Planning non-conservative probabilistically safe paths for LiDAR-based navigation under bounded map errors

Victor Zhang, Kihiro Bando
{zhangvwk, bandok}@stanford.edu

I. Introduction

A. Motivation

In urban areas, LiDAR is typically the preferred sensor for autonomous navigation of unmanned aerial systems and unmanned ground vehicles. Online global localization is usually achieved by matching features of a point cloud with an available reference map [3], [6]. However, maps are only accurate to a certain extent, and previous LiDAR map matching literature [3] uses publicly available 3D maps which have been shown to contain non-negligible errors [8].

Supposing map errors are bounded with known bounds, one possible approach to take into account said errors in path planning is to predict the state uncertainty bounds in the presence of bounded map errors using work developed in [9], and then considering the full range of map errors for calculating the collision probability. While this approach guarantees safety, it does so over-conservatively. Consider the setup shown in Fig. 1: on the left side is a rectangle (shown in blue) whose right edge has a bias with known bounds (the boundary of which is shown in densely dashed lines). Suppose we were to evaluate the safety of the straight nominal path (shown loosely dashed) on the right side of the rectangle. If the actual online configuration of the rectangle is one where its right edge is biased to the left, then the agent would also deviate to the left because it localizes itself *relatively* to that right edge in order to deduce its global position in the reference map. Conversely, the agent would deviate to the right if the actual configuration of the right edge of the rectangle is biased to the right. Considering the full error range of said right edge would result in over-conservatively concluding that this nominal path is not safe since the middle figure shows that the state trajectory falls within said range, when 100 Monte-Carlo (MC) rollouts (shown in red) clearly show that this nominal path is in reality probabilistically safe regardless of the online configuration.

In this work and building on A. Shetty's [9] from Pr. Gao's Stanford NAVLab, we therefore present a framework for planning non-conservative probabilistically safe paths under the assumption of known map error bounds, specifically in the context of LiDAR-based navigation.

B. Related work

Existing work in literature incorporates other metrics into the path planning process, such as perception [4], feature richness heuristics [7] or state uncertainty [2]. However, to our knowledge this is the first effort in characterizing bounded map errors into the planning framework and explicitly integrating the computation of a tight upper bound on the collision probability in the context of LiDAR-based navigation.

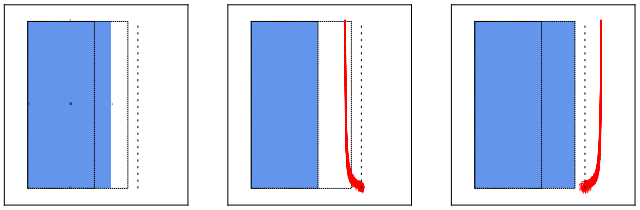

Fig. 1. Left: rectangle with bounded error on its right edge. Middle and right: MC rollouts of the dashed-line trajectory assuming the leftmost and rightmost edge bias, respectively. Considering the entire edge error range results in a loose and over-conservative upper bound on the collision probability.

II. Problem statement

A. System description

We assume a linear motion model $x_{k+1} = f(x_k, u_k) + w_k$ with $f(x_k, u_k) = A_k x_k + B_k u_k$, x the state, u the control input and $w \sim \mathcal{N}(0, Q_k)$ the process noise. We consider independent measurements $z^{(i)}$'s for which we assume a linear model $z_k^{(i)} = C_k^{(i)} x_k + e_k^{(i)} + \nu_k^{(i)}$ with $e^{(i)}$ an unknown bias with known bounds associated with the i^{th} measurement in the reference map, and $\nu^{(i)} \sim \mathcal{N}(0, R_k^{(i)})$ the measurement noise. The measurement model is the orthogonal distances to each obstacle surface seen by the agent, supposing that the LiDAR point cloud is processed with a line matching algorithm such as the one described in [3], such that the agent has as many measurements as the number of obstacle edges it currently perceives. For a time horizon T, we denote the nominal trajectory by $\check{\boldsymbol{x}}_{0:T} = (\check{x}_k)_{k \in [\![0,T]\!]}$ and $\check{\boldsymbol{u}}_{0:T-1} = (\check{u}_k)_{k \in [\![0,T-1]\!]}$, where $\check{x}_{k+1} = f(\check{x}_k, \check{u}_k)$. The online trajectory then tracks this nominal trajectory with a closed-loop feedback controller of the form $u_k = \check{u}_k - \check{K}_k(\hat{x}_k - \check{x}_k)$ with $\hat{x}$ the online state estimate given by a Kalman filter and $\check{K}$ the nominal gain, obtained via an infinite-horizon LQR controller.

