_k \sim \mathcal{N}(0, R_k) \tag{1}$$

where w_k is the random unobservable noise, modeled as a zero mean Gaussian. Objects are detected through the robot's sensors and assuming known data association, the observation model can be written as

$$\mathbf{z}_k = h(\mathbf{x}_k, O_k^i) + v_k \ , \ v_k \sim \mathcal{N}(0, Q_k) \tag{2}$$

where O_k^i is the $i-$th detected object and v_k is the zero mean Gaussian noise. We note here that the motion (1) and observation (2) models can be written probabilistically as $p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k)$ and $p(\mathbf{z}_k|\mathbf{x}_k, O_k^i)$, respectively.

Given the models in (1) and (2), in this paper we compute *safe* plans, wherein the probability of collision of the robot with any obstacle is guaranteed to be less than a specified bound while navigating to the goal. To this end, we consider the object uncertainties while localizing the robot. Further, we employ a Bayesian framework to estimate the current state of dynamic obstacles and use the estimated states to predict future obstacle states. The estimated states are

then used for non-myopic collision avoidance planning. Given the robot and obstacle locations, we compute the exact probability of collision under motion, sensing uncertainty, and the uncertainty in obstacle location.

3. Object Uncertainty

As discussed in Section 1, most localization approaches assume that the landmark locations are known precisely or with little uncertainty. This however, might not be true in practice due to noisy measurements and/or imperfect sensors. Thus, it is pertinent that landmark uncertainties are considered within the localization and planning framework. Below, we delineate the incorporation of object uncertainty within the Bayes filter.

3.1. Object Uncertainty Model for BSP

We define the collection of all objects in the environment to be the *object space* $\mathcal{O} = \{O^i | O^i$ is an object, and $1 \leq i \leq |\mathcal{O}|\}$. Let us posit that at time k the robot received a measurement $\mathbf{z}_k$ which was originated by observing the object O_k^i. Given an initial distribution $p(\mathbf{x}_0)$, and the motion and observation models in (1) and (2), the posterior probability distribution at time k is the *belief* $b[\mathbf{x}_k]$ and can be written as

$$b[\mathbf{x}_k] = p(\mathbf{x}_k | \mathbf{z}_k, O_k^i, \mathbf{z}_{0:k-1}, \mathbf{u}_{0:k-1}) \qquad (3)$$

where O_k^i is the object observed at time k, $\mathbf{z}_{0:k-1} \doteq \{\mathbf{z}_0, ..., \mathbf{z}_{k-1}\}$ and $\mathbf{u}_{0:k-1} \doteq \{\mathbf{u}_0, ..., \mathbf{u}_{k-1}\}$. This posterior is thus a multivariate Gaussian, that is, $b[\mathbf{x}_k] \sim \mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k)$, where $\boldsymbol{\mu}_k, \Sigma_k$ are the mean and covariance of $\mathbf{x}_k$, respectively. We note that this posterior belief is computed after incorporating the measurement at time k, that is, $\mathbf{z}_k$.

Given the belief $b[\mathbf{x}_k]$ and an action $\mathbf{u}_k$, the belief before incorporating a measurement will be called the propagated belief and can be written as

$$b[\mathbf{x}_{k+1}^-] = \int_{\mathbf{x}_k} p(\mathbf{x}_{k+1} | \mathbf{x}_k, \mathbf{u}_k) b[\mathbf{x}_k] \qquad (4)$$

Now let us consider that a measurement $\mathbf{z}_{k+1}$ is obtained that corresponds to observing the object O_{k+1}^i. The posterior distribution $b[\mathbf{x}_{k+1}]$ can then be computed using Bayes rule and the theorem of total probability. This expansion is obtained in terms of the belief at the previous time step since the Bayes filter is recursive. Thus we have

$$p(\mathbf{x}_{k+1}|\mathbf{z}_{k+1}, O_{k+1}^i, \mathbf{z}_{0:k}, \mathbf{u}_{0:k}) =$$

$$\eta p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}, O_{k+1}^i) p(O_{k+1}^i|\mathbf{x}_{k+1}) \int_{\mathbf{x}_k} p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k) b[\mathbf{x}_k] \quad (5)$$

where $\eta = 1/p(\mathbf{z}_{k+1}|\mathbf{z}_{0:k}, \mathbf{u}_{0:k})$ is the normalization constant. The term $p(O_{k+1}^i|\mathbf{x}_{k+1})$ denotes the probability of observing the object O_{k+1}^i from the pose $\mathbf{x}_{k+1}$. In other words, this term models the fact, how likely it is to observe O_{k+1}^i from $\mathbf{x}_{k+1}$ and thus models the object uncertainty. The term $p(O_{k+1}^i|\mathbf{x}_{k+1})$ also additionally model aspects such as occlusions due to static obstacles that hinder the observation, occlusions that results due to dynamic obstacles, faulty sensors or other aspects that impedes observations of objects of interest. Thus, given an object one can only reason probabilistically about observing it to obtain the corresponding measurement. However, when the object uncertainty and other additional aspects are ignored an object is observed whenever the robot is within the viewpoint space (see left hand side of Fig. 1). Thus, in the case of such an assumption, for poses within the viewpoint space the term is equal to unity, that is, $p(O_{k+1}^i|\mathbf{x}_{k+1}) = 1$. For poses that lie outside the viewpoint space, $p(O_{k+1}^i|\mathbf{x}_{k+1}) = 0$ and hence no measurement can be obtained. As such, when the object uncertainty is ignored, the term $p(O_{k+1}^i|\mathbf{x}_{k+1})$ can be removed from (5) and the posterior belief parameters can be computed using the standard EKF update equation as

$$K_{k+1} = \bar{\Sigma}_{k+1} H_{k+1}^T \left(H_{k+1} \bar{\Sigma}_{k+1} H_{k+1}^T + Q_{k+1} \right)^{-1}$$

$$\boldsymbol{\mu}_{k+1} = \bar{\boldsymbol{\mu}}_{k+1} + K_{k+1} \left(\mathbf{z}_{k+1} - h(\bar{\boldsymbol{\mu}}_{k+1}) \right) \quad (6)$$

$$\Sigma_{k+1} = (I - K_{k+1} H_{k+1}) \bar{\Sigma}_{k+1}$$

where H_{k+1} is the Jacobian of $h(\cdot)$ with respect to $\mathbf{x}_{k+1}$, and K_{k+1} is the Kalman gain.

The exposition so far has been agnostic to the actual model of $p(O^i_{k+1}|\mathbf{x}_{k+1})$. In general, this term can be modeled given the environment map, the sensing capabilities and the robot objectives. These aspects should hence be incorporated to obtain the actual object uncertainty model. However, in this work we approximate the object distribution as a Gaussian distribution:

$$p(O^i_{k+1}|\mathbf{x}_{k+1}) \sim \mathcal{N}(\boldsymbol{\mu}_{O^i_{k+1}}, \Sigma_{O^i_{k+1}}) \tag{7}$$

where $\boldsymbol{\mu}_{O^i_{k+1}}$ is the viewpoint/pose that corresponds to the maximum probability of observing O^i_{k+1} and $\Sigma_{O^i_{k+1}}$ is the associated uncertainty in the observation.

We will now consider the object uncertainty term $p(O^i_{k+1}|\mathbf{x}_{k+1})$ and derive the Gaussian belief parameters by expanding (5). Expanding the right hand side of (5) using the probability density function (pdf) of multivariate Gaussian distributions, we have $b[\mathbf{x}_{k+1}] = \eta' \int \exp(-\mathcal{J}_{k+1})$, where η' contains the non-exponential terms and $\mathcal{J}_{k+1}$ is given by

$$
\begin{aligned}
\mathcal{J}_{k+1} = \frac{1}{2} & \left(\mathbf{z}_{k+1} - h\left(\bar{\boldsymbol{\mu}}_{k+1}\right) - H_{k+1}\left(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1}\right)\right)^T \\
& Q_{k+1}^{-1}\left(\mathbf{z}_{k+1} - h\left(\bar{\boldsymbol{\mu}}_{k+1}\right) - H_{k+1}\left(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1}\right)\right) \\
& + \frac{1}{2}(\boldsymbol{x}_{k+1} - \boldsymbol{\mu}_{O^i_{k+1}})^T \Sigma_{O^i_{k+1}}^{-1}(\mathbf{x}_{k+1} - \boldsymbol{\mu}_{O^i_{k+1}}) \\
& + \frac{1}{2}(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1})^T \bar{\Sigma}_{k+1}^{-1}(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1}) \quad (8)
\end{aligned}
$$

where H_{k+1} is the Jacobian of $h(\cdot)$ with respect to $\mathbf{x}_{k+1}$. As shown in [31], the covariance Σ_{k+1} is obtained as the inverse of the second derivative of $\mathcal{J}_{k+1}$ with respect to $\mathbf{x}_{k+1}$. The expression for the second derivative is obtained as

$$\frac{\partial^2 \mathcal{J}_{k+1}}{\partial \mathbf{x}_{k+1}^2} = H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \Sigma_{O^i_{k+1}}^{-1} + \bar{\Sigma}_{k+1}^{-1} \tag{9}$$

Therefore the posterior covariance is obtained as

$$\Sigma_{k+1}^{-1} = H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \Sigma_{O^i_{k+1}}^{-1} + \bar{\Sigma}_{k+1}^{-1} \tag{10}$$

The mean of $b[\mathbf{x}_{k+1}]$ is the value that maximizes $b[\mathbf{x}_{k+1}]$ and hence is obtained by equating the first derivative of $\mathcal{J}_{k+1}$ to zero. The expression for the mean

$\boldsymbol{\mu}_{k+1}$ is obtained as (see Appendix A for derivation)

$$\boldsymbol{\mu}_{k+1} = \bar{\boldsymbol{\mu}}_{k+1} + K_{k+1} \left(\mathbf{z}_{k+1} - h\left(\bar{\boldsymbol{\mu}}_{k+1}\right) \right) + \Sigma_{k+1} \Sigma_{O_{k+1}^i}^{-1} \left(\boldsymbol{\mu}_{O_{k+1}^i} - \bar{\boldsymbol{\mu}}_{k+1} \right) \quad (11)$$

where $K_{k+1} = \Sigma_{k+1} H_{k+1}^T Q_{k+1}^{-1}$ is the Kalman gain. We note that when no object uncertainty is considered the update step of the standard EKF gives $\boldsymbol{\mu}_{k+1} = \bar{\boldsymbol{\mu}}_{k+1} + K_{k+1} \left(\mathbf{z}_{k+1} - h\left(\bar{\boldsymbol{\mu}}_{k+1}\right) \right)$. The additional term in (11) rightly adjusts the mean $\boldsymbol{\mu}_{k+1}$ accounting for the fact that the object location is uncertain.

As in the standard EKF based Bayes filter, the expression for the covariance Σ_{k+l} can also be derived in terms of the Kalman gain K_{k+1} and the predicted covariance $\bar{\Sigma}_{k+1}$. Using the matrix inversion lemma on (10), the following expression is obtained (see Appendix B for derivation)

$$\Sigma_{k+1} = (I - K_{k+1} H_{k+1}) \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} \quad (12)$$

where $\tilde{\Sigma}_{k+l} = \left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right)^{-1}$.

When object uncertainty is not considered, the update step of the standard EKF gives $\Sigma_{k+1} = (I - K_{k+1} H_{k+1}) \bar{\Sigma}_{k+1}$. The extra terms in (12) account for the object uncertainty and scale the posterior covariance accordingly. We note that when object uncertainty is not considered, $p(O_{k+1}^i|\mathbf{x}_{k+1}) = 1$ and hence the results in (11) and (12) reduce to that of the standard EKF case in (6). The method presented above is easily generalized to multiple objects observed at any time instant. This is done by following the sequential-sensor method ([32]), considering the fact that given the current state estimate, the observations are independent of each other.

3.2. Implications

Let us now analyse the effect of object uncertainty. As discussed above when object uncertainty is not assumed, $p(O_{k+1}^i|\mathbf{x}_{k+1}) = 1$, and therefore the posterior belief parameters reduce to that of the standard EKF case. However, in practice, one should consider object uncertainty and the posterior belief parameters are as delineated in (11) and (12). Yet, the impact of considering object uncertainty in localisation depends on the covariance of the estimated object

location. When the covariance of the object location is much larger compared to the predicted robot belief state covariance, the impact of considering object uncertainty is greatly reduced.

Lemma 1. *When the covariance of the estimated object location is much larger than the predicted robot belief state covariance, that is, when $\Sigma_{O_{k+1}^i} \gg \bar{\Sigma}_{k+1}$, then the object uncertainty has limited impact and can be ignored.*

Proof. In order to prove the above lemma, it suffices to show that when $\Sigma_{O_{k+1}^i} \gg \bar{\Sigma}_{k+1}$, the posterior belief parameters reduce to that of the standard EKF update case as given in (6). Let us first consider the expession in (10). Using the fact that $\Sigma_{O_{k+1}^i} \gg \bar{\Sigma}_{k+1}$, then $\Sigma_{O_{k+1}^i}^{-1} \ll \bar{\Sigma}_{k+1}^{-1}$ and hence it can be neglected when compared to $\bar{\Sigma}_{k+1}^{-1}$. This gives $\Sigma_{k+1} = \left(H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \bar{\Sigma}_{k+1}^{-1} \right)^{-1}$, and is the expression for the posterior belief covariance when object uncertainty is not considered. Again, using $\Sigma_{O_{k+1}^i} \gg \bar{\Sigma}_{k+1}$, $\bar{\Sigma}_{k+1}$ can be neglected from the sum $\left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right)$. The expression for Kalman gain thus reduces to

$$
K_{k+1} = \bar{\Sigma}_{k+1} \left(\Sigma_{O_{k+1}^i} \right)^{-1} \Sigma_{O_{k+1}^i} H_{k+1}^T
$$
$$
\left(H_{k+1} \bar{\Sigma}_{k+1} \left(\Sigma_{O_{k+1}^i} \right)^{-1} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1}
$$
$$
= \bar{\Sigma}_{k+1} H_{k+1}^T \left(H_{k+1} \bar{\Sigma}_{k+1} H_{k+1}^T + Q_{k+1} \right)^{-1} \quad (13)
$$

Thus, as can be seen in (6), the Kalman gain is exactly the gain obtained when object uncertainty is not considered. Similarly, we have $\Sigma_{k+1} \Sigma_{O_{k+1}^i}^{-1} \ll 1$, thus $\boldsymbol{\mu}_{k+1} = \bar{\boldsymbol{\mu}}_{k+1} + K_{k+1} \left(\mathbf{z}_{k+1} - h\left(\bar{\boldsymbol{\mu}}_{k+1} \right) \right)$. Following a similar argument, it is easily seen that $\Sigma_{k+1} = \left(I - K_{k+1} H_{k+1} \right) \bar{\Sigma}_{k+1}$. This completes the proof of Lemma 1.

Although the above result might seem counter-intuitive at first, we note here that the viewpoint space, when object uncertainty is not considered, is the space centred around the mean of the viewpoint space when the object uncertainty is considered. When the covariance of the object location is very high, then the probability values for viewpoints slightly away from the mean reduces drastically. Consequently considering these viewpoints adds little impact.

