

Autonomous Exploration using Ground Robots with Safety Guarantees

David Smith Sundarsingh, Jay Bhagiya, Jeel Chatrola, Pushpak Jagtap

Abstract—Autonomous exploration in an unknown environment is widely studied, and many exploration strategies exist. However, in most of the works, safety is not usually given top priority. The reason behind the violation of safety by most of the exploration algorithms in real-world applications is the ignorance of some or all of the following factors (i) the mathematical model of the robot, (ii) practical constraints on states (like constrained steering angle, speed, etc.) and inputs (like actuator saturation), and (iii) hardware constraints such as sampling time, sensor noise, modelling uncertainties, etc. In this work, we propose an autonomous exploration framework for the 2-D exploration problem that considers the factors above to provide safety guarantees for the robot and the environment. The effectiveness of this method is shown in a high-fidelity simulation of different robots with onboard sensors in different simulation environments and real-world implementation.

I. INTRODUCTION

Autonomous inspection, exploration, and mapping are some of the most critical use cases of autonomous robots. Autonomous exploration has many applications, including infrastructure and industrial inspection, search and rescue, security surveillance, and military reconnaissance [13]. Robots can now perform these tasks autonomously, dependably, and effectively thanks to recent developments in onboard sensing, processing, and several algorithmic contributions related to simultaneous localization and mapping (SLAM) and path planning. This paper will focus on the problem of safe autonomous exploration. We define exploration as mapping an unknown region while safely navigating through it without collisions. In general, the autonomous exploration process can be divided into three steps- First, the simultaneous localization and mapping (SLAM) algorithm generates a map of the given environment as perceived by the robot. Then, the frontier target detection module finds the next best target for further exploration. Finally, this target is fed to the motion planning algorithm, which navigates the robot to that target.

The reason behind the violation of safety in most autonomous exploration algorithms is due to ignorance of some or all of the following factors: (i) the mathematical model and dimensions of robots, (ii) practical constraints on states (like constrained steering angle) and inputs (like actuator saturation), and (iii) hardware constraints such as sampling time, sensor noise, and modelling uncertainties.

*This work was supported in part by the Google Research Grant, the SERB Start-up Research Grant, and the CSR Grant by Nokia Corporation.

†Authors contributed equally.

D. S. Sundarsingh, J. Bhagiya, J. Chatrola, and P. Jagtap are with Robert Bosch Centre for Cyber Physical Systems, IISc, Bangalore, India {jaybhagiya, jeelchatrola, pushpak}@iisc.ac.in

YouTube link: <https://www.youtube.com/watch?v=DTSIalccz6o&feature=youtu.be>

In this work, we propose an autonomous exploration framework for 2-D exploration that formally guarantees the robot's and environment's safety by considering all the abovementioned factors during frontier target detection and motion planning. We use the frontier-based exploration technique for the reasons stated in the next section. The proposed solution leverages the symbolic control approach [20], [15], which consists of creating a symbolic model of the continuous-state robot model and refining the controller synthesized for the symbolic model to a controller for the original robot, thereby providing guarantees of safe navigation to the target. We use symbolic controllers as they are not limited to systems of a particular class and can work with bounded state and input spaces. The contributions of the paper are highlighted below:

- A safe exploration target detection algorithm.
- An automated correct-by-construction motion planning module that considers all practical constraints on the system.
- Unlike conventional algorithms, the proposed framework needs very few tuning parameters that only depend on the system and not on the environment.
- A generalized approach that can be used on any robot given that the (approximate) mathematical model is known.

II. RELATED WORK

Simultaneous localization and mapping (SLAM) algorithms have been studied extensively. Many efficient SLAM algorithms have been designed. Some of the standard algorithms are eSLAM, GMapping, karto SLAM, and many more [25]. The proposed autonomous exploration framework works on a map generated by any SLAM algorithm.

Exploration can be done in two ways, decision-theoretic and frontier-based exploration. In decision-theoretic methods, [11], [7], information gain is mainly used to generate trajectories for exploration. However, these methods are computationally expensive and sometimes fall back to frontier exploration methods when they fail [11]. Frontiers, the region between free and unexplored areas, signify where the robot can explore more of the unknown region. Frontier-based exploration was first introduced in [23], where the size of the frontier and the distance between the robot and the frontier region were considered. Later, this technique was modified further by considering the robot's turning cost and maintaining the memory of all frontiers visited to avoid revisits [8]. Multiple Rapidly-exploring Randomized Trees (RRT) have also been used to detect frontiers within the explored region [22]. Recently, the concept of dynamic frontiers has

been introduced [4]. Due to their simplicity, intuitiveness, and ease of implementation, frontier exploration methods are preferred. Apart from this, various learning-based methods have been used for finding targets for frontier-based exploration, such as deep reinforcement learning [24], variational auto-encoders [17], and Q-networks [12]. However, they are difficult to generalise to various environments. For example, the model proposed in [24] is trained in office environments and so can only work effectively in office environments. This is problematic in an exploration scenario where the environment is unknown.