B. Optimization problem

For a given map configuration $c \in \mathscr{C}$ where $\mathscr{C}$ is the bounded set of all map configurations, let $\mathcal{S}^c$ and $\mathcal{S}^c_{\text{free}} \subset$

$\mathcal{S}^c$ denote the state space and the free space, respectively, and $\mathcal{S}^c_{\text{obs}} = \mathcal{S}^c \backslash \mathcal{S}^c_{\text{free}}$ the obstacle space. Denoting $\mathcal{S}_{\text{free}} = \bigcap_{c \in \mathscr{C}} \mathcal{S}^c_{\text{free}}$, We seek to compute an obstacle-free trajectory from a start state $x_{\text{start}} \in \mathcal{S}_{\text{free}}$ to a goal region $\mathcal{S}_{\text{goal}} \subset \mathcal{S}_{\text{free}}$.

We approach this motion planning problem by seeking a trajectory of minimum cost (quantified by a cost metric h) such that for any map configuration c, the state lies within $\mathcal{S}^c_{\text{free}}$ with high probability, at all steps:

$$\begin{aligned} \underset{\check{\boldsymbol{u}}_{0:T-1}}{\text{minimize}} \quad & h(\check{\boldsymbol{x}}_{0:T}, \check{\boldsymbol{u}}_{0:T-1}) \\ \text{subject to} \quad & \forall\ k \in [\![0, T-1]\!], \quad \check{x}_{k+1} = f(\check{x}_k, \check{u}_k) \\ & \forall\ k \in [\![1, T-1]\!], \quad \check{x}_k \in \mathcal{S}_{\text{free}} \\ & \check{x}_0 = x_{\text{start}} \\ & \check{x}_T \in \mathcal{S}_{\text{goal}} \\ & \forall\ k \in [\![0, T]\!], \quad \max_{c \in \mathscr{C}} \mathbb{P}[x^c_k \in \mathcal{S}^c_{\text{obs}}] \leqslant \epsilon \end{aligned} \tag{1}$$

where we denote x^c_k as $x_k \mid c$, i.e. the state given the map configuration c. Note that $x_k \mid c \neq x_k$ in LiDAR-based navigation because the agent localizes itself using $\mathcal{S}^c_{\text{obs}}$. The probability constraint Eq. (1) has a $\max$ because the actual online configuration c_{true} is unknown beforehand.

III. Collision probability local approximation

Without loss of generality, suppose the problem is 2D, and the obstacle space is represented as an occupancy map, with the i^{th} obstacle approximated as a rectangle, each of its four edges being subject to an error with known bounds. Given a unique indexing for every edge of every rectangle of the map, and denoting $\mathscr{E}_j$ as the range of errors of the j^{th} edge in the map, and denoting $n_{\text{obs}} = |\mathcal{S}_{\text{obs}}|$:

$$\mathscr{C} = \bigtimes_{j=1}^{4 \cdot n_{\text{obs}}} \mathscr{E}_j \tag{2}$$

We then write the obstacle space given a configuration c as the disjoint union of all the obstacles: $\mathcal{S}^c_{\text{obs}} = \bigcup_{i=1}^{n_{\text{obs}}} \mathscr{Z}^c_i$. The term inside the $\max$ of the LHS of Eq. (1) is then:

$$\mathbb{P}[x^c_k \in \mathcal{S}^c_{\text{obs}}] = \mathbb{P}\left[x^c_k \in \bigcup_{i=1}^{n_{\text{obs}}} \mathscr{Z}^c_i\right] \approx \mathbb{P}\left[x^c_k \in \bigcup_{i \in \mathcal{I}^c_k} \mathscr{Z}^c_i\right]$$

where $\mathcal{I}^c_k$ is the set of obstacle indices whose one or more edges are seen by the agent at step k, assuming a configuration c. This approximation is valid in practice given that the range of LiDAR is around 200 m. The rightmost term is then simply $\sum_{i \in \mathcal{I}^c_k} \mathbb{P}[x^c_k \in \mathscr{Z}^c_i]$.