4. Exact Collision Probability

We denote by $\mathcal{R}$ the set of all points occupied by a rigid-body robot at any given time. Similarly, let $\mathcal{S}$ represent the set of all points occupied by a rigid-body obstacle. A collision occurs if there exits a point such that it is in both $\mathcal{R}$ and $\mathcal{S}$. Thus the collision condition is defined as

$$\mathcal{R} \cap \mathcal{S} \neq \{\phi\} \tag{14}$$

and we denote the probability of collision as $P\left(\mathcal{R} \cap \mathcal{S} \neq \{\phi\}\right)$. In this work we assume spherical geometries for $\mathcal{R}$ and $\mathcal{S}$ with radii r_1 and s_1, respectively. We assign body-fixed reference frames to both the robot and the obstacle located at $\mathbf{x}_k$ and $\mathbf{s}_k$, respectively in the global frame. By abuse of notation we will use $\mathbf{x}_k$ and $\mathbf{s}_k$ equivalently to $\mathcal{R}$ and $\mathcal{S}$. The collision condition is thus defined in terms of the body-fixed frames as

$$\mathcal{C}_{\mathbf{x}_k,\mathbf{s}_k} : \mathcal{R} \cap \mathcal{S} \neq \{\phi\} \tag{15}$$

We recall here that the locations of the obstacles are in general uncertain. Let us now consider an obstacle at any given time instant, distributed according to the Gaussian $\mathbf{s}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{s}_k}\right)$, where $\boldsymbol{\mu}_{\mathbf{s}_k}$ represents the mean and $\Sigma_{\mathbf{s}_k}$ the associated covariance. Given the current robot state $\mathbf{x}_k$ and the obstacle state $\mathbf{s}_k$, the probability of collision can be formulated if the joint distribution between the robot and the obstacle state is known. In such a case the collision probability is given by

$$P\left(\mathcal{C}_{\mathbf{x}_k,\mathbf{s}_k}\right) = \int_{\mathbf{x}_k} \int_{\mathbf{s}_k} I_c(\mathbf{x}_k, \mathbf{s}_k) p(\mathbf{x}_k, \mathbf{s}_k) \tag{16}$$

where $\mathcal{C}_{\mathbf{x}_k,\mathbf{s}_k}$ as defined above represents the fact that the robot configuration $\mathbf{x}_k$ and its collision with an obstacle at location $\mathbf{s}_k$ is considered, and I_c is an indicator function defined as

$$I_c(\mathbf{x}_k, \mathbf{s}_k) = \begin{cases} 1 & \text{if } \mathcal{R} \cap \mathcal{S} \neq \{\phi\} \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

16

and $p(\mathbf{x}_k, \mathbf{s}_k)$ is the joint distribution of the robot and the obstacle. [19] compute an approximate upper bound for the collision probability by linearizing the collision condition. [20] use MCI to compute (16). However, the resulting double summation is approximated to a single summation to reduce computational complexity. [17], [18] approximate the integral in (16) as $Vp(\mathbf{x}_k, \mathbf{s}_k)$, where V is the volume occupied by the robot. For computing $p(\mathbf{x}_k, \mathbf{s}_k)$, they first assume a distribution centered around the obstacle with the covariance being the sum of the robot and obstacle location uncertainties. Then the density $p(\mathbf{x}_k, \mathbf{s}_k)$ is computed by assuming a constant robot location. Du Toit and Burdick use the robot center, whereas in [18] the maximum of $p(\mathbf{x}_k, \mathbf{s}_k)$ on the surface of the robot is used to obtain an upper bound. However, the approximation is valid only when the robot radius is negligible. To demonstrate, let us re-write the collision condition as

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) = \int_{\mathbf{x}_k} \left[\int_{\mathbf{s}_k \in \mathcal{R}} p(\mathbf{s}_k | \mathbf{x}_k) \right] p(\mathbf{x}_k) \tag{18}$$

If the robot radius is negligible then it can be assumed that $\mathbf{s}_k = \mathbf{x}_k$, giving

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) = \int_{\mathbf{x}_k} \left[p(\mathbf{s}_k = \mathbf{x}_k | \mathbf{x}_k) \int_{\mathbf{s}_k \in \mathcal{R}} \right] p(\mathbf{x}_k) \tag{19}$$

Thus assuming a constant value of the obstacle evaluated at the robot location, we have

$$V = \int_{\mathbf{s}_k \in \mathcal{R}} \tag{20}$$

where V is the volume occupied by the robot. The approximate collision probability is thus

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) \approx V \int_{\mathbf{x}_k} p(\mathbf{s}_k = \mathbf{x}_k | \mathbf{x}_k) p(\mathbf{x}_k) \tag{21}$$

Assuming that the robot and the obstacle locations are independent Gaussian distributions with $\mathbf{s}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{s}_k}\right)$ and $\mathbf{x}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_k}, \Sigma_{\mathbf{x}_k}\right)$, the collision probability can be approximately written as

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) \approx Vp(\mathbf{x}_k = \boldsymbol{\mu}_{\mathbf{x}_k}, \mathbf{s}_k = \boldsymbol{\mu}_{\mathbf{s}_k}) \tag{22}$$

17

where

$$p(\mathbf{x}_k = \boldsymbol{\mu}_{\mathbf{x}_k}, \mathbf{s}_k = \boldsymbol{\mu}_{\mathbf{s}_k}) = det\left(2\pi\left(\Sigma_{\mathbf{s}_k} + \Sigma_{\mathbf{x}_k}\right)\right)^{-\frac{1}{2}}$$
$$\exp\left(-\frac{1}{2}(\boldsymbol{\mu}_{\mathbf{x}_k} - \boldsymbol{\mu}_{\mathbf{s}_k})^T \Sigma^{-1} (\boldsymbol{\mu}_{\mathbf{x}_k} - \boldsymbol{\mu}_{\mathbf{s}_k})\right) \quad (23)$$

Other existing approaches truncate the state distributions or compute approximate upper bounds using chance-constraints. As such, these approaches compute an approximation of the collision probability. In contrast, we formulate the collision constraint as a quadratic form in random variables, allowing us to compute an exact expression for the collision probability. In the remainder of this section a rigorous treatment of the same is presented.

Since the robot and the obstacles are assumed to be spherical objects, the collision constraint is written as

$$\|\mathbf{x}_k - \mathbf{s}_k\|^2 \leq (r_1 + s_1)^2 \quad (24)$$

where (as before) $\mathbf{x}_k$ and $\mathbf{s}_k$ are the random vectors that denote the robot and obstacle locations, respectively. Let the current estimates of the two random vectors be distributed according to $\mathbf{s}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{s}_k}\right)$ and $\mathbf{x}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_k}, \Sigma_{\mathbf{x}_k}\right)$. Let us denote the difference between the two random variables by $\boldsymbol{w} = \mathbf{x}_k - \mathbf{s}_k$. Using the expression for the difference between two Gaussian distributions, we have $\boldsymbol{w} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_k} - \boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{x}_k} + \Sigma_{\mathbf{s}_k}\right)$. The collision constraint in (24) can now be written in terms of $\boldsymbol{w}$,

$$\boldsymbol{y} = \|\boldsymbol{w}\|^2 = \boldsymbol{w}^T \boldsymbol{w} \leq (r_1 + s_1)^2 \quad (25)$$

where $\boldsymbol{y}$ is a random vector distributed according to the squared L_2-norm of $\boldsymbol{w}$. Now, given the probability density function (pdf) of $\boldsymbol{y}$, the collision constraint reduces to solving the integral

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) = \int_0^{(r_1+s_1)^2} p(y) \quad (26)$$

where $p(y) = P_{\boldsymbol{y}}(\boldsymbol{y} = y)$ is the pdf of $\boldsymbol{y}$. It is noteworthy that the above integral is the cumulative distribution function (cdf) of $\boldsymbol{y}$, that is, $P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) = F_{\boldsymbol{y}}(y)$,

where $F_{\boldsymbol{y}}(y)$ denotes the cdf. Thus the collision condition reduces to finding the cdf of $\boldsymbol{y}$ such that $\boldsymbol{y} \leq (r_1 + s_1)^2$. As a consequence, we have

$$P\left(\mathcal{C}_{\mathbf{x}_k, \mathbf{s}_k}\right) = P\left(\boldsymbol{y} \leq (r_1 + s_1)^2\right) = F_{\boldsymbol{y}}\left((r_1 + s_1)^2\right) \tag{27}$$

In the following Sections, we will first show that the collision constraint is a quadratic form in random variables and later derive an exact expression for the cdf of the quadratic from.

4.1. Quadratic Form in Random Variables

We define a quadratic form in random variables:

Definition 1. *Let $\boldsymbol{x} = (x_1, \ldots, x_n)^T$ denote a random vector with mean $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$ and covariance matrix Σ. Then the quadratic form in the random variables $x_1, \ldots, x_n$ associated with an $n \times n$ symmetric matrix $A = (a_{ij})$, with i and j in $1, \ldots, n$, is*

$$Q(\boldsymbol{x}) = Q(x_1, \ldots, x_n) = \boldsymbol{x}^T A \boldsymbol{x} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j \tag{28}$$

Let us define $\boldsymbol{v} = \Sigma^{-\frac{1}{2}} \boldsymbol{x}$ and define a random vector $\boldsymbol{z} = \left(\boldsymbol{v} - \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}\right)$. The resulting distribution of $\boldsymbol{z}$ is thus zero mean with covariance being the identity matrix. Therefore, the quadratic form becomes

$$Q(\boldsymbol{x}) = \left(\boldsymbol{z} + \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}\right)^T \Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}} \left(\boldsymbol{z} + \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}\right) \tag{29}$$

Let us suppose there exists an orthogonal matrix P, that is, $PP^T = I$ which diagonalizes $\Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}}$, then $P^T \Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}} P = \text{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of $\Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}}$. The quadratic form can now be written as

$$\begin{aligned} Q(\boldsymbol{x}) &= \left(\boldsymbol{z} + \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}\right)^T \Sigma^{\frac{1}{2}} A \Sigma^{\frac{1}{2}} \left(\boldsymbol{z} + \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}\right) \\ &= (\boldsymbol{u} + \boldsymbol{b})^T \text{diag}(\lambda_1, \ldots, \lambda_n) (\boldsymbol{u} + \boldsymbol{b}) \end{aligned} \tag{30}$$

where $\boldsymbol{u} = P^T \boldsymbol{z} = (u_1, \ldots, u_n)^T$ and $\boldsymbol{b} = P^T \Sigma^{-\frac{1}{2}} \boldsymbol{\mu} = (b_1, \ldots, b_n)^T$. The expression in (30) can be written concisely,

$$Q(\boldsymbol{x}) = \boldsymbol{x}^T A \boldsymbol{x} = \sum_{i=1}^{n} \lambda_i (u_i + b_i)^2 \tag{31}$$

19

It is easily verified that the left hand side of (25), that is $\boldsymbol{w}^T\boldsymbol{w}$, is in the quadratic form $Q(\boldsymbol{w})$ with $A = I$, that is, the identity matrix. Thus the collision probability can be computed from the cdf of the quadratic form.

4.2. Series Expansion for the Quadratic Form

We describe below the most general method used to obtain a series expansion for the pdf and cdf of the quadratic form in random variables. Various other methods exists in the literature and we refer the interest readers to [33] for a brief survey. The series expansion that we seek for the pdf of the quadratic form is of the form

$$p_{\mathbf{y}}(y) = p(\mathbf{y} = y) = \sum_{k=0}^{\infty} c_k h_k(y) \tag{32}$$

where c_k is a sequence of complex number and $\{h_k\}$ is a known sequence of the form y^k. Let the Laplace transform of $h_k(y)$ be denoted by $L(h_k(y))$. In the expansion sought here, the Laplace transform is of the special form ([34])

$$L(h_k(y)) = \xi(s)\eta^k(s) \tag{33}$$

where, for $Re(s) > \alpha$ and α being a real constant, $\xi(s)$ is a non-vanishing (non-zero everywhere) analytic function and $\eta(s)$ is an analytic function with an inverse function $\eta(\zeta(\theta)) = \theta$. Now we are interested in the case where the series expansion is convergent, that is, $\sum_{k=0}^{\infty} c_k h_k(y) < \infty$. For any real number β, let us define

$$\sum_{k=0}^{\infty} c_k h_k(y) \leq \sum_{k=0}^{\infty} |c_k||h_k(y)| \leq \alpha e^{\beta y}, \ y \in [0,\infty] \tag{34}$$

If the above equation is satisfied almost everywhere, then computing the Laplace transform, we have

$$\int_0^{\infty} e^{-sy}\alpha e^{\beta y} = \alpha \int_0^{\infty} e^{-(s-\beta)y}dy < \infty \tag{35}$$

if $(s - a) > 0$. Therefore, from Lebesgue's dominated convergence theorem, we have the following lemma.

Lemma 2. *Let $\{h_k\}_0^\infty$ be a sequence of measurable complex valued functions on $[0, \infty]$ and $\{c_k\}_0^\infty$ be a sequence of complex numbers such that almost everywhere the following is satisfied*

$$\sum_{k=0}^{\infty} |c_k||h_k(y)| \leq \alpha e^{\beta y}, \ y \in [0, \infty] \tag{36}$$

where α, β are real constants. Then when $s > 0$ and $p_{\boldsymbol{y}}(y) = \sum_{k=0}^{\infty} c_k h_k(y)$, we have

$$L\left(p_{\boldsymbol{y}}(y)\right) = \sum_{k=0}^{\infty} c_k L(h_k(y)) \tag{37}$$

The implications of the above lemma are twofold. The first is that the series expansion is convergent. This however is rather straightforward from our construction of the series expansion. The second is the fact that the Laplace transform of the series $p_{\mathbf{y}}(y)$ can be obtained by taking the Laplace transform of the individual terms of the series. This fact will be used below to derive the pdf and the cdf of the quadratic from. We now state the following theorem without proof. The proof may be found in [33].

Theorem 1. *For $Q(\boldsymbol{x}) = \boldsymbol{y} = \boldsymbol{x}^T A \boldsymbol{x}$ with $A = A^T > 0, \boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \Sigma > 0$, the moment generating function $M_Q(t)$ of Q is given by*

$$M_Q(t) = \exp\left(-\frac{1}{2}\sum_{i=1}^{n} b_i^2\right) \exp\left(\frac{1}{2}\sum_{i=1}^{n} b_i^2(1 - 2t\lambda_i)^{-1}\right) \prod_{i=1}^{n}(1 - 2t\lambda_i)^{-\frac{1}{2}} \tag{38}$$

where the b_i, λ_i are the parameters of the quadratic form as defined in Section 4.1. Let us now define the series $M(\theta)$ such that

$$M(\theta) = \sum_{k=0}^{\infty} c_k \frac{L(h_k(y))}{\xi(\zeta(\theta))} = \sum_{k=0}^{\infty} c_k \theta^k \tag{39}$$

where the infinite series is a uniformly convergent series for θ in some region with $M(\theta) > 0$, $M(0) = c_0$ and $s = \zeta(\theta)$. We note here that if $p_{\mathbf{y}}(y) = 0$ for $y < 0$, then $M_Q(-t)$ represents the Laplace transform of $p_{\mathbf{y}}(y)$. Thus, from (38)

we have

$$L\left(p_{\mathbf{y}}(y)\right) = \exp\left(-\frac{1}{2}\sum_{i=1}^{n}b_i^2\right)\exp\left(\frac{1}{2}\sum_{i=1}^{n}b_i^2(1+2s\lambda_i)^{-1}\right)\prod_{i=1}^{n}(1+2s\lambda_i)^{-\frac{1}{2}}$$

(40)

Using $\zeta(\theta) = \theta^{-1}$, we have

$$L\left(p_{\mathbf{y}}(y)\right) = s^{-\frac{n}{2}}M(\theta)$$

(41)

Thus we obtain,

$$c_0 = M(0) = \exp\left(-\frac{1}{2}\sum_{i=1}^{n}b_i^2\right)\prod_{i=1}^{n}(2\lambda_i)^{-\frac{1}{2}}$$

(42)

Differentiating the natural logarithm of $M(\theta)$, we get the following form

$$\ln M(\theta) = d_0 + \sum_{k=1}^{\infty}d_k\frac{\theta^k}{k}$$

(43)

where

$$d_0 = -\frac{1}{2}\sum_{i=1}^{n}b_i^2 + \ln\prod_{i=1}^{n}(2\lambda_i)^{-\frac{1}{2}}$$

$$d_k = \frac{1}{2}\sum_{i=1}^{n}\left(1 - kb_i^2\right)(2\lambda_i)^{-k}$$

(44)

From (41), we have the following lemma.

Lemma 3.

$$L\left(p_{\mathbf{y}}(y)\right) = \sum_{k=0}^{\infty}c_k(-1)^k s^{-\left(\frac{n}{2}+k\right)}$$

(45)

We now obtain the required expressions for the pdf and cdf of the quadratic form of $Q(\mathbf{x})$.

