2020 20th International Conference on Control, Automation and Systems (ICCAS 2020)
Oct. 13∼16, 2020; BEXCO, Busan, Korea

Provably Safe Real-Time Receding Horizon Trajectory Planning for Linear Time-Invariant Systems

Inkyu Jang[1], Dongjae Lee[1], and H. Jin Kim[1*]

[1]Department of Mechanical and Aerospace Engineering, Seoul National University,
Seoul, 08826, Korea ({leplusbon, ehdwo713, hjinkim}@snu.ac.kr) * Corresponding author

Abstract: Safe operation in spaces under uncertainty is crucial for robotic systems, especially for mobile robots. Assorted unknown quantities such as external force or estimation error can be considered disturbances. However, many robust trajectory planning algorithms that take effects of disturbances into account often accompany heavy computational load or are excessive conservatism. In this paper, we present a provably safe real-time receding horizon trajectory planning algorithm for linear time-invariant (LTI) systems. The proposed method ensures the same level of safety that other reachability-based robust planning algorithms provide, while not overestimating the reachable set. We verify the proposed framework through simulation with a six-degree-of-freedom system, in which the proposed method generates safe trajectories faster than 100 Hz.

Keywords: Robust Motion Planning, Trajectory Planning, Safe Flight Corridor, Reachability Analysis

1. INTRODUCTION

Safe operation of robots even in the presence of external disturbance and noise has emerged as a big concern in the field of robotics, as they are now serving for various missions in more diverse environments. Consideration of disturbances is particularly important for mobile robots in obstacle-dense environments, as unknown disturbances including external forces, sensor noise, and/or model error may cause collision or loss of stability.

Related to this issue, in the realm of motion planning, several studies have been conducted to guarantee robust collision avoidance against unknown external disturbance. In [1] and [2], Hamilton-Jacobi (HJ) reachability analysis was employed to guarantee safety for relatively simple dynamic systems. These HJ reachability analysis frameworks involve solving HJ partial differential equations (PDEs) and may take heavy computation. [3] circumvents this difficulty by finding an ellipsoid that encompasses the forward reachable set (FRS) and while using it as the safety bound. Several robust model predictive control (MPC) approaches are also used to secure safety against disturbances. Approaches deploying min-max MPC [4], which was first suggested in [5], can guarantee safety for all unknown but bounded disturbances in a fixed or receding horizon manner. Another approach, tube-based model predictive control (TMPC) [6, 7], aims to find the forward invariant sets which the system is guaranteed to stay within. These works are successful in guaranteeing safety during operation, but often require long computation time. Approximation techniques can facilitate real-time planning. However, as a trade-off, they might induce degradation of theoretic robustness or overly conservative safety constraints.

This work was supported by Institute of Information Communications Technology Planning Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2019-0-00399, Development of A.I. based recognition, judgement and control solution for autonomous vehicle corresponding to atypical driving environment)

In this work, we present a safety-guaranteed real-time receding horizon planning algorithm for linear time-invariant (LTI) systems in environments populated with stationary and known obstacle. The contribution of this paper is twofold. First, to reduce the computational burden of solving for the reachable sets, a collision checking method based on safe flight corridor (SFC) [8, 9] is proposed, instead of solving forward reachable set at every sampling time. This method guarantees the same level of safety and robustness as other reachability-based approaches. Second, a real-time robust receding horizon planning algorithm using the aforementioned collision prediction method is proposed. Through simulation, we show that the proposed planning algorithm runs in real-time, over 100 Hz in case of a six-degree-of-freedom system.

This paper is structured as follows. In section 2, we state the planning problem and theoretical methodologies to solve it. In subsection 2.1, a recursive formula to find the FRS is suggested; in subsection 2.2, a fast method based on linear programming (LP) is proposed to check for possible collision between an affine wall and the FRS; and subsection 2.3 discusses the overall receding horizon planning algorithm. A Monte Carlo simulation to verify the proposed method is presented in section 3.