Next, notice that Eq. (1) is a $\max$ over the continuous domain $\mathscr{C}$ given by Eq. (2), but since the state evolves linearly with the configuration of a given line (see Fig. 1), then we can, without loss of correctness, restrict the $\max$ over just $\bigtimes_{j \in \mathcal{J}^{\text{tot}}_k} \partial\mathscr{E}_j$, where $\mathcal{J}^{\text{tot}}_k = \bigcup_{i=1}^{k} \mathcal{J}_k$ is the set of indices of obstacle edges seen up until step k. This would however still incur a $\max$ over potentially $2^{|\mathcal{J}^{\text{tot}}_k|}$ configurations, which quickly becomes intractable. We therefore continue to approximate the $\max$ operation by further restricting it over:

$$\mathscr{C}_{\text{proxy}} = \bigtimes_{j \in \mathcal{J}_k} \partial\mathscr{E}_j \quad \times \bigtimes_{j \in \mathcal{J}^{\text{tot}}_k \backslash \mathcal{J}_k} \mathscr{E}_j \tag{3}$$

i.e., the boundary of the edges' error range only for edges currently seen at step k, and the full error range for edges seen prior to but not at step k, thus bringing the maximization down to at most $2^{|\mathcal{J}_k|}$ evaluations. The rationale behind $\mathscr{C}_{\text{proxy}}$ (Eq. (3)) is that the agent's state is more affected by edges still seen. The entire analysis can be easily extended to 3D.

IV. Algorithm

Rapidly-Exploring Random Belief Trees [2] and Multiobjective Perception-Aware Planning (MPAP) [5] were considered as the planning under uncertainty framework to solve the above optimization problem. We chose the latter because of its ease of understanding and multi-query feature.

MPAP aims at finding a Pareto-optimal plan using a quasi-exhaustive search based on the cost and some perception heuristic describing the quality of the localization model. The output of the algorithm is quasi-optimal in the sense that it is the cost-wise best trajectory that satisfies a perception heuristic among all trajectories so far. It relies on a parallel treatment in order to cover a broad spectrum of possible trajectories. We modify the second objective to be a measure of a probabilistic safety instead of some heuristic assessing perception quality.

The algorithm proceeds in two stages. First the state space is sampled and a graph is constructed. Two states are connected if the infinite-horizon LQR controller can safely join them without colliding with any obstacles at a cost lower than a threshold r. Then, we run `Explore`(ϵ, α) defined in Alg. 1, wherein three sets of plans P, P_{open} and G are defined and the following operations are performed:

- for a plan `p` and a node n neighbor to its head `p.head`, `p.add(n)` forward propagates the state stochastic reachable sets of certain map configurations (cf. Section V-A);
- for a plan `p`, `p.max_prob` is the maximum collision probability over all possible map configurations computed via said reachable sets (cf. Sections IV,V-B);
- `RemoveDominated`(P, P_{open}) removes any plan `p` $\in P_{\text{open}}$ in P and P_{open} such that there exists a plan `p'` $\in P$ of lower cost and a reachable set that is strictly enclosed;
- `Prune`$(P_{\text{open}}, \alpha \in [0, 1[)$ is a pruning heuristic that removes the last α^{th} of P_{open} sorted according `cost` $\cdot \frac{\text{dist. to origin+dist. to goal}}{\text{dist. to origin}}$, to remove plans that circle around without making substantial progress towards the goal.

The environment, the zonotope representation and the algorithm were all implemented from scratch using the Python language. In particular, this algorithm requires a minimum level of parallel support in the implementation. Processing plans at each iteration is an embarrassingly parallel operation which was implemented using the `joblib` library.