Lemma 4. *The cdf of $Q(\boldsymbol{x}) = \boldsymbol{y} = \boldsymbol{x}^T A\boldsymbol{x}$ with $A = A^T > 0$, $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, $\Sigma > 0$ is*

$$F_{\boldsymbol{y}}(y) = p(\boldsymbol{y} \le y) = \sum_{k=0}^{\infty}(-1)^k c_k \frac{y^{\frac{n}{2}+k}}{\Gamma\left(\frac{n}{2}+k+1\right)}$$

(46)

22

and its pdf is given by

$$p_{\boldsymbol{y}}(y) = p(\boldsymbol{y} = y) = \sum_{k=0}^{\infty} (-1)^k c_k \frac{y^{\frac{n}{2}+k-1}}{\Gamma\left(\frac{n}{2}+k\right)} \tag{47}$$

where Γ denotes the gamma function, c_0 and d_0, d_k are the terms defined in (42) and (44), respectively. The expression for c_k is given by (see Appendix C for derivation)

$$c_k = \frac{1}{k} \sum_{j=0}^{k-1} d_{k-j} c_j, \ \ k \geq 1 \tag{48}$$

Proof. From Lemma 3, we have $L\left(p_{\mathbf{y}}(y)\right) = \sum_{k=0}^{\infty} c_k (-1)^k s^{-\left(\frac{n}{2}+k\right)}$. The lemma is proved by noting that $s^{-\left(\frac{n}{2}+k\right)}$ is Laplace transform of $y^{\frac{n}{2}+k-1}/\Gamma\left(\frac{n}{2}+k\right)$. Integrating the expression for $p_{\mathbf{y}}(y)$, we obtain the required expression for $F_{\mathbf{y}}(y)$.

Theorem 2. *The collision probability for the collision constraint formulated in (25) is given by*

$$P\left(\mathcal{C}_{\boldsymbol{x}_k, \boldsymbol{s}_k}\right) = \sum_{k=0}^{\infty} (-1)^k c_k \frac{y^{\frac{n}{2}+k-1}}{\Gamma\left(\frac{n}{2}+k\right)} \tag{49}$$

where $y = (r_1 + s_1)^2$.

Proof. From (25), the collision constraint is in the quadratic form $Q(\boldsymbol{y})$, with $\boldsymbol{w} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_k} - \boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{x}_k} + \Sigma_{\mathbf{s}_k}\right)$. We recall here that $\boldsymbol{w} = \mathbf{x}_k - \mathbf{s}_k$, where $\mathbf{x}_k$ and $\mathbf{s}_k$ are the random vectors that denote the robot and obstacle locations, respectively and are distributed according to $\mathbf{s}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{s}_k}, \Sigma_{\mathbf{s}_k}\right)$ and $\mathbf{x}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{x}_k}, \Sigma_{\mathbf{x}_k}\right)$. As noted before, the collision probability is the cdf of the quadratic form $Q(\boldsymbol{y})$. Thus from Lemma 4, the above theorem is proved.

4.3. Revisiting Convergence of the Series Expansion

As seen in Lemma 2, the cdf and the pdf of the quadratic form is convergent. In the following, we will derive upper bounds for the truncation error of the series expansions for the pdf and the cdf of the quadratic form.

If the infinite series pdf in (47) is truncated after N terms, the truncation error is

$$e(N) = \sum_{k=N+1}^{\infty} |c_k h_k(y)| = \left| \sum_{k=N+1}^{\infty} c_k \frac{y^{\frac{n}{2}+k-1}}{\Gamma\left(\frac{n}{2}+k\right)} \right| \tag{50}$$

Using Cauchy's inequality, we get

$$|c_k| \leq \frac{m(\rho)}{\rho^k}, \quad m(\rho) = \max_{|\theta|=\rho}|M(\theta)| \tag{51}$$

Thus we have

$$e(N) \leq \frac{m(\rho)}{\rho^k} \left| \sum_{k=N+1}^{\infty} \frac{y^{\frac{n}{2}+k-1}}{\Gamma\left(\frac{n}{2}+k\right)} \right| \leq$$
$$m(\rho) \left(\Gamma\left(\frac{n}{2}\right) N! \right)^{-1} \left(\frac{y}{2}\right)^{\frac{n}{2}-1} \left(\frac{y}{2\rho}\right)^{N+1} \exp\left(\frac{y}{2\rho}\right) \tag{52}$$

where we have used the gamma function identity, $\forall \varsigma > 0$, $\Gamma(\varsigma+1) = \varsigma\Gamma(\varsigma)$. In a similar manner, we obtain the truncation error for the infinite series cdf in (46)

$$E(N) \leq m(\rho) \left(\Gamma\left(\frac{n}{2}\right) (N+1)! \right)^{-1} \left(\frac{y}{2}\right)^{\frac{n}{2}} \left(\frac{y}{2\rho}\right)^{N+1} \exp\left(\frac{y}{2\rho}\right) \tag{53}$$
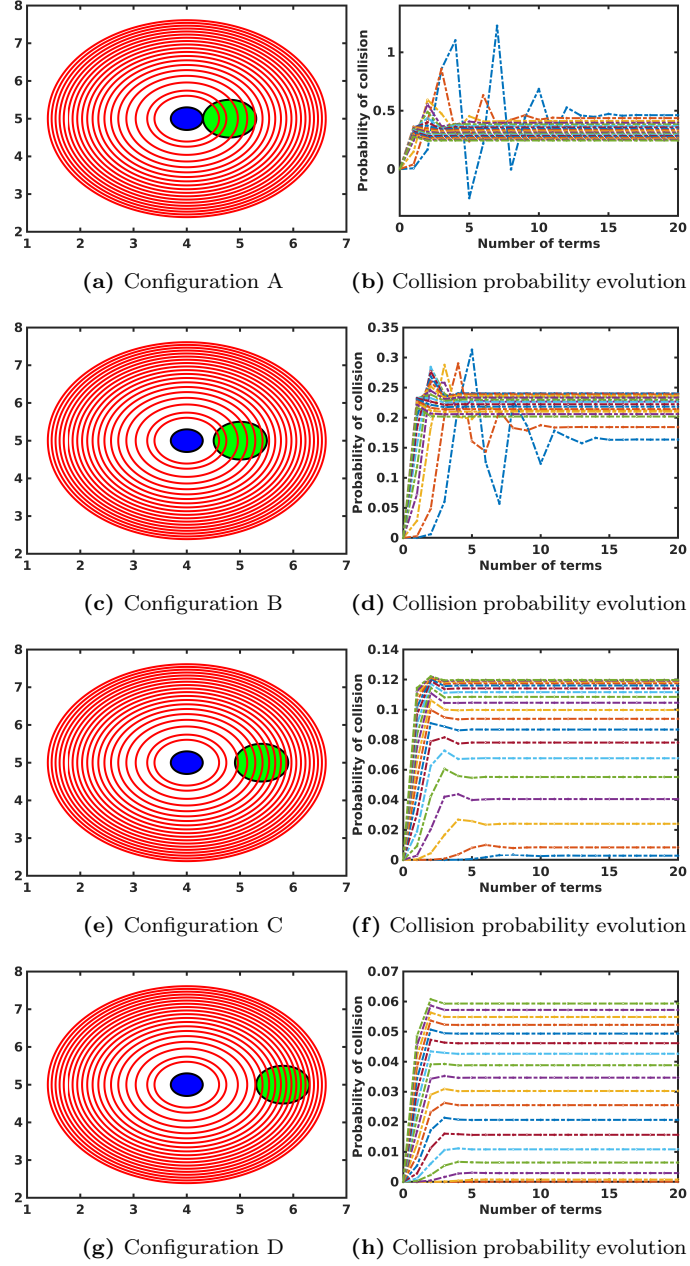
The expression for $m(\rho)$ is obtained from [35],

$$m(\rho) = \prod_{i=1}^{n} \lambda_i^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \sum_{i=1}^{n} \frac{b_i^2 \lambda_i}{\lambda_i + \rho} \right) \prod_{i=1}^{n} (1 - \frac{\rho}{\lambda_i})^{-\frac{1}{2}} \tag{54}$$
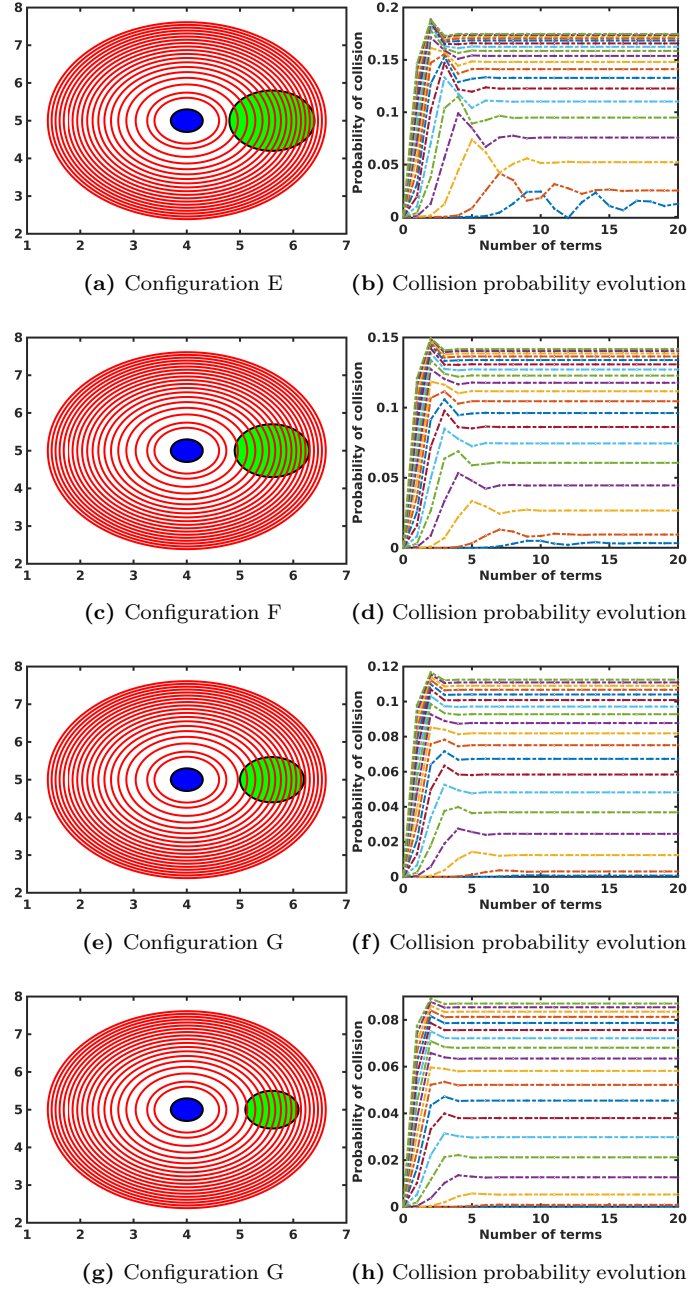
For the expression in (54) to be valid, it is required that $\rho < \lambda_i$ and therefore we have $\rho < \min \lambda_i$. As a result, $m(\rho)$ vanishes with $\sum_{i=1}^{n} b_i^2 \to \infty$. We recall here that $\boldsymbol{b} = P^T \Sigma^{-\frac{1}{2}} \boldsymbol{\mu} = (b_1, \ldots, b_n)^T$. Thus, larger the distance from the obstacles and lower the uncertainty in the robot and obstacle positions, the greater is the b_i value. In such scenarios, based on our experience, convergence is often attained within the first few terms of the series.

It is worth noting that for a given robot configuration and obstacle parameters, the varying term in (53) is $(y/2\rho)^{N+1}/(N+1)!$. This term is inversely proportional to the parameter ρ. As discussed above ρ depend on λ_i's, that is, the eigenvalues of $\Sigma_{\mathbf{x}_k} + \Sigma_{\mathbf{s}_k}$. Thus at time instant k, the parameter that

24

(a) Configuration A

(b) Collision probability evolution

(c) Configuration B

(d) Collision probability evolution

(e) Configuration C

(f) Collision probability evolution

(g) Configuration D

(h) Collision probability evolution

Figure 2: Different configurations for a robot of radius $0.3m$ and obstacle of radius $0.5m$. Different covariances are plotted as red circles. For each configuration the evolution of collision probability is plotted for different covariances in (b), (d), (f) and (h). In each of the 4 configurations, the maximum terms for convergence correspond to the minimum covariance of $diag(0.04, 0.04)$.

25

(a) Configuration E

(b) Collision probability evolution

(c) Configuration F

(d) Collision probability evolution

(e) Configuration G

(f) Collision probability evolution

(g) Configuration G

(h) Collision probability evolution

Figure 3: Different configurations for a robot of radius $0.3m$ and obstacle of radius (a) $0.8m$, (c) $0.7m$, (e) $0.6m$ and (g) $0.5m$. In the second column, for each of these configurations the evolution of collision probability is plotted against different covariances— the covariances are plotted as red circles in the figures on the left.

26

influences the convergence is the degree of uncertainty in both the robot and obstacle locations, that is, $\Sigma_{\mathbf{x}_k} + \Sigma_{\mathbf{s}_k}$. This is visualized for different configurations in Fig. 2. The blue and green circles represent a robot and an obstacle, respectively. The red ellipses corresponds to the 3σ uncertainties for different covariances $diag(0.04, 0.04)$, $diag(0.08, 0.08)$, $\ldots$, $diag(0.74, 0.74)$. For all the scenarios discussed we choose $E(N) = 0.001$. In Fig. 2(a) the robot and the obstacle are touching each other. For each of these covariances, the number of terms for convergence is shown in Fig. 2(b). The worst case corresponds to the covariance of $diag(0.04, 0.04)$, requiring 16 terms for convergence (dashed blue line with spikes in Fig. 2(b)). In Fig. 2(c) the distance between the robot and the obstacle is increased by $0.2m$ and the covariance $diag(0.04, 0.04)$ needs 12 terms for convergence. The distances are further increased by $0.4m$ and $0.8m$ in Fig. 2(e),(g) and their worst case convergences are 9 and 5 respectively, as seen in Fig.2(f),(h). The number of terms for the worst case convergence corresponds to covariance $diag(0.04, 0.04)$ and the respective timings for collision probability computation are shown in Table 1.

Similarly, the term $(y/2\rho)^{N+1}/(N + 1)!$ is directly proportional to y which quantifies the size of the robot and the obstacle. We recall here from (27) that $y = (r_1 + s_1)^2$, that is, the square of the sum of robot and obstacle radius. By keeping the robot size constant and varying the obstacle size, the influence of y on convergence is visualized for four different configurations in Fig. 3. In Fig. 3(a) $y = 1.1^2 (m^2)$ and convergence is obtained within 7 terms. In Fig. 3(c),(e),(g) we have $y = 1^2, 0.9^2, 0.8^2$ and the number of terms required for convergence are 4, 3 and 2, respectively. The collision probability computation times are as given in Table 2. For $y > 1.1^2$, it can be seen that the number of terms for convergence did not exceed 7 and for $y < 0.8^2$ convergence is achieved with the first two terms. Thus this shows that ρ plays a much larger role in convergence than y.

Table 1: The maximum number of terms required for convergence and the corresponding collision probability computation time. The values correspond to the covariance $diag(0.04, 0.04)$ for each of the configurations.

Configuration	Terms for convergence	Computation time (s)
A	16	0.0412 ± 0.0086
B	12	0.0044 ± 0.0041
C	9	0.0008 ± 0.0003
D	5	0.0004 ± 0.0002

Table 2: The maximum number of terms required for convergence and the corresponding collision probability computation time. Each configuration corresponds to different y values with the robot and obstacle locations remaining the same; only obstacle size varies.

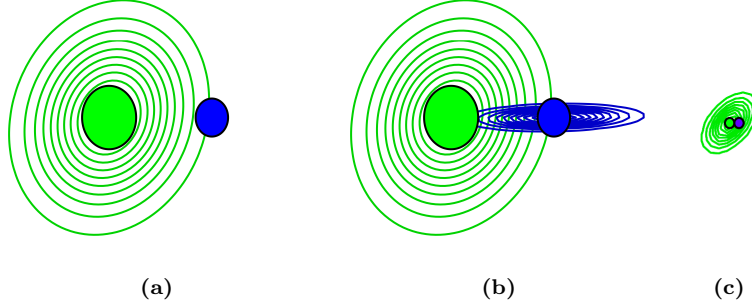| Configuration	Terms for convergence	Computation time (s)
E	7	0.0006 ± 0.0005
F	4	0.0004 ± 0.0002
G	3	0.0004 ± 0.0001
H	2	0.0001 ± 0.0000

4.4. Safe Configurations

In the presence of perception and motion uncertainty, providing safety guarantees for robot navigation is imperative. In this Section, we certify safety by defining the notion of a "safe" robot configuration. Let us assume that the obstacle position is known with high certainty as a result of perfect sensing, that is, no significant noise is present. However, since the true state of the robot is not known and only a distribution of these states can be estimated, collision checking has to be performed for this distribution of states. Moreover, in practice, the observations are noisy and this renders the estimated obstacle location (and shape) uncertain. Hence, this uncertainty should be taken into account while considering collision avoidance.