2. PRELIMINARIES & PROBLEM DESCRIPTION

In the following subsections, we consider a discrete-time LTI system under disturbance in the following form:

$$x_{t+1} = Ax_t + Bu_t + Dw_t, \tag{1}$$

where $x_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$, and $w_t \in W \subseteq \mathbb{R}^{n_w}$ represent the state vector, input, and external disturbance, respectively, at time t, and A, B, D are matrices of appropriate sizes. The set W represents the disturbance bound and is assumed to be convex throughout this paper. However, W does not need to be full-dimensional in $\mathbb{R}^{n_w}$.

With a feasible reference trajectory x_t^{ref} and open-loop reference input u_t^{ref} such that $x_{t+1}^{\mathrm{ref}} = Ax_t^{\mathrm{ref}} + Bu_t^{\mathrm{ref}}$ and a reference-tracking linear controller $u_t = u_t^{\mathrm{ref}} - K\left(x_t - x_t^{\mathrm{ref}}\right)$, we obtain the following closed-loop error dynamics of the system:

$$e_{t+1} = (A - BK)\, e_t + Dw_t = A_c e_t + Dw_t, \tag{2}$$

where $e_t = x_t - x_t^{\mathrm{ref}}$ and K is the gain matrix. The closed-loop dynamics is assumed to be stable.

2.1 Calculation of the FRS

The FRS of the closed-loop system at time $t+k$ given information at time t, $X_{t+k|t}$, is defined to be the set of all reachable x_{t+k}'s to which a sequence of disturbance $w_\tau \in W$, $\tau \in \{t, t+1, \cdots, t+k-1\}$ can drive the system, starting from the initial state $x_t \in X_{t|t}$. Mathematically, it is defined as

$$X_{t+k|t} = \left\{ x_{t+k} \;\middle|\; \begin{array}{c} \forall\tau \in \{t, \cdots, t+k-1\}, \\ x_{\tau+1} = x_{\tau+1}^{\mathrm{ref}} + A_c(x_\tau - x_\tau^{\mathrm{ref}}) + Dw_\tau, \\ x_t \in X_{t|t},\ w_\tau \in W \end{array} \right\}. \tag{3}$$

Consider two reachable sets $X_{t+1|t_0}$ and $X_{t|t_0}$, where t_0 and $t \geq t_0$ are two arbitrary epochs. To obtain $X_{t+1|t_0}$ given $X_{t|t_0}$, the only input we need to consider is w_t, thus it is straightforward to find out that

$$X_{t+1|t_0} = x_{t+1}^{\mathrm{ref}} + A_c\left(X_{t|t_0} - x_t^{\mathrm{ref}}\right) + DW, \tag{4}$$

where we allowed some abuse of notation here, i.e., $PQ = \{Pq \mid q \in Q\}$ and $Q \pm v = \{q \pm v \mid q \in Q\}$ for a matrix P, a vector v, and a set of vectors Q. The plus sign between sets denotes the Minkowski sum. Using Eq. (4) recursively, we can calculate the reachale set using the following.

$$\begin{aligned} X_{t+k|t} &= x_{t+k}^{\mathrm{ref}} + A_c\left(X_{t+k-1|t} - x_{t+k-1}^{\mathrm{ref}}\right) + DW \\ &= x_{t+k}^{\mathrm{ref}} + A_c(A_c(X_{t+k-2|t} - x_{t+k-2}^{\mathrm{ref}}) \\ &\quad + DW) + DW \\ &= \cdots \\ &= x_{t+k}^{\mathrm{ref}} + A_c^k(X_{t|t} - x_t^{\mathrm{ref}}) \\ &\quad + A_c^{k-1}DW + A_c^{k-2}DW \\ &\quad + \cdots + A_cDW + DW \end{aligned} \tag{5}$$

Note that the multiplications of Eq. (5) are not distributive over Minkowski addition, i.e., for two matrices A_1 and A_2, $A_1W + A_2W \neq (A_1 + A_2)W$ in general. Thus, the terms involving DW cannot be merged.