Algorithm 1: `Explore`(ϵ, α)

$P_{\text{open}} \leftarrow \{p_0\}$
$P \leftarrow P_{\text{open}}$
$G \leftarrow P_{\text{open}}$
while $P_{\text{open}} \neq \varnothing \ \wedge \ \{p \in P \,|\, p.\texttt{head} \in \mathcal{S}_{\text{goal}}\} = \varnothing$ **do**
  foreach *plan* $p \in G$ **do**
    foreach *neighbor* n *of* $p.\texttt{head}$ **do**
      `p.add(n)`
      if $p.\texttt{max\_prob} \leqslant \epsilon$ **then**
        $P \leftarrow P \bigcup \{p\}$, $P_{\text{open}} \leftarrow P_{\text{open}} \bigcup \{p\}$
      end
    end
  end
  $(P, P_{\text{open}}) \leftarrow \texttt{RemoveDominated}(P, P_{\text{open}})$
  $P_{\text{open}} \leftarrow \texttt{Prune}(P_{\text{open}}, \alpha)$
  $P_{\text{open}} \leftarrow P_{\text{open}} \backslash G$ `// discard processed plans`
  $G \leftarrow P_{\text{open}}$
end
$P_{\text{candidates}} = \{p \in P \,|\, p.\texttt{head} \in \mathcal{S}_{\text{goal}}\}$
return $\text{argmin}_{p \in P_{\text{candidates}}} p.\texttt{cost}$

V. Stochastic Reachability Analysis

Probabilistic zonotopes come into play in two ways: as 1) a unified representation of the state reachable set regardless of whether the means are certain or uncertain; and 2) an efficient way to over-approximate collision probabilities.

A. State reachable set

As mentioned, assuming map errors are bounded with known bounds, previous work [9] propagates the state uncertainty accounting for bounded errors using the Kalman filtering framework, and provides the stochastic reachable set of the state, which we denote $\mathcal{X}_k$ at time step k. This probabilistic set represents the uncertainty of the state given the bounded errors on the measurements, and can be represented as a probabilistic zonotope. A probabilistic zonotope $\mathcal{Z}$ can be written as $\mathscr{Z}(c, G, \Sigma)$ where c is the center, G is a matrix where each column is a generator and Σ is the covariance of the Gaussian tails. Specifically, this representation is convenient for two reasons: 1) it captures the notion of an uncertain bounded mean m which belongs to the convex set $\{m \,|\, m = c + \sum_{i=1}^{e} \beta_i G[:, i];\ \forall i, \beta_i \in [-1, 1]\}$, and 2) supports two common operations on standard normal distributions, namely rescaling by a matrix T as $T \cdot \mathcal{Z} = \mathscr{Z}(Tc, TG, T\Sigma T^\top)$ and the addition of two independent variables which translates into the Minkowski sum $\mathcal{Z}_1 \oplus \mathcal{Z}_2 = \mathscr{Z}(c_1 + c_2, [G_1, G_2], \Sigma_1 + \Sigma_2)$, where columns of the matrices of generators have been concatenated.

Deterministic zonotopes including simple convex obstacles can be easily represented as probabilistic zonotopes with zero covariance. Conversely, a normal distribution with certain mean can be represented as a probabilistic zonotope with only one null generator.

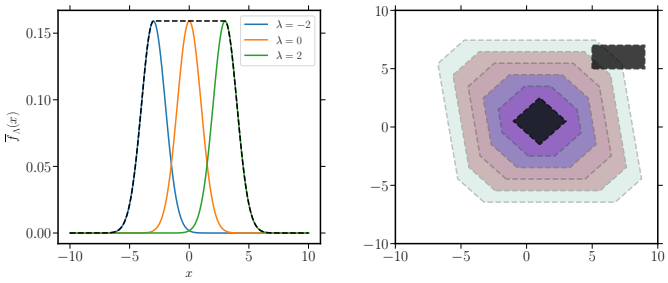

Fig. 2. Left: example of an enclosing probabilistic hull encompassing a normal distribution with uncertain mean. Right: confidence sets for a probabilistic zonotope with a randomly generated covariance matrix. Here, the collision probability with the obstacle is 0.0048.