Given a robot configuration $\mathbf{x}_k$, we define the following notion of $\epsilon-$safe configuration.

Definition 2. *A robot configuration $\boldsymbol{x}_k$ is an $\epsilon-$safe configuration with respect to an obstacle configuration $\boldsymbol{s}$, if the probability of collision is such that $P\left(\mathcal{C}_{\boldsymbol{x}_k,\boldsymbol{s}}\right) \leq 1 - \epsilon$.*

For example, a $0.99-$safe configuration implies that the probability of this configuration colliding with the obstacle is at most 0.01. On the one hand, sampling-based motion planning approaches such as the Probabilistic Roadmap (PRM) [36] consider a discrete state space or a set of controls. As a result, it can only guarantee probabilistic completeness for returning $\epsilon-$safe configurations since the PRM motion planner is probabilistically complete [37], that is the probability of failure decays to zero exponentially with the number of samples used in the construction of the roadmap. As a result, for sampling-based BSP approaches [6, 3], the failure to find an $\epsilon-$safe configuration might be because such a configuration indeed does not exist or simply because there are not enough samples. On the other hand, continuous state and action space BSP approaches [4, 38, 39, 40] do not always guarantee $\epsilon-$safe configurations. This is merely because there might not be enough measurements to localize the robot or to estimate obstacle locations or both and hence this may preclude computing

29

Figure 4: Comparison of our approach with [20, 19, 18, 17]. (a) The robot state (in blue) is known perfectly, however the obstacle location (in green) is uncertain. The green ellipses denote the Gaussian uncertainty contours. (b) Robot state uncertainty is considered and the uncertainty contours are shown as blue ellipses. The approaches in [17, 18, 19] computes higher values. (c) Point-like robot and obstacle considered. The values computed with [17, 18] are much lower than expected while that of [19] is very high.

appropriate control commands.

4.5. Comparison to Other Approaches

[20] compute the collision probability by performing MCI. The joint distribution between the robot and the obstacle $p(\mathbf{x}_k, \mathbf{s}_k)$ is simplified as the product of the individual distributions. This MCI approach results in an expression with double summation for computing the probability of collision. [20] approximate this to a single summation expression to decrease computational complexity. Though this approximation compute values closer to the actual collision probability, it can either be bounded from below (when uncertainty is too large) or above. The approches in [17, 18, 19] compute upper bounds for collision probability. [19] compute an upper bound using Gaussian chance constraints. [18] compute the collision probability by finding the $\mathbf{x}_k$ that maximizes $p(\mathbf{x}_k, \mathbf{s}_k)$ and formulate the problem as an optimization problem with a Lagrange multiplier. Unlike in [18], which computes the maximum density, [17] use the density associated with the center of the robot. Yet, [17, 18] compute

30

Table 3: Comparison of collision probability approaches.

Case	Algorithm	Collision probability	Computation time (s)	Feasible
(a)	Numerical integral	4.62%	0.8648 ± 0.0418	Yes
	[20]	4.41%	0.0272 ± 0.0023	Yes
	[17]	5.84%	0.0017 ± 0.0002	Yes
	[18]	33.26%	0.2495 ± 0.3093	No
	[19]	9.60%	0.0021 ± 0.0003	No
	Our approach	4.61%	0.0254 ± 0.0034	Yes
(b)	Numerical integral	8.25%	1.1504 ± 0.0318	Yes
	[20]	7.87%	0.0325 ± 0.0024	Yes
	[17]	14.20%	0.0011± 0.0002	No
	[18]	36.31%	0.2156 ± 0.4068	No
	[19]	16.73%	0.0013 ± 0.0003	No
	Our approach	8.22%	0.0216 ± 0.0023	Yes
(c)	Numerical integral	14.82%	1.1341 ± 0.0211	No
	[20]	15.26%	0.0287 ± 0.0059	No
	[17]	0.46%	0.0015 ± 0.0007	Yes
	[18]	0.61%	0.3233 ± 0.5405	Yes
	[19]	50.00%	0.0018 ± 0.0007	No
	Our approach	14.83%	0.0280 ± 0.0093	No

lower values when the joint robot and obstacle covaraince is very small. We formulate the problem as exactly given in each of the works mentioned above to compare it with our approach[4]. The MCI approach of [20] is evaluated using $10,000$ samples. The numerical integration of the expression in (16) gives the exact collision probability value. Thus to validate the value computed using our approach, we perform the numerical integration of (16), using Monte Carlo method with $10,000$ samples.

Three different cases are considered as shown in Fig. 4. The solid green circle denotes an obstacle of radius $0.5m$ and its corresponding uncertainty contours are shown as green circles. The solid blue circle denotes a robot of radius $0.3m$ with the blue circles showing the Gaussian contours. We define a collision probability threshold of 0.09, that is, a $0.91-$safe configuration. The collision probability values and the computation times are summarized in Table 3. In Fig. 4(a), the robot position is known with high certainty. The numerical integration of (16) gave a value of 4.62% and hence the given configuration is a $0.91-$safe configuration. Our approach computes the collision probability as 4.61%, corroborating the exactness. The approach of [20] gave a close value of 4.41% but is a lower bound for the actual value. The other three approaches compute upper bounds as discussed previously. [17] estimates the configuration to be feasible, giving a collision probability value of 5.84%. The collision probability computed as given in [18] is 33.26% (not a $0.91-$safe configuration). Moreover, the value computed is almost seven times higher than the one computed using our approach. Similarly, the value computed using the approach in [19] is 9.60%, predicting the configuration to be unsafe. The higher values are due to the overly conservative nature of the estimates as loose upper bounds are computed. In Fig. 4(b), there is robot uncertainty along the horizontal axis and the numerical integration gave a collision probability value of 8.25%. As

---

[4]For comparison, the computation of other approaches have been reproduced to the best our understanding and the reproduced codes can be found here- https://bitbucket.org/1729antony/comparison/src/master/

compared to the previous case, the probability has almost doubled. This is quite intuitive as seen from the robot location uncertainty spread and hence there is greater chance for intersection between the two spheres. The collision probability computed using our approach is 8.22%. The increased chance for collision is also rightly communicated by the values computed using other approaches. The value computed using the approach in [18] gave a much higher value of 36.31%, an increase by 342% as compared to our approach. As in the previous case, the approaches in [17], [19] also gave higher values of 14.20% and 16.73%, respectively, while [20] gave a feasible value of 7.87% but a value lower than the actual.

The approaches in [17, 18] assume that the robot radius is negligible and that the obstacle size is relatively small compared to their location uncertainties. We also compute the collision probabilities for a robot and an obstacle with radius $0.05m$ each, where the robot and the obstacle are touching each other (Fig. 4(c)). The obstacle location is also much more certain, with the uncertainty reduced by 97% as compared to cases in Fig. 4(a),(b). The numerical integration gave a collision probability value of 14.82%. The probability of collision computed using our approach is 14.83%, whereas, using the approach in [18], the computed value is 0.61%. A lower value of 0.46% is obtained using the approach in [17]. As noted before, the lower values are a consequence of the covariance being very small. The approach of [19] gave an overly conservative estimate of 50%. The value computed using [20] is 15.26% , an upper bound to the actual value. To get a sense of the actual value, we compute the area of the covariance matrix, which is $6.28 \times 10^{-4} m^2$. This clearly indicates that 0.61%, 0.46% and 1.69% are too small values while 50.00% is a very high value. Our approach computes the exact probability of collision and outperforms the approaches in [20, 17, 18, 19].

We now provide a comparison in simulation using a scenario shown in Fig. 5(a). The robot has to reach the goal position (black star) by avoiding the obstacles in-between. To make the implications of overly conservative estimates [17, 18, 19]

explicit, we make the following assumptions. During each planning session[5], the robot can choose from a restrcitive set of nine different actions. A description of the motion and observation models can be found in Section 7. An action is chosen based on an additive cost of distance to the goal and the collision probability value. If the collision probability for an action is greater than 0.01, then the collision probability value for this action is penalized by redefining the value to be equal to a large number M. In this way, if all the actions lead to configurations with collision probability greater than 0.01, these actions are assigned a cost M. In reality this would mean that there exist no feasible plan and the robot would not proceed ahead. Thus if all actions are assigned a collision probability value of M, all these actions lead to collision and therefore we mark the trajectory as stopped. The trajectory executed by the robot using our collision probability computation approach is shown in blue (Path 1) in Fig. 5(b). The robot footprint (bounded circle) is also shown as the robot curves past the obstacle. The goal was reached in seven planning sessions. For the approaches in [17, 18, 19] the trajectory is stopped (Path 2) before the obstacle since all actions are assigned a value of M due to collision probability values greater than 0.01. As noted before, the action set from which the robot can choose an action is restriced and each action from this restricted set gives configurations with collision probability greater than 0.01. This is due to the fact that these approaches compute loose upper bounds and hence the values in the collision cost are redefined to M. The restrictive action set does not affect our approach as the exact value is computed and hence the robot reaches the goal safely. We now remove the restriction on the action set and all the other approaches are able to compute a path with a greater curve than Path 1. One such trajectory is shown in cyan (Path 3), with the collision probabilities computed using the approach by [19]. The planner is now able to choose an action

---

[5]By a planning session we mean an L look-ahead step planning at the current time and choosing an optimal control. Thus if n planning sessions are required to reach a goal this means that the control action was executed n times.
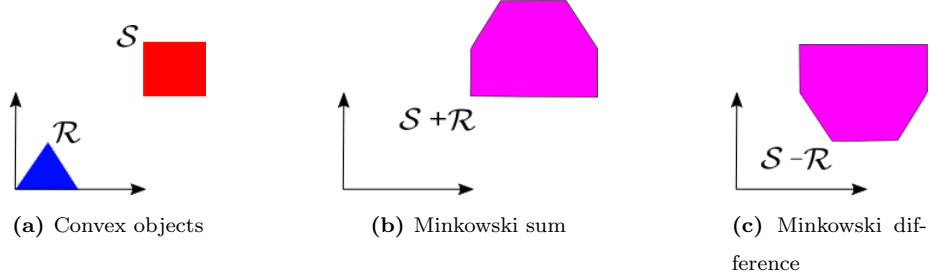
Figure 5: Comparison to other approaches in simulation: (a) Top view of the environment in gazebo. (b) Path 1 is the trajectory executed by the robot following our approach. The trajectory is executed in seven planing sessions. The robot footprint can be seen as it navigates past the obstacle. Path 2 leads to collision as upper bounds are computed by other approaches deeming the plans infeasible– robot is hence halted. The action set is extended and a longer trajectory is executed by the robot using the approach of [19]. The goal is reached in 15 planning sessions.

with collision probability less than 0.01. Thus it is seen that loose upper bounds for collision probability can lead to longer trajectories or in some cases deem all plans to be infeasible.

4.6. Non Circular Geometry

Given two objects (represented as convex polygons), in this Section we derive the collision constraint as a measure of the distance between the mid points of the objects. As before, let us consider two objects, $\mathcal{R} \subset \mathbb{R}^2$ and $\mathcal{S} \subset \mathbb{R}^2$. Let us assume that $\mathcal{S}$ is static and $\mathcal{R}$ can perform translational motions and is approaching $\mathcal{S}$. Then, subtracting $\mathcal{R}$ from $\mathcal{S}$ gives a convex polygon $\mathcal{P}$ such that for any $c \in \mathcal{P}$, then $\mathcal{R} \cap \mathcal{S} \neq \{\phi\}$ ([41]), that is, the convex polygon $\mathcal{P}$ is the set of configurations of $\mathcal{R}$ that leads to collision with obstacle $\mathcal{S}$. Note that, $\mathcal{R}$ and $\mathcal{S}$ are essentially two sets whose elements are the (x, y) pairs belonging to the respective polygons that they represent. Therefore, $\mathcal{P}$ is essentially the Minkowski difference between the two sets $\mathcal{R}$ and $\mathcal{S}$.

(a) Convex objects   **(b)** Minkowski sum   **(c)** Minkowski difference

Figure 6: (a) Two convex objects and the object formed by considering the (b) Minkowski sum and (c) Minkowski difference.

Definition 3. *The Minkowski sum of two sets $\mathcal{S}$, $\mathcal{R} \subseteq \mathbb{R}^d$ is*

$$\mathcal{S} + \mathcal{R} = \{s + r \mid s \in \mathcal{S}, \ r \in \mathcal{R}\} \tag{55}$$

Definition 4. *The Minkowski difference of two sets $\mathcal{S}$, $\mathcal{R} \subseteq \mathbb{R}^d$ is*

$$\mathcal{S} - \mathcal{R} = \{s - r \mid s \in \mathcal{S}, \ r \in \mathcal{R}\} \tag{56}$$

The Minkowski sum and difference of two objects are visualized in Fig. 6. The Minkowski difference between the two sets $\mathcal{S}$ and $\mathcal{R}$, also called the configuration space obstacle, is the set of (translational) configurations of $\mathcal{R}$ that brings it into collision with $\mathcal{S}$ ([41, 42]). However, we would like to obtain a collision constraint of the form (24). In order to obtain such a constraint, we first compute the Minkowski difference between the set $\mathcal{S}$ and the mid-point of $\mathcal{R}$. This gives a new convex set $\mathcal{P}'$ whose elements are formed by subtracting each element of the set $\mathcal{S}$ by the mid-point of object $\mathcal{R}$. In other words, the set $\mathcal{P}'$ is the set of all configurations of the mid-point of $-\mathcal{R}$[6] obtained by shifting/translating this point by each element in the set $\mathcal{S}$.

Lemma 5. *The maximum distance from a point P to any other point on a polygon $\mathcal{Q}$ is obtained by computing*

$$\sup \{PV_i \mid V_i \text{ is the vertex of } \mathcal{Q}\} \tag{57}$$

---

[6]It holds that $-\mathcal{R} = \{-r \mid r \in \mathcal{R}\}$

where PV_i denotes the line segment from point P to the vertex V_i.

Proof. Given a point inside (or outside) a polygon, farthest distance to any other point on the polygon is obtained when this point lies on the boundary of the polygon. It is known that the maximum distance from a point to a line segment occurs at the end-points of the line segment. Hence it is only sufficient to compute the distances to the vertices of the polygon and the maximum among them is the required distance.

For convex polygons $\mathcal{R}$ and $\mathcal{S}$, the boundary configurations of the Minkowski difference represents configurations that lead to contact between $\mathcal{R}$ and $\mathcal{S}$ ([42]), that is, the configurations where $\mathcal{R}$ and $\mathcal{S}$ touch each other. Also note that the polygon $\mathcal{P}'$ obtained by computing the Minkowski difference between the mid-point of $\mathcal{R}$ and the set $\mathcal{S}$ is fundamentally the set of all translated configurations of the mid-point of $-\mathcal{R}$ in the set $\mathcal{P} = \mathcal{S} - \mathcal{R}$. Thus the collision constraint is obtained by computing the maximum distance between the mid-point of the obstacle $\mathcal{S}$ and the polygon $\mathcal{P}'$.

Theorem 3. *Given a convex polygonal set $\mathcal{R}$ and an obstacle set $\mathcal{S}$, the collision constraint is given by*

$$\sup\left\{SV_i \mid V_i \text{ is the vertex of } \mathcal{P}'\right\} \tag{58}$$

where S is the mid-point of $\mathcal{S}$ and $\mathcal{P}'$ is the set obtained by computing the Minkowski difference between $\mathcal{S}$ and the mid-point of $\mathcal{R}$.

Proof. We saw above that the collision constraint is obtained by computing the maximum distance between the mid-point of the obstacle $\mathcal{S}$, that is S and the polygon $\mathcal{P}'$. From Lemma 5, the maximum distance is achieved at the vertices of the polygon. Hence, it follows from Lemma 5 that the collision constraint is $\sup\left\{SV_i \mid V_i \text{ is the vertex of } \mathcal{P}'\right\}$.