Fig. 1 shows an exemplary FRS of the following two-degree-of-freedom system under disturbance:

$$x_{t+1} = \begin{bmatrix} 1.0 & 0.1 \\ -0.2 & 0.8 \end{bmatrix} x_t + w_t, \quad |w_t| \preceq 0.1, \tag{6}$$

where $a \preceq b$ if all components of a is less than or equal to b for vectors (of the same size) a and b. As it can be seen in Fig. 1, the number of vertices of the FRS grows very rapidly.

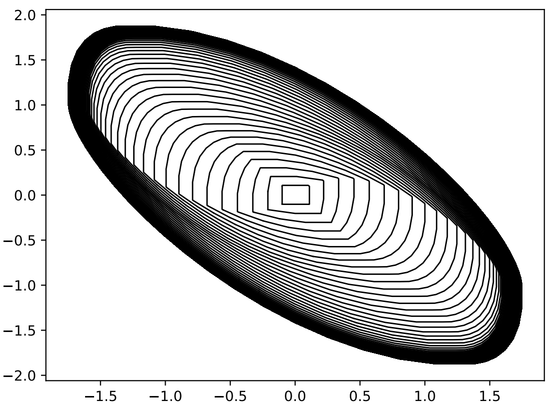

Fig. 1 The FRS of the given system Eq. (6) over 500 time steps starting from $X_{0|0} = \{0\}$. The FRS was calculated directly using Eq. (4) recursively and the quickhull algorithm. [10] At time step 250, it has more than 1000 vertices.

2.2 Safety Guarantee using FRS and SFC

Using Eq. (5) to calculate the FRS is, however, not an efficient way of detecting potential collisions in practice, since the formula requires many Minkowski additions which are in general computationally burdensome. In order to reduce this computational load, instead of calculating the actual reachable set, we plan a trajectory such that the FRS at each epoch stays within an SFC, which is a convex obstacle-free region. We consider polytopic SFC, which can be obtained in real-time using the algorithm proposed in [9], i.e.,

$$\mathrm{SFC}_t = \left\{x \mid c^\top x \leq d,\ \ \forall (c,d) \in \mathcal{W}(t)\right\}, \tag{7}$$

where $\mathcal{W}(t)$ selects appropriate halfspaces (c, d) that compose the SFC. To ensure safety for a time window $\tau \in \{t, t+1, \cdots, t+N_c\}$, the FRS $X_{\tau|t}$ should be contained within the SFC, i.e.,

$$\begin{aligned} d &\geq \sup_{x \in X_{t+k|t}}.\ c^\top x \\ &= c^\top x_{t+k}^{\mathrm{ref}} + \sup_{x \in X_{t|t}}.\ c^\top A_c^k(x - x_t^{\mathrm{ref}}) \\ &\quad + \sum_{j=0}^{k-1} \sup_{w \in W}.\ c^\top A_c^j Dw, \end{aligned} \tag{8}$$

for all $k \in \{0, 1, \cdots, N_c\}$ and $(c, d) \in \mathcal{W}(t+k)$. Eq. (8) consists of k LP problems on domain W and one LP on $X_{t|t}$. If W is convex, LPs can be readily solved in general. We assume that the disturbance bound W is given as a polytope. For fast computation, vertices of W and their connectivity are obtained and stored in the initialization process. Then, the LP problems of Eq. (8) can be solved by jumping between neighboring vertices of W. Matrices D, A_cD, $\cdots$, $A_c^{N_c-1}D$ can also be pre-evaluated to reduce running time.

Algorithm 1 Planner initialization (offline)

```
Input: Obstacle points($\mathcal{O}$)
Output: Initial trajectory ($p_{s_t}$), $\mathrm{SFC}_t$
1: $p_s \leftarrow \mathrm{LinearInterpolate}\left(\mathrm{GeneratePath}(\mathcal{O})\right)$
2: Find $s_t$ such that $\left\|p_{s_{t+1}} - p_{s_t}\right\|_{P_v} = v_{\mathrm{ref}}$.
3: for $i = 1$ to $N_s$ do
4:     $\mathrm{SFC}_t \leftarrow \mathrm{SFC}(p_{i-1}, p_i)\ \forall t,\ \text{s.t. } i-1 < s_t \le i$
5: end for
6: $\mathcal{W}(t) \leftarrow \mathrm{Walls}(\mathrm{SFC}_t)$
7: return $p_{s_t},\ \mathrm{SFC}_t$
```