In order to compute the reachable set $\mathcal{X}_k$ at step k, we unroll the recursive update and express x_k in terms of $\check{x}_0$, w_j and $\tilde{\nu}_j$ (among others) for $j \in [\![1, k]\!]$, where the latter takes into account the unknown bounded map and therefore – in the LiDAR context – measurement biases. Since w_j is a certain-mean Gaussian, it is represented equivalently by $\mathscr{Z}(0, 0, Q_j)$. The treatment for a measurement, on the other hand, depends on whether or not the full range of the uncertain mean is considered for a given configuration (cf. Section IV). If that is the case, $\tilde{\nu}_j^{(i)}$ will be represented as $\mathscr{Z}(0, \Delta e_j^{(i)}, R_j^{(i)})$ where $\Delta e_j^{(i)}$ is the half-width of the range of the possible means of the i^{th} measurement. This range is centered at zero. If only one extremum is considered, then it is represented as $\mathscr{Z}(\pm\Delta e_j^{(i)}, 0, R_j^{(i)})$. Then, we apply the Minkowski sum as well as the re-scaling operation on zonotopes described earlier to obtain $\mathcal{X}_k$. For the `RemoveDominated` operation in Alg. 1, the reachable sets compared are those corresponding to the full range of uncertain map errors. The coefficients involved in this unrolled equation can be computed recursively. Details of the exact calculation of $\mathcal{X}_k$ are omitted for lack of space.

B. Polytope approximation

When the uncertainty of the mean needs to be considered, the zonotope representation can be used to compute collision probabilities using enclosing probabilistic hulls. Given a set of probability density functions f_λ, $\lambda \in \Lambda$, the enclosing hull $\overline{f}_\Lambda = \sup_{\lambda \in \Lambda} f_\lambda(x)$ (cf. Fig. 2). In the case of uncertain bounded means as is the case for obstacle edges per Section IV, Λ is the set of possible means and f_λ is the normal probability distribution function centered at λ with covariance Σ. Therefore, the collision probability with an obstacle $\mathcal{Z}_{\text{obs}}$ can be computed as $\max\left[\int_{\mathcal{Z}_{\text{obs}}} \overline{f}_\Lambda, 1\right]$, which can be *efficiently over-approximated* via the polytope approximation described in [1], the details of which are omitted for lack of space. An example of a such approximation is shown in Fig. 2.

VI. Experiments

A. Model assumptions

For our experiments, we defined two separate 2D environments shown in Figs. 3 and 4, where certain edges of

the obstacles (shown in blue) contain errors for which the boundaries are shown in dashed lines. For example, the top and right edges of rectangle 0 in the first environment have errors while the other two are error-free. Moreover, we assume position-only states and velocity inputs with model matrices $A = I_2$, $B = dt \cdot I_2$, with $dt = 0.2s$. In practice, this could represent an omni-directional robot. The LQR planner for the connecting part of the graph building phase described Section III assumes $Q = I_2$ and $R = 0.1I_2$. The process noise is time varying by being scaled according to the ratio $\|u_k\| / \max_k \|u_k\|$ where the max is over the steps across the edge of the graph the agent is currently at, times 5% of the environment size. We set the observation noise to 0.1, $\epsilon = 0.1$ and $\alpha = 0.1$. The cost metric is the L_2 distance of the path.

B. Results

In both environments, the setup of the map errors is such that taking a distance-wise shortest path would be probabilistically unsafe.

Fig. 3 (top right) shows in red the path output by our algorithm, which is the shortest path that is probabilistically safe. The distance-wise shortest path is not: 100 MC rollouts (Fig. 3 top right) give a collision probability of 0.14 which is greater than $\alpha = 0.1$. In contrast, MC rollouts in the same configuration for the path output by the algorithm (going below the two obstacles in the middle), and in another example configuration are shown, and both report a collision probability of 0.1. To be rigorous in this verification though, we would actually have to perform rollouts for all the different possible map configurations to confirm that it is indeed safe.

Similarly, the path output by the algorithm in the second environment (shown in red on Fig. 4) is not the distance-wise shortest path but instead goes around the obstacles. Going to the left or right of obstacle 0 results in an unsafe path, as shown by MC rollouts (top right and middle left). To reiterate, these paths might be perfectly safe given some specific map configuration but because we take the maximum collision probability over all possible configurations since we do not know the actual online errors beforehand, said paths are considered *not* safe. In contrast, MC rollouts of the probabilistically safe path given three different map configurations are shown, and all report a collision probability of 0.