Thus, if $\mathcal{R}$ and $\mathcal{S}$ correspond to the set of points occupied by the robot and the obstacle, respectively, the collision constraint in (24) can be written as $\|\mathbf{x}_k - \mathbf{s}_k\|^2 \leq \left(\sup\left\{SV_i\right\}\right)^2$.
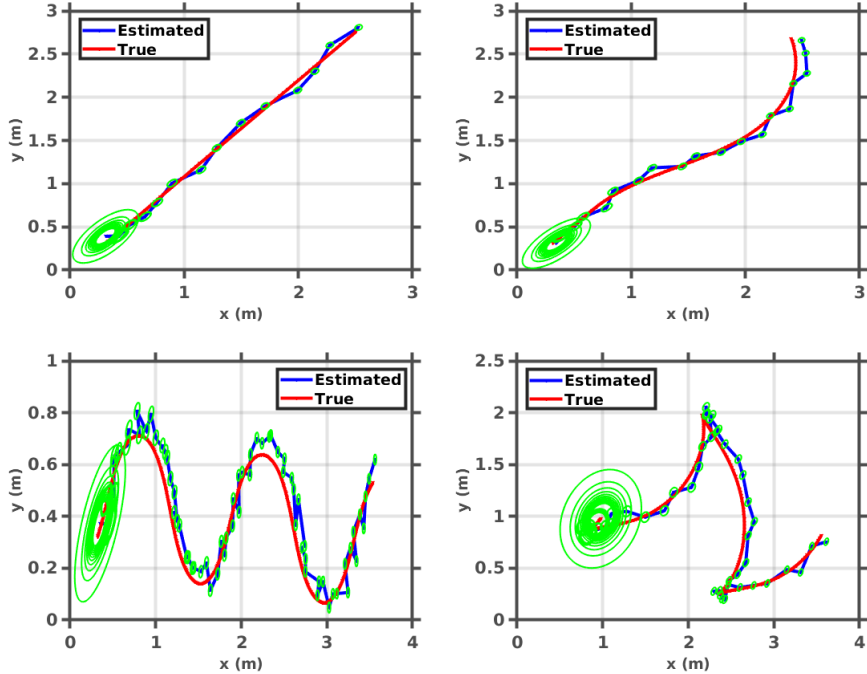
The Minkowski sum or difference are not invariant to rotations and hence rotation about a reference axis elicits different sets. The resulting sets are obtained by pre-multiplying the starting configuration with the standard rotation matrix of the corresponding angle. This renders different collision constraints for the two given sets. However, while planning for future control commands, the robot pose is often estimated using the motion model and by simulating possible future observations. As a result, an estimate of the robot orientation is computed. Moreover, for static obstacles, both in known and unknown environments, the geometry of the obstacle is a constant[7]. In the case of dynamic obstacles, the orientation of this geometry changes. Thus, assuming that the orientation of the obstacle is known and using the estimated robot orientation, the collision constrained is obtained as elucidated in Theorem 3.

4.7. Complexity Analysis

Finding a trajectory to the goal requires performing Bayesian (EKF) update operations. This involves performing matrix operations, that is, matrix multiplication and inversion of matrices. For a state of size n, the covariance matrix is of size $O(n^2)$. Therefore, each step of the Bayesian update has a complexity of $O(n^3)$. Let L denote the number of time steps in the trajectory or the look-ahead horizon, then the overall computational complexity is $O(n^3 L)$. Note that this is the complexity while computing the objective function at each time step. The number of times the computation is to be performed cannot be expressed beforehand as it depends on the specific application and objective to be achieved. Let us now analyze the complexity of collision probability computation. From (53) we see that for each iteration, the truncation error varies with $(y/2\rho)$. Therefore, to achieve $E(N) \leq \delta$, for an $\epsilon-$safe configuration, $k = O\left(\log \frac{\delta \rho}{y(1-\epsilon)}\right)$ iterations are required. We note that for each obstacle, the runtime is increased by this factor.

---

[7]In this work we assume non-deformable objects.

Figure 7: True obstacle trajectories plotted along with the estimated obstacle trajectories. In all the cases the linear velocity of the obstacle is greater than or equal to $0.5m/s$. A laser rangefinder is placed at the origin pointing towards the north-east direction. The green ellipses show the estimated covariances. Initial large ellipses correspond to the prior uncertainties. The prior uncertainties shrink as measurements are obtained due to obstacle detection.

5. Obstacle State Estimation

We adapt the approaches in [43, 18] and describe below the process for estimating future obstacle states. Let us consider that at time instant k, the robot at state is $\mathbf{x}_k$ and the estimated obstacle location is $\mathbf{s}$. Since the obstacle is following an unknown trajectory, the robot receives a series of measurements $\mathbf{z}_k^1, \ldots, \mathbf{z}_k^n$. Note that since the obstacle is moving, then each measurement $\mathbf{z}_k^i$ corresponds to a different location of the dynamic obstacle. Given the robot pose $\mathbf{x}_k$ and the measurement $\mathbf{z}_k^i$, the obstacle location $\mathbf{s}^i$ can be estimated

using the Bayesian approach,

$$p(\mathbf{s}^i|\mathbf{x}_k, \mathbf{z}_k^i) = \eta \; p(\mathbf{z}_k^i|\mathbf{x}_k, \mathbf{s}^i) \; p(\mathbf{s}^i|\mathbf{x}_k) \tag{59}$$

where $\eta = 1/p(\mathbf{z}_k^i|\mathbf{x}_k)$ is the normalization constant. Since the obstacle state $\mathbf{s}^i$ is independent of the robot state $\mathbf{x}_k$, we obtain

$$p(\mathbf{s}^i|\mathbf{x}_k, \mathbf{z}_k^i) = \eta \; p(\mathbf{z}_k^i|\mathbf{x}_k, \mathbf{s}^i) \; p(\mathbf{s}^i) \tag{60}$$

where $p(\mathbf{s}^i)$ is the prior density. Given the current robot belief $b[\mathbf{x}_k]$ and the measurement $\mathbf{z}_k^i$, the expression $p(\mathbf{z}_k^i|\mathbf{x}_k, \mathbf{s}^i)$ is computed using the measurement model (2). Therefore, the mean of the obstacle state $\mathbf{s}^i$ is obtained as

$$\bar{\mathbf{s}}^i = \arg\max_{\mathbf{s}^i} \; p(\mathbf{s}^i|\mathbf{x}_k, \mathbf{z}_k^i) \tag{61}$$

with the covariance matrix defined accordingly. Once n measurements are acquired at time k, we use it to estimate future obstacle states within in Model Predictive Control (MPC) strategy, where the robot plans for an optimal sequence of controls for L look-ahead steps. At each look-ahead step, the second term in (60), that is the obstacle belief has to be updated as per the obstacle motion model which is unknown. Given the state $\mathbf{s}^n$ obtained from the last measurement $\mathbf{z}_k^n$, the new state $\mathbf{s}'$ can then be predicted as

$$p(\mathbf{s}') = \int_{\mathbf{s}^n} p(\mathbf{s}'|\mathbf{s}^n) \; p(\mathbf{s}^n) \tag{62}$$

whose state space form is given by

$$\mathbf{s}(t+1) = A\mathbf{s}(t) + B\mathbf{u}(t) + \nu(t) \tag{63}$$

where $\mathbf{u}(t)$ is the control and $\nu(t)$ is the process noise and A, B are matrices which will be defined later. Now we discuss how this prediction can be achieved. From each estimated location $\mathbf{s}^i$ we can then compute the approximate velocities in the x and y directions using the forward difference method. Note that we assume that the obstacle does not change its velocity very drastically and that any two consecutive velocities differ by an $\varepsilon \ll 1m/s$. Therefore, given $\mathbf{s}^1, \ldots, \mathbf{s}^n$

we obtain the sets

$$\frac{\boldsymbol{\Delta}\mathbf{x}}{\Delta t} = \left\{ \frac{\bar{x}^2 - \bar{x}^1}{\Delta t}, \dots, \frac{\bar{x}^n - \bar{x}^{n-1}}{\Delta t} \right\} = \left\{ \frac{\Delta \bar{x}_1^2}{\Delta t}, \dots, \frac{\Delta \bar{x}_{n-1}^n}{\Delta t} \right\}$$

$$\frac{\boldsymbol{\Delta}\mathbf{y}}{\Delta t} = \left\{ \frac{\bar{y}^2 - \bar{y}^1}{\Delta t}, \dots, \frac{\bar{y}^n - \bar{y}^{n-1}}{\Delta t} \right\} = \left\{ \frac{\Delta \bar{y}_1^2}{\Delta t}, \dots, \frac{\Delta \bar{y}_{n-1}^n}{\Delta t} \right\} \quad (64)$$

where $\bar{x}^i, \bar{y}^i$ are the two components of $\bar{\mathbf{s}}^i$ and Δt is the time between two measurements. In a similar way we also compute the rate of change of the velocities in the x and y directions. From this computed sets, we choose the maximum change in velocities in both directions and denote the corresponding covariances[8] as Σ_v^x and Σ_v^y. From the Taylor series, each component of $\mathbf{s}'$ can be written as

$$x'(t') = x^n(t + \Delta t) \approx x^n(t) + \frac{\Delta \bar{x}_{n-1}^n}{\Delta t} \Delta t + \frac{1}{2} \frac{\Delta^2 \bar{x}_{n-1}^n}{\Delta t^2} \Delta t^2$$

$$y'(t') = y^n(t + \Delta t) \approx y^n(t) + \frac{\Delta \bar{y}_{n-1}^n}{\Delta t} \Delta t + \frac{1}{2} \frac{\Delta^2 \bar{y}_{n-1}^n}{\Delta t^2} \Delta t^2 \quad (65)$$

Note that the above equation is in the form of (63). The process noise is hence defined as

$$\nu(t) \sim \mathcal{N} \left(0, \begin{bmatrix} \frac{1}{4}\Sigma_v^x(\Delta t)^4 & 0 \\ 0 & \frac{1}{4}\Sigma_v^y(\Delta t)^4 \end{bmatrix} \right) \quad (66)$$

We use a 2D laser scanner to estimate the state of dynamic obstacles. It is assumed that the geometry of the obstacle is spherical and is known beforehand. From each scan of the laser rangefinder, the ray with the minimum distance r_j and the corresponding orientation is computed to form a measurement $\mathbf{z}_k^i$. This is repeated to obtain n distinct measurements. Given these measurements and the current robot state estimated using the standard EKF, the x and y components of the obstacle location are estimated. These estimated values are then used to compute the respective velocities using (64). The location estimates of the last scan $\mathbf{z}_k^n$ is then used as the prior in (62) to estimate future obstacle

---

[8]Note that since each variable is Gaussian, their differences are also Gaussian and the corresponding covariances can be computed trivially.

states. The respective mean and covariance are computed using (65) and (66). To illustrate our approach, in Fig. 7 we plot the true and estimated locations for different obstacle trajectories.

The approach is readily extended to estimate the state of all obstacles detected by the laser scanner. We note here that advanced strategies exits in the literature to efficiently segment laser rangefinder's scans, but it is not the main focus of the current paper. We therefore employ a rather simpler method sufficient to demonstrate the approach discussed herein. The laser rangefinder returns a sequence of distance measurements and these distances are less than the maximum range when obstacles are encountered. We assume that the obstacles are not too close, that is, there is a least one distance measurement between two obstacles that gives the maximum range. This discontinuity in the distance measurements between two obstacles allows us to separate the laser scanner measurements into different clusters belonging to different obstacles. From each cluster, we estimate the state of the corresponding obstacle. Note that it does not guarantee estimating the state of all the obstacles since some of them could be completely occluded by the others. It is also worth mentioning that estimating the location of static obstacles is a special case of the approach discussed here since for static obstacles both $\frac{\Delta \mathbf{x}}{\Delta t}$ and $\frac{\Delta \mathbf{y}}{\Delta t}$ equate to zero.

6. Objective Function

At each time instant k, the robot plans for L look-ahead steps to obtain a control policy $\mathbf{u}^\star_{k:k+L-1}$ given by

$$\mathbf{u}^\star_{k:k+L-1} = \underset{\mathbf{u}_{k:k+L-1}}{\arg\min} \, J_k(\mathbf{u}_{k:k+L-1}) \qquad (67)$$

where $J_k(\mathbf{u}_{k:k+L-1})$ is the objective function. As per the standard MPC, at each time step the first control command $\mathbf{u}^\star_k$ is then applied. At each time step, the robot is required to minimize its control usage and proceed towards the goal $\mathbf{x}^g$ avoiding collisions, while minimizing its state uncertainty. We quantify the state uncertainty by computing the trace of the marginal covariance of the

robot position. As a result, we have the following objective function

$$J_k(\mathbf{u}_{k:k+L-1}) \doteq \sum_{l=0}^{L-1} \left\|\xi(\mathbf{u}_{k+l})\right\|^2_{M_u} + tr\left(\|M_\Sigma\|^2_{\Sigma_{k+l}}\right) + M_C P(\mathcal{C}_{\mathbf{x}_{k+l},\mathbf{s}_{k+l}})$$

$$+ \underset{\mathbf{z}_{k+L}}{\mathbb{E}}\left[\|\mathbf{x}_{k+L} - \mathbf{x}^g\|^2_{M_g} + tr\left(\|M_\Sigma\|^2_{\Sigma_{k+L}}\right)\right] \quad (68)$$

where $\|x\|_S = \sqrt{x^T S x}$ is the Mahalanobis norm, M_u, M_g, M_C are weight matrices and $\xi(\cdot)$ is a function that quantifies control usage. The choice of weight matrices and the control function vary with the application. The term $tr\left(\|M_\Sigma\|^2_{\Sigma_k}\right) = tr\left(M_\Sigma^T \Sigma_k M_\Sigma\right)$ returns the marginal covariance of the robot location. Therefore, $M_\Sigma = \tau \bar{M}_\Sigma$, where τ is a positive scalar and $\bar{M}_\Sigma$ is a matrix filled with zero or identity entries. M_C penalizes the belief states with higher collision probabilities. Since future observations are not available at planning time and are stochastic, the expectation is taken to account for all possible future observations.

Our approach is summarized in Algorithm 1. At each time instant, the robot state is estimated using EKF (lines 4, 7). As described in the previous Section, obstacles are detected using a laser rangefinder. For the $j-$th detected obstacle, its future state is then estimated (line 8) using the approach discussed in Section 5. The total collision cost is then computed by adding the collision cost with each obstacle (line 10). Please note that if no $\epsilon-$safe configuration exists then the algorithm terminates. Finally the total cost is computed as given in (68). This is repeated for each horizon step to obtain the optimal control policy $\mathbf{u}^\star_{k:k+L-1}$. The control command $\mathbf{u}^\star_k$ is then applied and the process is repeated till the goal is reached.