For $c \in \mathbb{R}^{n_x}$ and $k \in \{0, 1, \cdots, N\}$, let $\delta_{c,k}$ and $\mu_{c,t,k}$ be

$$\delta_{c,k} = \begin{cases} \displaystyle\sum_{j=0}^{k-1} \sup_{w \in W}.\ c^\top A_c^j D w & (k \neq 0) \\ 0 & (k = 0), \end{cases} \tag{9}$$

and

$$\mu_{c,t,k} = \sup_{x \in X_{t|t}}.\ c^\top A_c^k (x - x_t^{\mathrm{ref}}). \tag{10}$$

Then, the condition in Eq. (8) can be rewritten as

$$c^\top x_{t+k}^{\mathrm{ref}} \le d - \delta_{c,k} - \mu_{c,t,k}. \tag{11}$$

The terms $\delta_{c,k}$ and $\mu_{c,t,k}$, which are usually greater than zero, can be interpreted as degrees of conservatism needed for x_{t+k}^{ref} to rule out any possibility of sticking out of the halfspace (c, d) due to external disturbance and estimation uncertainty, respectively.

2.3 Receding Horizon Trajectory Planning

In this subsection we delineate the overall structure of the receding horizon planning algorithm. The overall trajectory algorithm consists of two parts: global initial trajectory and SFC generation, and receding horizon reference trajectory update. The former step is executed once offline, while the latter one repeatedly runs online. The overall structure of the proposed algorithm runs as follows.

2.3.1 Global Initial Trajectory and SFC Generation

The initial trajectory and SFC are generated in the following manner. First, we obtain an initial path $p_s \in \mathbb{R}^{n_x}$, $s \in \{0, 1, \cdots, N_s\}$, which is an ordered set of via points. The line segment connecting two neighboring via points should be collision-free. For each of these line segments, a polytopic SFC containing the line segment is built. We denote the SFC containing p and q as $\mathrm{SFC}(p, q)$.

Next, the *time allocation* process is needed to supply the initial path with temporal information, thereby ensuring that the resulting reference trajectory is tractable. To assign time information, we start by linearly interpolating between neighboring via points, i.e., p_s now runs for continuous $s \in [0, N_s] \subseteq \mathbb{R}$. We will require the reference trajectory to run at a reference speed $v_{\mathrm{ref}} > 0$, where speed v_τ is calculated using

$$\begin{aligned} v_\tau &= \|x_{\tau+1} - x_\tau\|_{P_v} \\ &= \sqrt{(x_{\tau+1} - x_\tau)^\top P_v\,(x_{\tau+1} - x_\tau)}, \end{aligned} \tag{12}$$

where P_v is a constant positive semidefinite matrix that serves as weights for the state components in evaluating the speed. For that, the monotonically increasing sequence $s_\tau \in \mathbb{R}$ is defined for $\tau \ge 0$ to satisfy

$$\left\|p_{s_{\tau+1}}^{\mathrm{ref}} - p_{s_\tau}^{\mathrm{ref}}\right\|_{P_v} = v_{\mathrm{ref}}, \tag{13}$$

where p_{s_τ} is the global initial trajectory that will be used in the trajectory optimization process. The SFC corresponding to epoch τ is given based on the time allocation results, i.e., $\mathrm{SFC}_\tau = \mathrm{SFC}(p_s, p_{s+1})$ where $s \in \{0, 1, \cdots, N_s\}$ and $s < s_\tau \le s + 1$.

The initialization step is summarized in Algorithm 1. Lines 1 and 2 perform time allocation to the initial path, lines 3 through 6 generate and assign the SFC to each time step.

2.3.2 Receding Horizon Trajectory Optimization

In this step, we consider the running situation at time t, where a measurement or estimation of the state, $\hat{x}_t \in X_{t|t}$, is given. Using the global trajectory p_{s_τ} obtained in the previous step, the reference trajectory is found by solving the following optimization problem.