VII. Discussion and Future Work

We have explicitly investigated the effect of bounded map errors by integrating the computation of a tight upper bound on the collision probability in a planning framework, so as to plan probabilistically safe global paths non-conservatively. However, the main issue with the current proposed approach is the fact that a reachable set $\mathcal{X}_k$ at step k requires $\mathcal{O}(k)$ zonotope operations, and while the approximation described in Section IV helps with containing the number of map configurations to consider, said number still increases exponentially in the number of edges seen at the current step. Future work therefore includes decreasing the complexity of the algorithm and extending the experiments to higher state dimensions.

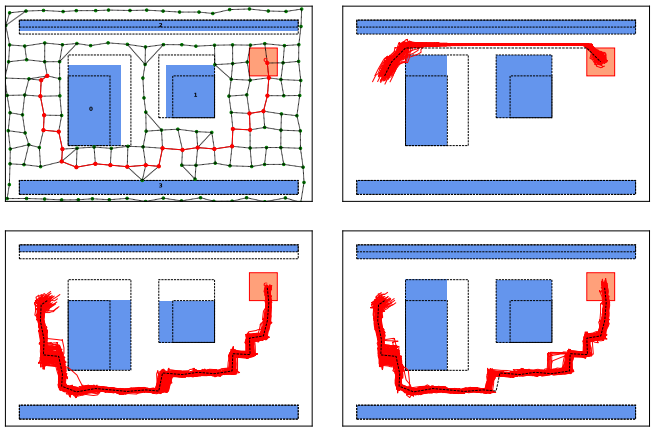

Fig. 3. Results on Environment 1. The algorithm outputs the path shown in red (top left). Top right: MC rollouts of the distance-wise shortest but unsafe path in a configuration where collision happens ($p_{\text{collision}} = 0.14$). Bottom left: MC rollouts of the probabilistically safe path in said configuration ($p_{\text{collision}} = 0.1$). Bottom right: MC rollouts of probabilistically safe path in another example configuration ($p_{\text{collision}} = 0.1$).

Fig. 4. Results on Environment 2. The algorithm outputs the path shown in red (top left). Top right and middle left: MC rollouts of two distance-wise shortest but unsafe paths in configurations where collision happens ($p_{\text{collision}} = 0.99$ and 0.49, resp.). Middle right and bottom left: MC rollouts of the probabilistically safe path in said configurations. Bottom right: MC rollouts of the probabilistically safe path in another example configuration.

REFERENCES

[1] M. Althoff, O. Stursberg, and M. Buss. Safety assessment for stochastic linear systems using enclosing hulls of probability density functions. In *2009 European Control Conference (ECC)*, pages 625–630, 2009.

[2] A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *2011 IEEE International Conference on Robotics and Automation*, pages 723–730, 2011.

[3] Derek Chen and Grace Xingxin Gao. Probabilistic graphical fusion of lidar, gps, and 3d building maps for urban uav navigation. *NAVIGATION*, 66(1):151–168, 2019.

[4] Gabriele Costante, Christian Forster, Jeffrey Delmerico, Paolo Valigi, and Davide Scaramuzza. Perception-aware Path Planning. *arXiv e-prints*, page arXiv:1605.04151, May 2016.

[5] Brian Ichter, Benoit Landry, Edward Schmerling, and Marco Pavone. Robust motion planning via perception-aware multiobjective search on gpus. *CoRR*, abs/1705.02408, 2017.

[6] Ehsan Javanmardi, Mahdi Javanmardi, Yanlei Gu, and Shunsuke Kamijo. Autonomous vehicle self-localization based on probabilistic planar surface map and multi-channel lidar in urban area. pages 1–8, 10 2017.

[7] S. A. Sadat, K. Chutskoff, D. Jungic, J. Wawerla, and R. Vaughan. Feature-rich path planning for robust navigation of mavs with mono-slam. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3870–3875, 2014.

[8] Sukhjit Sehra, Jaiteg Singh, and Hardeep Rai. Analysing openstreetmap data for topological errors. *International Journal of Spatial, Temporal and Multimedia Information Systems*, 1:87, 01 2016.

[9] Akshay Shetty and Grace Gao. Predicting state uncertainty for gnss-based uav path planning using stochastic reachability. pages 131–139, 10 2019.