7. Experiments

In this Section we describe our implementation and then illustrate and explore the capabilities of our proposed approach. First, we present a theoretical example to conceptually understand the proposed approach. Next, we consider both single and multi-robot experiments, which are performed using different

43

Algorithm 1: Safe motion planning.

    input : $b[\mathbf{x}_k], L, \epsilon, N$, radii, $\mathbf{z}_k^1, \ldots, \mathbf{z}_k^n$

1 $J_k = 0$, $l = 0$

2   compute $p(\mathbf{s}^i|\mathbf{x}_{k+l}, \mathbf{z}_{k+l}^i) \; \forall i, 1 \leq i \leq n$ and $\frac{\Delta \mathbf{x}}{\Delta t}, \frac{\Delta \mathbf{y}}{\Delta t}$

3   **while** *true* **do**

4 $b[\mathbf{x}_{k+l+1}^-] \leftarrow b[\mathbf{x}_{k+l}]p(\mathbf{x}_{k+1+l}|\mathbf{x}_{k+l}, \mathbf{u}_{k+l})$

5 $\{\mathbf{z}_{k+l+l}\} \leftarrow$ simulate future observations

6   **for** *each* $\{z_{k+l+l}\}$ **do**

7   compute $b[\mathbf{x}_{k+l+1}]$

8   predict obstacle state $\mathbf{s}_{k+l}$ (using (65))

9   compute $\sum_j P(\mathcal{C}_{\mathbf{x}_{k+l+1}, \mathbf{s}_{k+l}})$

10   compute total_cost (using (68))

11   **end**

12 $J_k \leftarrow J_k +$ total_cost

13   **end**

14 $\mathbf{u}_{k:k+L-1}^{\star} \leftarrow \arg\min_{\mathbf{u}_{k:k+L-1}} J_k(\mathbf{u}_{k:k+L-1})$
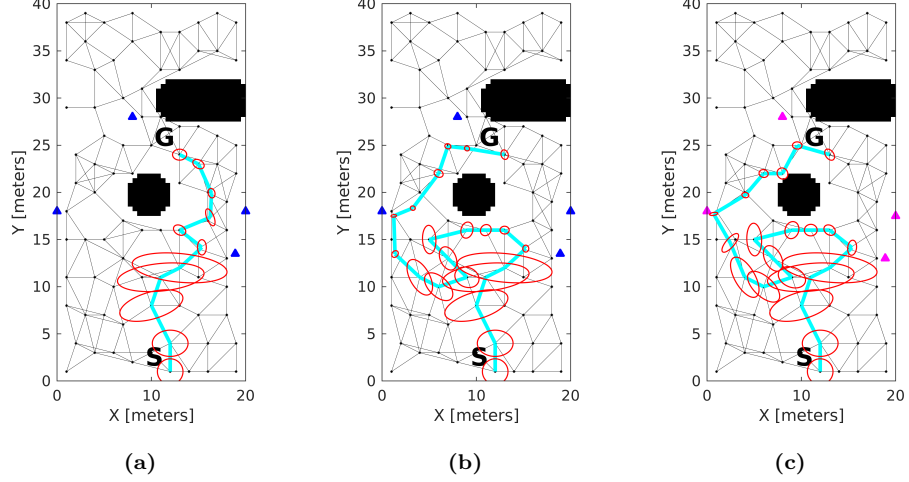
15   **return** $\mathbf{u}_{k:k+L-1}^{\star}$

Gazebo-based realistic simulations. For all the experiments we use a TurtleBot3 Waffle robot with a radius of $0.22m$. The robot is equipped with a Laser Distance Sensor LDS-01 and we use the same to acquire obstacle range and bearing. The performance is evaluated on an Intel® Core i7-6500U CPU@2.50GHz×4 with 8GB RAM under Ubuntu 16.04 LTS. In all the Gazebo based experiments, the initial uncertainty in robot pose is $\Sigma_0 = diag(0.1m, 0.1m, 0.02rad)$. The LDS detections/measurements are only from the obstacles whose motion is unknown and the EKF is employed to predict the robot state at each time step. The ground truth odometry from Gazebo is used to measure the pose of the robot, mimicking a motion capture system. This measurement is then corrupted with noise to perform state estimation. However, this estimation is not performed at each time step and we randomly select the times steps to carry out

the same. In this way we explore the robustness of our approach to localization uncertainties.

Remark 1. *We note here that comparison to other approaches have been provided in Section 4.5 and the computation of collision probability with these approaches have been reproduced to the best our understanding. While this may be accurate for static scenarios as demonstrated in 4.5, we believe that extending this comparison to online planning scenarios would not be an accurate portrayal of these works. For example, the work in [18] finds the position with the maximum probability by formulating it as an optimization problem. [19] require linearizing the collision constraint and computation of the inverse of the standard error function. There are a number of ways to perform the optimization, the linearization or the computation of the error functions. So unless we know the exact methods used by these approaches, extending the comparisons to online planning would lead to an inaccurate depiction of these approaches. We thus limit the comparison to these approaches to static scenarios presented in Section 4.5. The approaches in [17, 18, 19] compute upper bounds for collision probability and if these methods are employed, we expect them to produce longer paths than the ones depicted in this section.*

7.1. Theoretical Example

We consider the case of a mobile robot navigating in a 2D environment of $20m \times 40m$. Fig. 8 shows the underlying PRM graph (sampled nodes in green, connected by edges) with 90 nodes. In this domains, the robot (radius $0.3m$), starting from its initial belief state (mean pose denoted by S in the figure) has to reach the node $\mathbf{x}_g$ (G in the figure), minimizing its cost function (68). The blue/magenta triangles denote the landmarks in the environment and the solid black blobs represent the obstacles in the environment. The red ellipses denote the 3σ covariances (only the (x,y) portion is shown). Unless otherwise mentioned, in all the experiments, $0.99-$safe configurations are solicited and the total planning time is the average time for 25 different runs.
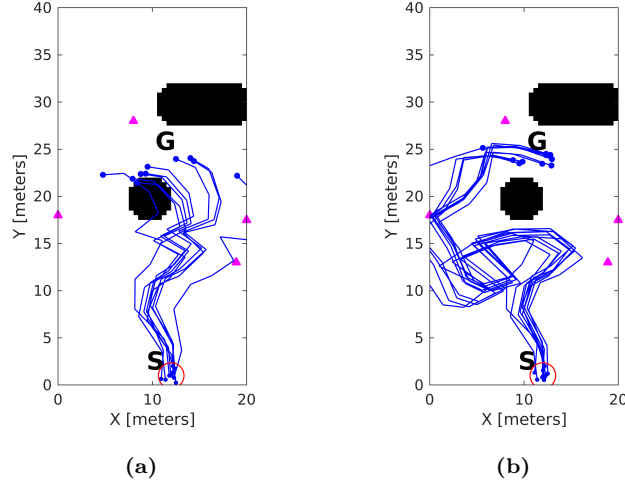
Figure 8: Trajectory and the covariance evolution for single planning for the 2D environment. (a) Plan obtained when object uncertainty is not considered. (b) The planned trajectory when object uncertainty is considered (c) Planned trajectory with true landmark locations.

The state $\mathbf{x}_k \doteq (x_k, y_k, \theta_k)$ is the robot pose (position and orientation) at time k. The applied control vector $\mathbf{u}_k \doteq (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$ consists of an initial rotation δ_{rot1}, followed by a translation of δ_{trans} and a final rotation of δ_{rot2}, orienting the robot in the required direction. As a result, the following non-linear dynamics is obtained ([31])

$$
\begin{aligned}
x_{k+1} &= x_k + \delta_{trans}\cos(\theta_k + \delta_{rot1}) \\
y_{k+1} &= y_k + \delta_{trans}\sin(\theta_k + \delta_{rot1}) \\
\theta_{k+1} &= \theta_k + \delta_{rot1} + \delta_{rot2}
\end{aligned}
\tag{69}
$$

For robot localization, we consider a landmark based measurement model that returns the range and bearing. The measurement model with noise is thus
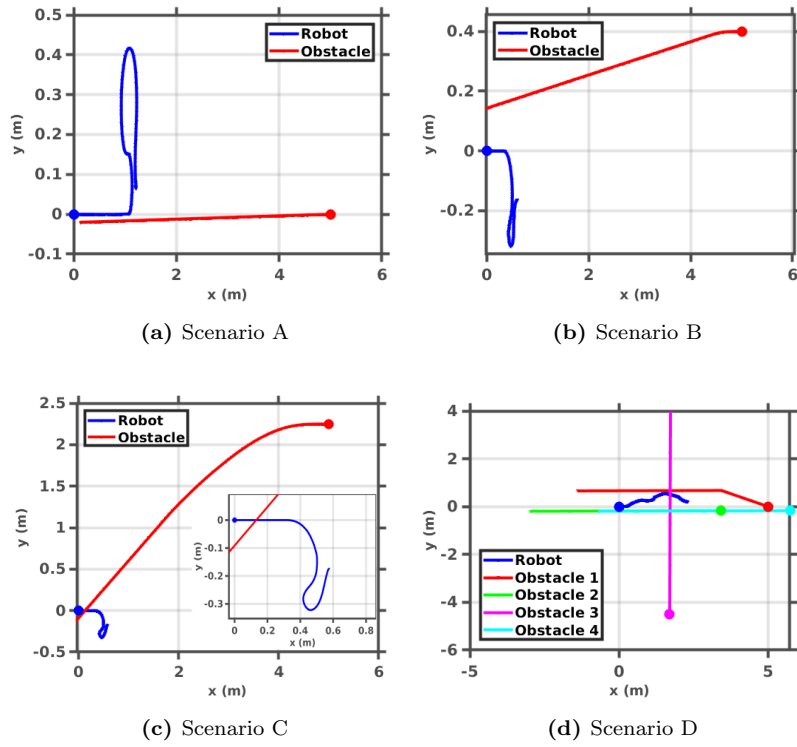
Figure 9: Execution traces of robot's true state across ten simulation with initial state drawn from the known initial belief. (a) Computed control when object uncertainty not considered is followed. (b) Traces of robot's true state while following the computed control considering object uncertainty.

obtained as

$$
\mathbf{z}_k = \begin{bmatrix} r_k^i = \sqrt{(O_k^i(1) - x_k(1))^2 + O_k^i(2) - x_k(2))^2} \\ \\ \phi_k^i = \arctan(\frac{O_k^i(2) - x_k(2)}{O_k^i(1) - x_k(1)}) - x_k(3) \end{bmatrix} + v_k \ , \ v_k \sim \mathcal{N}(0, Q_k) \ (70)
$$
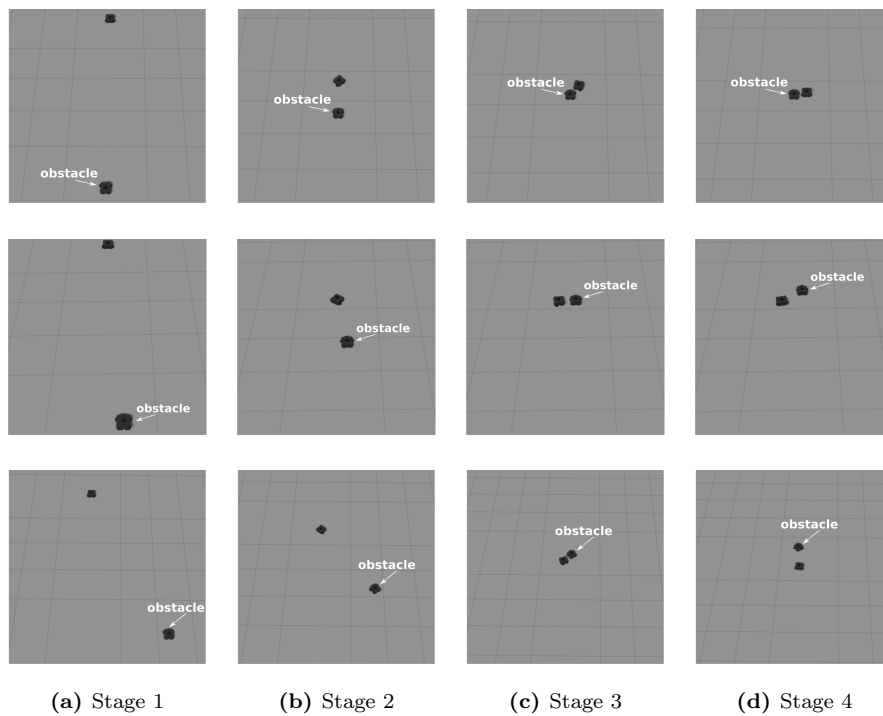
where r_k^i and ϕ_k^i are the range and bearing of the i-th object O_k^i (at time k) relative to the robot frame.

The mean landmark locations are $(0, 18), (8, 28), (20, 18), (18.9, 13.5)$. The landmarks at $(20, 18), (18.9, 13.5)$ are not precisely known and has an associated uncertainty of $diag(0.02, 0.02)$ in each of their locations. We first neglect the uncertainty and plan using the mean landmark locations. The planned trajectory is seen in cyan in Fig. 8(a) and the associated beliefs are seen in red. The overall planning time is $0.0041s(\pm 0.0003s)$. We note here that the overall planning time also includes the collision probability computation time. Next,

47

(a) Scenario A

(b) Scenario B

(c) Scenario C
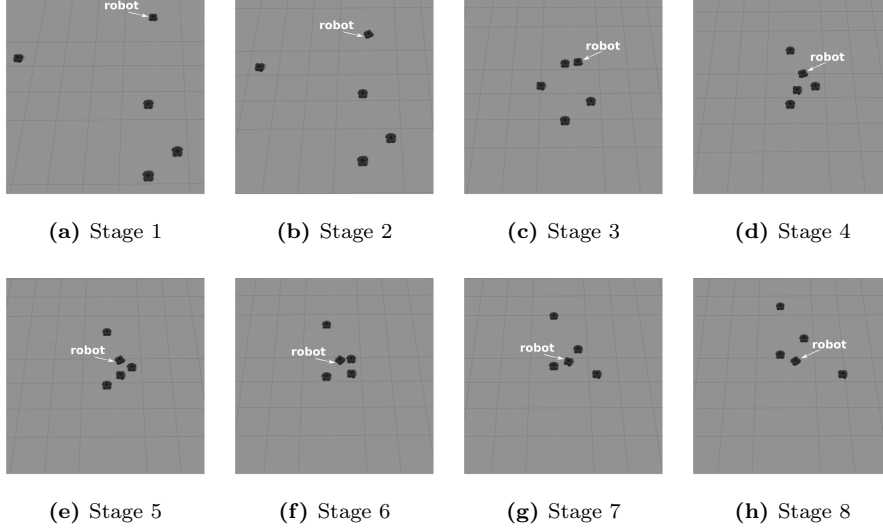
(d) Scenario D

Figure 10: Top view of robot and obstacle trajectories are plotted with the starting locations marked as round blobs. The robot trajectory is shown in blue. (a) Single obstacle with velocity of $0.5m/s$. (b) Obstacle velocity is $1.0m/s$. (c) Obstacle velocity is $2.5m/s$ and the zoomed figure is shown in the inset. (d) Four obstacles with different velocities.

48

(a) Stage 1   **(b)** Stage 2   **(c)** Stage 3   **(d)** Stage 4

Figure 11: Top view snapshots of the robot and the obstacle at four different stages (from left to right) of the experiment in scenarios A (row 1), B (row 2) and C (row 3) shown in Fig. 10. Positive x-axis is vertically downwards.

49

(a) Stage 1   **(b)** Stage 2   **(c)** Stage 3   **(d)** Stage 4

(e) Stage 5   **(f)** Stage 6   **(g)** Stage 7   **(h)** Stage 8

Figure 12: Top view snapshots of the robot and the obstacles in the Gazebo environment at different stages of the experiment in scenario D shown in Fig. 10.

we consider the landmark uncertainty during planning. The planned trajectory and the associated beliefs are seen in Fig. 8(b). We note here that there is a significant change in the planned trajectory. The total planning time in this case is $0.0042s(\pm0.0008s)$. Finally, we plan using the true landmark locations of $(0, 18), (8, 28), (20, 17.5), (18.9, 13)$ which are seen in magenta in Fig. 8(c). The overall planning time is $0.0044s(\pm0.0011s)$. As seen in the figure, the planned trajectory is similar to the case when landmark uncertainty is considered. However, executing the plan synthesized by not considering the landmark uncertainty (scenario in Fig. 8(a)) would lead to collision and larger goal state covariance. This is visualized in Fig. 9. The traces of true robot state across ten simulations while executing the plan synthesized by neglecting object uncertainty (scenario in Fig. 8(a)) is shown in Fig. 9(a). The initial state is sampled from the known initial belief (plotted as red circle in figure) and 60% of the executions lead to collision. Fig. 9(b) shows the traces of true robot state across ten simulations while executing the computed control policy by considering ob-

ject uncertainty (scenario in Fig. 8(b)). Therefore, not considering the object uncertainty lead to localization errors and thereby synthesize inefficient plans.