Problem 1 (Trajectory update)**.** *Find the reference input $u_{\tau-1}^{\mathrm{ref}}$ and trajectory x_τ^{ref} for $\tau \in \{t+1, \cdots, t+N\}$ that optimizes*

$$\begin{aligned} \min.\ & \frac{1}{2}\sum_{j=0}^{N} (x_{t+j}^{\mathrm{ref}} - p_{s_{t+j}})^\top P_j (x_{t+j}^{\mathrm{ref}} - p_{s_{t+j}}) \\ & + \frac{1}{2}\sum_{j=0}^{N} (u_{t+j}^{\mathrm{ref}})^\top Q_j\ u_{t+j}^{\mathrm{ref}} \\ \text{s.t. } & x_t^{\mathrm{ref}} = \hat{x}_t, \\ & u_t^{\mathrm{ref}} = u_t, \\ & (u_{t+k-1}^{\mathrm{ref}}, x_{t+k}^{\mathrm{ref}}) \in L, \\ & x_{t+k}^{\mathrm{ref}} = A x_{t+k-1}^{\mathrm{ref}} + B u_{t+k-1}^{\mathrm{ref}}, \\ & c^\top x_{t+k_c}^{\mathrm{ref}} \le d - \delta_{c,k_c} - \mu_{c,t,k_c}, \\ & \forall (c, d) \in \mathcal{W}(t + k_c), \\ & \forall k \in \{1, 2, \cdots, N\}, \forall k_c \in \{1, 2, \cdots, N_c\}. \end{aligned} \tag{14}$$

The matrices P_j and Q_j are symmetric positive semidefinite, and L is the set of practicable control input and state. If L, W, and $X_{t|t}$ are all convex sets, Problem 1 is a convex quadratic programming (QP) problem and can be solved using the following process. Possible collisions are checked for $\tau \in \{t, t+1, \cdots, t+N_c\}$, where $N_c \le N$ defines a time interval in which safety is guaranteed. Unless infeasibility occurs, safety is guaranteed throughout the mission if N_c is smaller than the time step of the receding horizon planning.

- First, solve the QP without the safety constraint $c^\top x^{\mathrm{ref}} \le d - \delta - \mu$.
- For $k \le N_c \le N$, check if x_{t+k}^{ref} is closer than a certain bound r to each wall of SFC_{t+k}. If so, evaluate δ and μ to check for possible collision.
- Add the wall constraint if collision is predicted and update the QP solution.

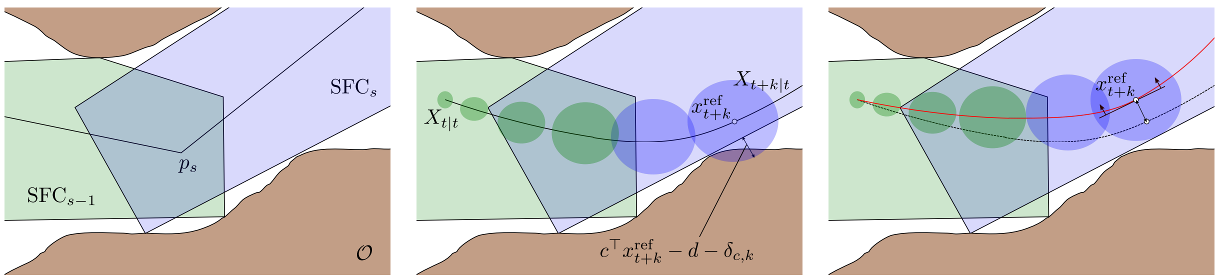

Fig. 2 An example for the trajectory update algorithm. Initially, via points and SFC are created. The obstacle regions are denoted as $\mathcal{O}$. (left) The optimization in Problem 1 is solved without the SFC constraints to obtain a smooth trajectory. The colors of the circles represent to which SFC they belong. The minimum distance between each SFC and the FRS is checked. The last blue FRS at epoch $t + k$ intrudes into unsafe region. (mid) With safety constraint at $t + k$, the trajectory is replanned. (right) This process repeats until all reachable sets satisfy the safety constraints. The full reachable sets are not evaluated during the planning algorithm but are depicted here for efficient explanation.

As described above, evaluation of the conservatism measure δ, which involves solving a number of LP problems and may consume some computation time, is suspended unless the system gets close to the wall. In most cases where the system is far enough from obstacles, the trajectory planning terminates without any update.

Fig. 2 illustrates the online trajectory update step.

3. SIMULATION RESULTS

To validate the aforementioned framework, we simulate using a jerk-controlled multirotor system in n-dimensional space, which is a $3n$-degree-of-freedom LTI system. The time-discretized system is governed by the following equations:

$$\begin{aligned} r_{t+1} &= r_t + v_t\delta t + \frac{1}{2}a_t\delta t^2 + \frac{1}{6}j_t\delta t^3 + w_t\delta t, \\ v_{t+1} &= v_t + a_t\delta t + \frac{1}{2}j_t\delta t^2, \\ a_{t+1} &= a_t + j_t\delta t, \end{aligned} \tag{15}$$

where $\delta t > 0$ is the length of the time step, r_t, v_t, a_t, w_t, $j_t \in \mathbb{R}^n$ and $x_t = [r_t^\top,\ v_t^\top,\ a_t^\top\ ]^\top$, $u_t = j_t$. The disturbance $w_t \in W$ represents unpredictable but bounded external wind, where the bound W is described as

$$W = \{w = [w_1, \cdots w_n]^\top \mid |w_i| \le w_{\max}\}. \tag{16}$$

The system is equipped with a linear controller that ensures convergence to the reference, i.e.,

$$j_t = j_t^{\mathrm{ref}} - K\left(x_t - x_t^{\mathrm{ref}}\right), \tag{17}$$

where

$$K = \begin{bmatrix} k_r\mathbf{1}_n & k_v\mathbf{1}_n & k_a\mathbf{1}_n \end{bmatrix} \tag{18}$$

is the gain matrix with positive gains (k_r, k_v, k_a), and $\mathbf{1}_n$ is the identity matrix of size $n \times n$. The control signal is assumed to be generated without any time delay. The weighting matrices P_k and Q_k of Problem 1 are considered constant and given as

$$\begin{aligned} P_k &= \mathrm{blkdiag}(w_r\mathbf{1}_n, w_v\mathbf{1}_n, w_a\mathbf{1}_n), \\ Q_k &= w_j\mathbf{1}_n. \end{aligned} \tag{19}$$

We assumed accurate estimation of the full state, i.e., $X_{t|t} = \{\hat{x}_t\} = \{x_t\}$ and thus $\mu = 0$. The obstacles are given as a set of stationary points in n-dimensional space. The actual values of the parameters used in the simulation are summarized in Table 1.

Table 1 Simulation parameters

Parameter	Value
Dimension (n)	2
Degrees of freedom	6
Length of the time window (N)	100
Length of the collision checking time window (N_c)	40
Number of via points (N_s)	5
Time step (δt)	0.01 s
Control gains (K)	$k_r = 400$, $k_v = 120$, $k_a = 10$
Control limits (L)	$\lvert a\rvert \preceq 10\ \mathrm{m/s^2}$
Cost weights	$w_r = 1000$, $w_v = 0$, $w_a = 0$, $w_j = 1$
Disturbance bound (W)	$\lvert w\rvert \preceq 0.7\ \mathrm{m/s}$
Reference velocity (v_{ref})	$0.9\ \mathrm{m/s}$

With the same setting, 100 Monte Carlo simulation experiments were conducted. The results are depicted in Fig. 3. To emulate adversarial disturbance, the wind is assumed to be heading in random directions with maximal speed. It is clearly shown that the simulated trajectories never intrude into the unsafe region. The algorithm was implemented using Python and NumPy of versions 3.7 and 1.18.4, and OSQP [11] version 0.6.0 was used to solve QP. It requires 0.3 seconds to initialize (SFC generation), and runs at a frequency of 100-200 Hz depending on whether replanning is needed, on a desktop computer equipped with 16 GB RAM and a hexa core AMD Ryzen$^{\mathrm{TM}}$ 5 1600 Processor whose base clock runs at 3.2 GHz.