7.2. Single-robot Scenarios

In this Section, we discuss our collision avoidance approach considering single-robot scenarios in the Gazebo simulator. Dynamic obstacles are simulated using different robots whose motion model is unknown to the considered robot. The robot kinematics is as follows
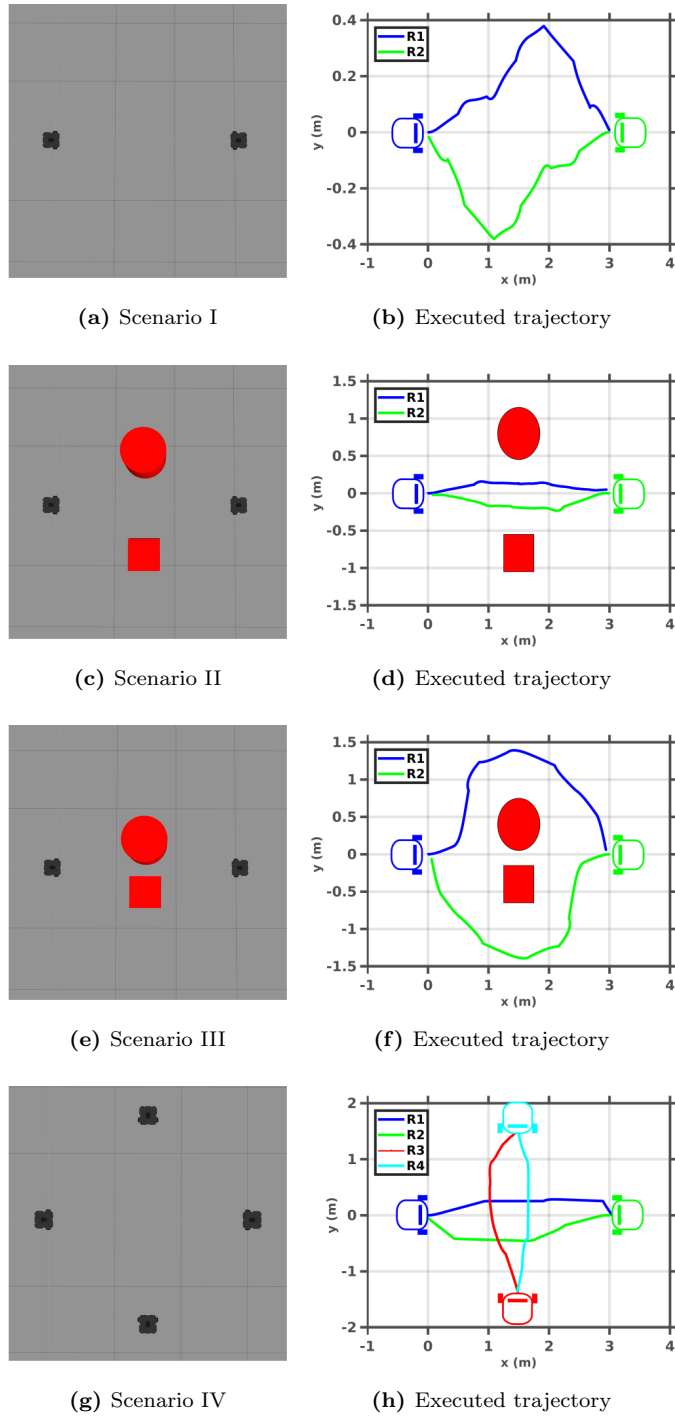
$$\mathbf{x}_{k+1} = \begin{bmatrix} x_k - \frac{V_k}{\omega_k}\sin(\theta_k) + \frac{V_k}{\omega_k}\sin(\theta_k + \omega_k \Delta t) \\ y_k + \frac{V_k}{\omega_k}\cos(\theta_k) - \frac{V_k}{\omega_k}\cos(\theta_k + \omega_k \Delta t) \\ \theta_k + \omega_k \Delta t \end{bmatrix} + w_k \qquad (71)$$

where the applied control $\mathbf{u}_k = (V_k, \omega_k)^T$ is made up of the linear and angular velocities and w_k is the noise as defined in Section 1. We define the prior uncertainty in the obstacle location as $diag(0.1m, 0.1m)$ and corrupt the range data returned by LDS with varying noise with variance $0.1 \times rand(1)m^2$. By default, $0.99-$safe configurations are solicited and we use a look-ahead horizon of $L = 7$. Since the TurtleBot3 robot is used, the collision constraint is $\|\mathbf{x}_k - \mathbf{s}_k\|^2 \leq (0.22 + 0.22)^2$. In each experiment, we consider a robot starting from the location $(0,0)$ and having to reach the goal location of $(3,0)$, subject to minimizing the objective function in (68).

First, we consider three scenarios $(A, B$ and $C)$ where the robot has to avoid head-on collisions with dynamic obstacles. The obstacle linear velocities in each scenario are $0.5m/s$, $1.0m/s$ and $2.5m/s$, respectively. This is however unknown to the robot and at each time step obstacle states are estimated using the approach detailed in Section 5. The robot and obstacle trajectories for the three scenarios are shown in Fig. 10(a)-(c). Note that evading a collision is the main focus of these experiments and hence only the relevant trajectories are plotted. Snapshots of four different stages during each the trajectory execution are shown in Fig. 11. The first row corresponds to scenario A, the second to scenario B and the third row displays snapshots of scenario C. For all the scenarios, stage 1

shows the initial configuration of the robot and the obstacle. Once the obstacle is detected, a control command for evading the obstacle is computed to move towards a $0.99-$safe configuration. The beginning of execution of such a control command is seen in stage 2. Stage 3 shows the snapshot when the obstacle and the robot are very close to each other with the robot evading the obstacle to avoid collision. In stage 4 it is seen that the robot has successfully avoided collisions. The video accompanying this paper demonstrates these results. For each scenario, the experiment is performed 50 times and the average time for computing the associated collision probability is shown in the first three rows of Table 4. The last row corresponds to Scenario D, a multi-obstacle scenario which will be described soon. As a safety metric, the minimum distance between the two robots is also measured and the results are shown in Table 4. For all the scenarios a success rate of 100% is achieved, that is, in all the 50 experiments, there were no collisions. However, lower look-ahead horizon, that is, $L < 7$ did not give 100% success rate as most often the obstacles were too close before executing the appropriate control command. Another parameter that affects the success rate is the value of ϵ. For example, a $0.4-$safe configuration always resulted in collision for scenarios B and C.

In scenario D, we consider four obstacles, each with different velocities. The robot successfully evades collision with all the four obstacles and the results are shown in the last row of Table 4. The trajectories of the robot and the obstacles can be seen in Fig. 10(d). Aerial snapshots at different time instants are shown in Fig. 12. Stage 1 corresponds to the initial configuration of the robot and the obstacles. Stages 2 and 3 show the robot moving to evade a head-on collision with obstacle 2 (obstacle numbers in Fig. 10(d)). Stages 4 through 7 show different instances while the robot tries to evade the remaining obstacles. Finally, in stage 8, the robot has successfully avoided potential collisions. The mean computation time for collision probability is $0.3682s$ and the computation time of the entire framework is $0.4230s$. The entire framework time includes the time for collision probability computation, uncertainty propagation, and obstacle state estimation.

(a) Scenario I

(b) Executed trajectory

(c) Scenario II

(d) Executed trajectory

(e) Scenario III

(f) Executed trajectory

(g) Scenario IV

(h) Executed trajectory

Figure 13: Different multi-robot scenarios and the corresponding trajectories executed by the robots.

53

Table 4: The minimum distance between the robot and the obstacles and the collision probability computation time for four different scenarios. The minimum distance corresponds to the minimum among all the distances between robots and the obstacles.

Scenario	Minimum distance (m)	Collision probability computation time (s)
A	0.16	0.0267 ± 0.0078
B	0.31	0.0189 ± 0.0074
C	0.12	0.0191 ± 0.0072
D	0.20	0.0368 ± 0.0023

7.3. Multi-robot Scenarios

In this Section we demonstrate our approach with multi-robot planning scenarios. In this setting, each robot considers all other robots as dynamic obstacles. However, there is no communication between the robots and the obstacle/robot states are estimated using the approach described in Section 5.

We first consider different scenarios with two robots. The initial pose of the robots are $(0,0,0)$ and $(3,0,-\pi)$ and the goal for each robot is to navigate towards the starting location of the other robot. The starting configuration and the executed trajectory of scenario I can be seen in Fig. 13(a), (b). Scenario II, which includes a cube and a cylinder as static obstacles, is shown in Fig.13(c), (d). It can be seen that the robots evade collision with each other and the static obstacles and navigate between the obstacles. The locations of the static obstacles are unknown to the robots and they are estimated using the approach discussed in Section 5. However, we assume known data association and we apply the collision constraint derived in Theorem 3. The obstacles in scenario II are pulled closer in scenario III (Fig. 13(e),(f)) to prevent the robots from passing between the obstacles. This is rightly estimated by the robots and they navigate around the obstacles to reach the goal. However it was seen that for $L < 7$, both robots turned to the same side and 20% (10 out of 50) of the time this leads to collision. This is so because, as the robots turn to the side of the

Table 5: Minimum distance between the robot and the obstacles in four scenarios and the corresponding collision probability computation time.

Scenario	Minimum distance (m)	Collision probability computation time (s)
I	0.33	0.0117 ± 0.0044
II	0.08	0.0137 ± 0.0123
III	0.51	0.0099 ± 0.0013
IV	0.10	0.0211 ± 0.0052

cube, the cube occludes one robot from the other. By the time each robot turns around the cube and see the other, they are already too close to avoid collision. In scenario IV (Fig. 13(g),(h)), we consider four robots, where the robots facing each other are required to swap their positions. The initial poses of each robot are $(0, 0, 0)$, $(1.5, -1.5, \frac{\pi}{2})$, $(3, 0, -\pi)$ and $(1.5, 1.5, -\frac{\pi}{2})$, respectively.

Table 5 shows the statistics for the four scenarios discussed above. The minimum distance between the robot and the obstacle and the average computation time for evaluating the collision probability are reported. In scenario IV, it was seen that $\epsilon < 0.99$ leads to collision in 80% of the experiments. For the other scenarios, $\epsilon < 0.9$ successfully evaded collision in all the experiments.

8. Discussion

In Section 4.5, we have compared our approach to other similar techniques [20, 17, 18, 19] and it is seen that our approach outperforms them. In this section we outline few limitations and discuss how to overcome them by proposing suitable extensions. These extensions would enhance the capability and robustness of our approach in challenging scenarios.

In Section 5, we have modelled the object uncertainty as a Gaussian. The assumption is justified in the case of Gaussian belief states, and works for all practical situations. Yet, in general, the model might not be Gaussian and it has to be determined based on the environment, sensing model and the robot

55

task that needs to be achieved. Finding an appropriate model, especially for non-Gaussian belief states is a work for the future.

The collision probability approach discussed in this paper is not restrictive to mobile robots and is readily extended to any 3D rigid body robot. For example, a quad-rotor can be approximated using a minimum volume enclosing sphere and therefore our approach can be used directly. Similarly, in the a manipulator robot each link is approximated by minimum volume bounding spheres that tightly enclose the link. For such robots, the collision with an obstacle has to be checked for each bounding volume. For example, let us consider a manipulator robot with l bounding spheres. Then the collision condition for the $i-$th sphere is given by $\mathcal{C}_{\mathbf{x}_k^i, \mathbf{s}_k}$, where $\mathbf{x}_k^i$ is the center of the $i-$th sphere. Furthermore, an alternative and more appropriate approach is to consider the minimum-volume enclosing ellipsoid for each link [44]. For every convex polyhedron, there exists a unique ellipsoid of minimal volume that contains the polyhedron and is called the *Löwner-John ellipsoid* of the polyhedron [45]. Thus each link can be represented by their corresponding Löwner-John ellipsoids. However, the collision condition in (24) is no longer valid. The collision condition should be reformulated using the distance between two ellipsoids. Please note that the representation using Löwner-John ellipsoid is also extended to the 3D obstacles.

While formulating the objective function in Section 6, we assume that the set of actions from which the robot can plan its future control is known *a priori*. In other words, a finite action set is considered. This justifies the inclusion of the collision cost term $P(\mathcal{C}_{\mathbf{x}_{k+l}, \mathbf{s}_{k+l}})$ in (68). However, our approach is not limited to any specific set of actions or trajectories. The general approach would be to include the set of all possible control actions. The objective function in (68) is then reformulated as an optimization problem with the collision cost term included as a constraint to keep the collisions within the $1 - \epsilon$ bound.

9. Conclusion

In this paper, we have presented an approach that incorporates reasoning regarding the landmark uncertainties within the BSP framework. We consider a Gaussian parametrization of the belief dynamics and derive the corresponding mean and covariance of the belief state when the object uncertainty is considered. We also analyse the effect of adding the object uncertainty for belief estimation and provide the conditions when the effect is negligible. Furthermore, we present a novel approach to compute an exact expression for the collision probability when the robot and obstacle states are uncertain. In contrast, existing works compute an approximation of the actual collision probability. The collision condition is formulated as a quadratic form in random variable and the associated collision probability is the cdf of the quadratic from. We derive the cdf is derived as an infinite series and we prove its convergence and provide an upper bound for the truncation error. We further relax the spherical geometry (of robot and obstacles) assumption by considering the exact convex footprints of the robot and the obstacles and derive the collision constraints for convex polygons. A method to estimate the states of dynamic obstacles and further estimate its future states to enable non-myopic planning is also discussed. Gazebo based simulation using single and multi-robot scenarios with both static and dynamic obstacles demonstrate the real-time online capability of our approach. We further discuss the limitations of our approach and delineate possible directions for future work.

APPENDIX A: Derivation of (11)

The mean of $b[\mathbf{x}_{k+1}]$ is the value that minimizes $\mathcal{J}_{k+1}$, and therefore it is obtained by equating its first derivative to zero. The first derivative of $\mathcal{J}_{k+1}$ with respect to $\mathbf{x}_{k+1}$ is obtained as

$$\frac{\partial \mathcal{J}_{k+1}}{\partial \mathbf{x}_{k+1}} = -H_{k+1}^T Q_{k+1}^{-1} \left(\mathbf{z}_{k+1} - h(\bar{\boldsymbol{\mu}}_{k+1}) - H_{k+1}(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1}) \right) +$$

$$\Sigma_{O_{k+1}^i}^{-1} \left(\mathbf{x}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) + \bar{\Sigma}_{k+1}^{-1} \left(\mathbf{x}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) \quad (72)$$

Since we are evaluating an expression for the mean, we will substitute $\boldsymbol{\mu}_{k+1}$ for $\mathbf{x}_{k+1}$. Thus setting the first derivative of $\mathcal{J}_{k+1}$ to zero, we have

$$
\begin{aligned}
H_{k+1}^T Q_{k+1}^{-1} \left(\mathbf{z}_{k+1} - h(\bar{\boldsymbol{\mu}}_{k+1}) \right) &= H_{k+1}^T Q_{k+1}^{-1} H_{k+1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \\
&\quad \Sigma_{O_{k+1}^i}^{-1} \left(\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) + \bar{\Sigma}_{k+1}^{-1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) \\
&= H_{k+1}^T Q_{k+1}^{-1} H_{k+1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \bar{\Sigma}_{k+1}^{-1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \\
&\quad \Sigma_{O_{k+1}^i}^{-1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} + \bar{\boldsymbol{\mu}}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) \\
&= H_{k+1}^T Q_{k+1}^{-1} H_{k+1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \bar{\Sigma}_{k+1}^{-1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \\
&\quad \Sigma_{O_{k+1}^i}^{-1} \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \Sigma_{O_{k+1}^i}^{-1} \left(\bar{\boldsymbol{\mu}}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) \\
&= \left(H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \bar{\Sigma}_{k+1}^{-1} + \Sigma_{O_{k+1}^i}^{-1} \right) \left(\boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} \right) + \\
&\quad + \Sigma_{O_{k+1}^i}^{-1} \left(\bar{\boldsymbol{\mu}}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) \quad (73)
\end{aligned}
$$

From (10) we have $\Sigma_{k+1}^{-1} = H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \bar{\Sigma}_{k+1}^{-1} + \Sigma_{O_{k+1}^i}^{-1}$. Also using the fact that $K_{k+1} = \Sigma_{k+1} H_{k+1}^T Q_{k+1}^{-1}$, (73) simplifies to

$$
K_{k+1} \left(\mathbf{z}_{k+1} - h(\bar{\boldsymbol{\mu}}_{k+1}) \right) = \boldsymbol{\mu}_{k+1} - \bar{\boldsymbol{\mu}}_{k+1} + \Sigma_{k+1} \Sigma_{O_{k+1}^i}^{-1} \left(\bar{\boldsymbol{\mu}}_{k+1} - \boldsymbol{\mu}_{O_{k+1}^i} \right) \quad (74)
$$