4. CONCLUSION

This paper presents a real-time receding horizon trajectory planning algorithm for LTI systems, while guaranteeing safety against any possible sequence of bounded disturbance. This is done by constructing SFCs through-

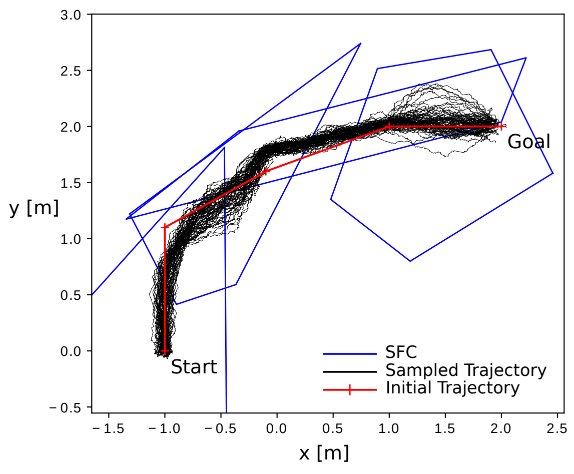

Fig. 3 Monte Carlo simulation. 100 trajectories are sampled and plotted. The environment is generated randomly, five via points are manually created. SFCs are denoted as blue regions.

out the environment, in which a line segment connecting neighboring via points lies. To avoid possible collisions, it is checked within the time window whether the FRS of the system is always located within the borders of the SFC, using linear programming at a fast computation speed. In a simulation environment where a six-degree-of-freedom LTI system runs in an obstacle-dense environment, the trajectory updates were computed in real-time at 100 Hz or faster.

One possible vulnerability of the proposed framework is that the reachability-based collision check is sometimes overly conservative and may result in infeasibilities. This issue is common in many approaches that make use of reachability analysis, because they try to find trajectories that remain collision-free under adversarial disturbance which always drives the system towards collision. This can be resolved by relaxing hard inequality constraints (of Problem 1) or actively adjusting control gains to shape the reachable set to fit within the safe flight corridor. We expect to address this problem in future works.

REFERENCES

[1] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. “A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games”. In: *IEEE Transactions on automatic control* 50.7 (2005), pp. 947–957.

[2] Sylvia L Herbert et al. “FaSTrack: A modular framework for fast and guaranteed safe motion planning”. In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE. 2017, pp. 1517–1522.

[3] Hoseong Seo et al. “Robust trajectory planning for a multirotor against disturbance based on hamilton-jacobi reachability analysis”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 3150–3157.

[4] Mario E Villanueva et al. “Robust MPC via min–max differential inequalities”. In: *Automatica* 77 (2017), pp. 311–321.

[5] Pierre OM Scokaert and David Q Mayne. “Min-max feedback model predictive control for constrained linear systems”. In: *IEEE Transactions on Automatic control* 43.8 (1998), pp. 1136–1142.

[6] Gowtham Garimella et al. “Robust Obstacle Avoidance using Tube NMPC.” In: *Robotics: Science and Systems*. 2018.

[7] Anirudha Majumdar and Russ Tedrake. “Funnel libraries for real-time robust feedback motion planning”. In: *The International Journal of Robotics Research* 36.8 (2017), pp. 947–982.

[8] Robin Deits and Russ Tedrake. “Computing large convex regions of obstacle-free space through semidefinite programming”. In: *Algorithmic foundations of robotics XI* (2015), pp. 109–124.

[9] S. Liu et al. “Planning Dynamically Feasible Trajectories for Quadrotors Using Safe Flight Corridors in 3-D Complex Environments”. In: *IEEE Robotics and Automation Letters* 2.3 (2017), pp. 1688–1695.

[10] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. “The quickhull algorithm for convex hulls”. In: *ACM Transactions on Mathematical Software (TOMS)* 22.4 (1996), pp. 469–483.

[11] B. Stellato et al. “OSQP: an operator splitting solver for quadratic programs”. In: *Mathematical Programming Computation* (2020).