Rearranging, we get the final expression

$$
\boldsymbol{\mu}_{k+1} = \bar{\boldsymbol{\mu}}_{k+1} + K_{k+1} \left(\mathbf{z}_{k+1} - h(\bar{\boldsymbol{\mu}}_{k+1}) \right) + \Sigma_{k+1} \Sigma_{O_{k+1}^i}^{-1} \left(\boldsymbol{\mu}_{O_{k+1}^i} - \bar{\boldsymbol{\mu}}_{k+1} \right) \quad (75)
$$

APPENDIX B: Derivation of (12)

In this Appendix we derive the expression for Σ_{k+1} in terms of the Kalman gain K_{k+1} and the predicted covariance $\bar{\Sigma}_{k+1}$. For convenience we write down the matrix inversion lemma which states that for any invertible matrices B and C and any matrix D with appropriate dimensions, the following holds true

$$
\left(B + D C D^T \right)^{-1} = B^{-1} - B^{-1} D \left(C^{-1} + D^T B^{-1} D \right)^{-1} D^T B^{-1} \quad (76)
$$

We note here that the Kalman gain $K_{k+1} = \Sigma_{k+1} H_{k+1}^T Q_{k+1}^{-1}$ in (11) is a function of Σ_{k+1}. Thus we first need to derive an expression for K_{k+1} that does not depend Σ_{k+1}. To obtain such an expression, we follow the approach

for the standard EKF case presented in [31]. We begin by post-multiplying $\Sigma_{k+1}H_{k+1}^T Q_{k+1}^{-1}$ with an identity matrix $I = AA^{-1}$, where

$$A = \left(H_{k+1}\bar{\Sigma}_{k+1} \left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right)^{-1} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right) \tag{77}$$

To avoid clutter, let us further define $\tilde{\Sigma}_{k+l} = \left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right)^{-1}$. The expression for K_{k+1} can then be written as

$$
\begin{aligned}
K_{k+1} = {}& \Sigma_{k+1}H_{k+1}^T Q_{k+1}^{-1} \Big(H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + \\
& Q_{k+1}) \left(H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1} \\
= {}& \Sigma_{k+1} \Big(H_{k+1}^T Q_{k+1}^{-1} H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + \\
& H_{k+1}^T \Big) \left(H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1} \\
= {}& \Sigma_{k+1} \Big(H_{k+1}^T Q_{k+1}^{-1} H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + \\
& \left(\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} \right)^{-1} \bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T \Big) \\
& \left(H_{k+1}\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1} \tag{78}
\end{aligned}
$$

We will now compute the inverse of the term $\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i}$. This can be done as follows:

$$
\begin{aligned}
\left(\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} \right)^{-1} &= \left(\bar{\Sigma}_{k+1} \left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right)^{-1} \Sigma_{O_{k+1}^i} \right)^{-1} \\
&= \Sigma_{O_{k+1}^i}^{-1} \left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i} \right) \bar{\Sigma}_{k+1}^{-1} \\
&= \Sigma_{O_{k+1}^i}^{-1} \bar{\Sigma}_{k+1}\bar{\Sigma}_{k+1}^{-1} + \Sigma_{O_{k+1}^i}^{-1} \Sigma_{O_{k+1}^i}\bar{\Sigma}_{k+1}^{-1} = \Sigma_{O_{k+1}^i}^{-1} + \bar{\Sigma}_{k+1}^{-1} \tag{79}
\end{aligned}
$$

The expression in (78) simplifies to

$$
K_{k+1} = \Sigma_{k+1} \left(H_{k+1}^T Q_{k+1}^{-1} H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T + \right.
$$

$$
\left(\Sigma_{O_{k+1}^i}^{-1} + \bar{\Sigma}_{k+1}^{-1} \right) \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T \right)
$$

$$
\left(H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1}
$$

$$
= \Sigma_{k+1} \left(H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \Sigma_{O_{k+1}^i}^{-1} + \bar{\Sigma}_{k+1}^{-1} \right)
$$

$$
\bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T
$$

$$
\left(H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1}
$$

$$
= \Sigma_{k+1} \left(\Sigma_{k+1} \right)^{-1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T
$$

$$
\left(H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1} \quad (80)
$$

where we have used the fact that $\Sigma_{k+1}^{-1} = H_{k+1}^T Q_{k+1}^{-1} H_{k+1} + \Sigma_{O_{k+1}^i}^{-1} + \bar{\Sigma}_{k+1}^{-1}$. Thus we obtain

$$
K_{k+1} = \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T \left(H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T + Q_{k+1} \right)^{-1} \quad (81)
$$

Let us now define $\Xi_{k+1} = \Sigma_{O_{k+1}^i}^{-1} + \bar{\Sigma}_{k+1}^{-1}$. Applying the matrix inversion lemma to the right hand side of (10), we have

$$
\Sigma_{k+1} = \Xi_{k+1}^{-1} - \Xi_{k+1}^{-1} H_{k+1}^T \left(Q_{k+1} + H_{k+1} \Xi_{k+1}^{-1} H_{k+1}^T \right)^{-1} H_{k+1} \Xi_{k+1}^{-1} \quad (82)
$$

From (79), we have

$$
\Xi_{k+1}^{-1} = \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} \quad (83)
$$

We note here that the expression Ξ_{k+1}^{-1} appears four times in (82). Substituting for Ξ_{k+1}^{-1} using (83) in the second and third expression of Ξ_{k+1}^{-1} in (82), we get

$$
\Sigma_{k+1} = \Xi_{k+1}^{-1} - \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T
$$

$$
\left(Q_{k+1} + H_{k+1} \bar{\Sigma}_{k+1} \tilde{\Sigma}_{k+l} \Sigma_{O_{k+1}^i} H_{k+1}^T \right)^{-1} H_{k+1} \Xi_{k+1}^{-1} \quad (84)
$$

From (81) and (83), it is easily seen that the expression in (84) simplifies to

$$
\begin{aligned}
\Sigma_{k+1} = \Xi_{k+1}^{-1} - K_{k+1}H_{k+1}\Xi_{k+1}^{-1} &= (I - K_{k+1}H_{k+1})\,\Xi_{k+1}^{-1} \\
&= (I - K_{k+1}H_{k+1})\,\bar{\Sigma}_{k+1}\tilde{\Sigma}_{k+l}\Sigma_{O_{k+1}^i} \\
&= (I - K_{k+1}H_{k+1})\,\bar{\Sigma}_{k+1}\left(\bar{\Sigma}_{k+1} + \Sigma_{O_{k+1}^i}\right)^{-1}\Sigma_{O_{k+1}^i} \quad (85)
\end{aligned}
$$

This completes the derivation.

APPENDIX C: Derivation of (48)

From (43), we have

$$
\ln M(\theta) = d_0 + \sum_{k=1}^{\infty} d_k \frac{\theta^k}{k} \tag{86}
$$

For differentiable $M(\theta)$, we have

$$
\frac{d}{d\theta}\ln M(\theta) = \frac{1}{M(\theta)}\frac{d}{d\theta}M(\theta) = \sum_{k=1}^{\infty} c_k\theta^{k-1} \tag{87}
$$

where we have used the definition of $M(\theta)$ given in (39). Also note that by construction $M(\theta) > 0$. Re-arranging (87), we obtain

$$
M(\theta)\frac{d}{d\theta}\ln M(\theta) = \sum_{k=1}^{\infty} c_k\theta^{k-1} \tag{88}
$$

From (86), we have

$$
\frac{d}{d\theta}\ln M(\theta) = \sum_{k=1}^{\infty} d_k\theta^{k-1} \tag{89}
$$

From (87) and (89), we thus obtain

$$
\left(\sum_{k=0}^{\infty} c_k\theta^k\right)\left(\sum_{k=1}^{\infty} d_k\theta^{k-1}\right) = \sum_{k=1}^{\infty} c_k\theta^{k-1} \tag{90}
$$

Comparing the coefficient of θ^{k-1} on both sides of the equation, we get the required expression for c_k as

$$
c_k = \frac{1}{k}\sum_{j=0}^{k-1} d_{k-j}c_j \tag{91}
$$

61

References

References

[1] L. P. Kaelbling, M. L. Littman, A. R. Cassandra, Planning and acting in partially observable stochastic domains, Artificial Intelligence 101 (1-2) (1998) 99–134.

[2] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, J. P. How, Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns, Autonomous Robots 35 (1) (2013) 51–76.

[3] S. Prentice, N. Roy, The belief roadmap: Efficient planning in belief space by factoring the covariance, The International Journal of Robotics Research 28 (11-12) (2009) 1448–1465.

[4] J. Van Den Berg, S. Patil, R. Alterovitz, Motion planning under uncertainty using iterative local optimization in belief space, The International Journal of Robotics Research 31 (11) (2012) 1263–1278.

[5] L. P. Kaelbling, T. Lozano-Pérez, Integrated task and motion planning in belief space, The International Journal of Robotics Research 32 (9-10) (2013) 1194–1227.

[6] A.-A. Agha-Mohammadi, S. Chakravorty, N. M. Amato, FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements, The International Journal of Robotics Research 33 (2) (2014) 268–304.

[7] H. Kurniawati, V. Yadav, An Online POMDP Solver for Uncertainty Planning in Dynamic Environment, in: Robotics Research: The 16th International Symposium ISRR, 2016, pp. 611–629.

[8] S. Pathak, A. Thomas, V. Indelman, A unified framework for data association aware robust belief space planning and perception, The International Journal of Robotics Research 37 (2-3) (2018) 287–315. `doi:` `10.1177/0278364918759606.`

[9] A. Thomas, F. Mastrogiovanni, M. Baglietto, Task-Motion Planning for Navigation in Belief Space, in: The International Symposium on Robotics Research, 2019.

[10] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, D. Fox, Online replanning in belief space for partially observable task and motion problems, arXiv preprint arXiv:1911.04577.

[11] W. Liu, M. H. Ang, Incremental sampling-based algorithm for risk-aware planning under motion uncertainty, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 2051–2058.

[12] S. Patil, J. Van Den Berg, R. Alterovitz, Estimating probability of collision for safe motion planning under Gaussian motion and sensing uncertainty, in: IEEE International Conference on Robotics and Automation, 2012, pp. 3238–3244.

[13] N. L. Johnson, S. Kotz, N. Balakrishnan, Continuous univariate distributions. john wiley& sons, New York, NY.

[14] J. Hardy, M. Campbell, Contingency planning over probabilistic obstacle predictions for autonomous road vehicles, IEEE Transactions on Robotics 29 (4) (2013) 913–929.

[15] A. Bry, N. Roy, Rapidly-exploring random belief trees for motion planning under uncertainty, in: IEEE International Conference on Robotics and Automation, 2011, pp. 723–730.

[16] A. Lee, Y. Duan, S. Patil, J. Schulman, Z. McCarthy, J. Van Den Berg, K. Goldberg, P. Abbeel, Sigma hulls for gaussian belief space planning for imprecise articulated robots amid obstacles, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 5660–5667.

[17] N. E. Du Toit, J. W. Burdick, Probabilistic collision checking with chance constraints, IEEE Transactions on Robotics 27 (4) (2011) 809–815.

[18] C. Park, J. S. Park, D. Manocha, Fast and bounded probabilistic collision detection for high-DOF trajectory planning in dynamic environments, IEEE Transactions on Automation Science and Engineering 15 (3) (2018) 980–991.

[19] H. Zhu, J. Alonso-Mora, Chance-constrained collision avoidance for mavs in dynamic environments, IEEE Robotics and Automation Letters 4 (2) (2019) 776–783.

[20] A. Lambert, D. Gruyer, G. Saint Pierre, A fast Monte Carlo algorithm for collision probability estimation, in: 10th IEEE International Conference on Control, Automation, Robotics and Vision, 2008, pp. 406–411.

[21] L. Janson, E. Schmerling, M. Pavone, Monte Carlo motion planning for robot trajectory optimization under uncertainty, in: Robotics Research, Springer, 2018, pp. 343–361.

[22] L. Blackmore, M. Ono, B. C. Williams, Chance-constrained optimal path planning with obstacles, IEEE Transactions on Robotics 27 (6) (2011) 1080–1094.

[23] B. Axelrod, L. P. Kaelbling, T. Lozano-Pérez, Provably safe robot navigation with obstacle uncertainty, The International Journal of Robotics Research 37 (13-14) (2018) 1760–1774.

[24] L. Shimanuki, B. Axelrod, Hardness of 3D Motion Planning Under Obstacle Uncertainty, Workshop on Algorithmic Foundations of Robotics.

[25] O. Salzman, B. Hou, S. Srinivasa, Efficient motion planning for problems lacking optimal substructure, in: Twenty-Seventh International Conference on Automated Planning and Scheduling, 2017.

[26] A. M. Jasour, B. C. Williams, Risk contours map for risk bounded motion planning under perception uncertainties, Robotics: Science and Systems.

[27] A. Hakobyan, G. C. Kim, I. Yang, Risk-Aware Motion Planning and Control Using CVaR-Constrained Optimization, IEEE Robotics and Automation Letters 4 (4) (2019) 3924–3931.

[28] X. C. Ding, A. Pinto, A. Surana, Strategic planning under uncertainties via constrained markov decision processes, in: IEEE International Conference on Robotics and Automation, 2013, pp. 4568–4575.

[29] D. Sadigh, A. Kapoor, Safe control under uncertainty with probabilistic signal temporal logic, Robotics: Science and Systems.

[30] A. Thomas, F. Mastrogiovanni, M. Baglietto, An Integrated Localization, Motion Planning and Obstacle Avoidance Algorithm in Belief Space, Intelligent Service Robotics 14 (2) (2021) 235–250.

[31] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT press, 2005.

[32] H. Durrant-Whyte, T. C. Henderson, Multisensor data fusion, in: Springer handbook of robotics, Springer, 2016, pp. 867–896.

[33] S. B. Provost, A. Mathai, Quadratic forms in random variables: theory and applications, M. Dekker, 1992.

[34] S. Kotz, N. L. Johnson, D. Boyd, Series representations of distributions of quadratic forms in normal variables. I. Central case, The Annals of Mathematical Statistics 38 (3) (1967) 823–837.

[35] S. Kotz, N. L. Johnson, D. Boyd, Series representations of distributions of quadratic forms in normal variables ii. non-central case, The Annals of Mathematical Statistics 38 (3) (1967) 838–848.

[36] L. E. Kavraki, P. Svestka, J.-C. Latombe, M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Transactions on Robotics and Automation 12 (4) (1996) 566–580.

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

[38] R. Platt Jr, R. Tedrake, L. Kaelbling, T. Lozano-Perez, Belief space planning assuming maximum likelihood observations, in: Proceedings of Robotics: Science and Systems, Zaragoza, Spain, 2010.

[39] S. Patil, G. Kahn, M. Laskey, J. Schulman, K. Goldberg, P. Abbeel, Scaling up Gaussian Belief Space Planning Through Covariance-Free Trajectory Optimization and Automatic Differentiation, in: Eleventh International Workshop on the Algorithmic Foundations of Robotics, WAFR, Boğaziçi University, İstanbul, Turkey, Vol. 107 of Springer Tracts in Advanced Robotics, Springer, 2014, pp. 515–533.

[40] V. Indelman, L. Carlone, F. Dellaert, Planning in the Continuous Domain: a Generalized Belief Space Approach for Autonomous Navigation in Unknown Environments, International Journal of Robotics Research 34 (7) (2015) 849–882.

[41] Lozano-Perez, Spatial planning: A configuration space approach, IEEE Transactions on Computers C-32 (2) (1983) 108–120. `doi:10.1109/TC.1983.1676196`.

[42] S. Cameron, R. Culley, Determining the minimum translational distance between two convex polyhedra, in: Proceedings. 1986 IEEE International Conference on Robotics and Automation, Vol. 3, 1986, pp. 591–596.

[43] D. Schulz, W. Burgard, Probabilistic state estimation of dynamic objects with a moving mobile robot, Robotics and Autonomous Systems 34 (2-3) (2001) 107–115.

[44] E. Rimon, S. P. Boyd, Obstacle collision detection using best ellipsoid fit, Journal of Intelligent and Robotic Systems 18 (2) (1997) 105–126.

[45] M. Grötschel, L. Lovász, A. Schrijver, Geometric Algorithms and Combinatorial Optimization, Springer-Verlag, New York, 1988.