Motion planning involves computing a sequence of actions for moving a robot from its source to its destination while avoiding obstacles. Some techniques, such as the Dynamic Window Approach (DWA) planner [5], use cost functions to generate trajectories that avoid obstacles and reach the target as fast as possible while taking into account a dynamic window defined by the maximum acceleration of the system which places a constraint on the input to the system. Other techniques like Timed Elastic Band (TEB) planner [16] include modification of a global path based on path length and temporal aspects like velocity and acceleration with collision avoidance. For better control in difficult terrain, traversability has also been considered to modify a Hybrid A* algorithm [21]. These techniques require parameters that have to be tuned depending on the environment, which is not known.

There are a few approaches that consider safety as a factor, notably, authors in [18], use control barrier functions as a safety shield over an existing planner. However, the approach is limited to control-affine systems and unbounded control input. Also, [3] proposes an exploration framework based on Hamilton-Jacobi analysis however, it needs to convert the occupancy map into a regular, well-behaved function for the computation of value function which is challenging in a cluttered environment.

III. PROBLEM DEFINITION

Notations: For $a, b \in (\mathbb{R} \cup \{-\infty, \infty\})^n$, $a \leq b$ component-wise, the closed hyper-interval is denoted by $\llbracket a, b \rrbracket := \mathbb{R}^n \cap ([a_1, b_1] \times [a_2, b_2] \times \dots \times [a_n, b_n])$. We identify the relation $R \subseteq X \times \hat{X}$ with the map $R : X \rightarrow \hat{X}$ defined by $\hat{x} \in R(x)$ iff $(x, \hat{x}) \in R$ and it is a strict relation if $R(x) \neq \emptyset, \forall x \in X$. Given a relation $R \subseteq X \times \hat{X}$, $R^{-1} = \{(\hat{x}, x) \in \hat{X} \times X \mid (x, \hat{x}) \in R\}$. $Q \circ R$ denotes the composition of maps Q and R , $Q \circ R(x) = Q(R(x))$. Given a set $X \subset \mathbb{R}^n$, we define a uniform grid of X with quantization $\eta \in \mathbb{R}^+$ by $\eta\mathbb{Z}^n = \{c \in \mathbb{R}^n \mid \exists k_i \in \mathbb{Z}^n \forall i \in \{1, 2, \dots, n\} c_i = k_i \eta_i\}$ and for a given $c \in \eta\mathbb{Z}^n$, the cell is given by $c + \llbracket -\frac{\eta}{2}, \frac{\eta}{2} \rrbracket$.

Consider the mathematical model of a robot given by:

$$\dot{\xi}(t) \in f(\xi(t), v(t)) + \llbracket -\underline{w}, \bar{w} \rrbracket, \quad (1)$$

for any $t \in \mathbb{R}_0^+$, where $\xi(t) \in X \subset \mathbb{R}^n$ is a state vector, $v(t) \in U \subset \mathbb{R}^m$ is an input vector, $f : X \times U \rightarrow \mathbb{R}^n$ is a locally Lipschitz continuous map (the assumption is to ensure the existence and uniqueness of solution ξ [19]), and $\llbracket -\underline{w}, \bar{w} \rrbracket \subset \mathbb{R}^n$ represents various bounded uncertainties

in the system (such as modelling uncertainties, bounded external disturbance, etc.) with $\underline{w} = [\underline{w}_1, \dots, \underline{w}_n]^T \in (\mathbb{R}_0^+)^n$ and $\bar{w} = [\bar{w}_1, \dots, \bar{w}_n]^T \in (\mathbb{R}_0^+)^n$. Note that the state-space X and input space U are bounded (which is always the case in real-world applications).

We use the notation $\xi_{x,v}(t)$ to denote the solution at time t under the input signal v and starting from initial state $x = \xi_{x,v}(0)$. For a given system (1) and a set $X^0 \subset \mathbb{R}^n$, the reachable set of states that can be reached starting from set X^0 under input signal v at time τ is given by $\text{Reach}_\tau(X^0, v) := \bigcup_{x \in X^0} \xi_{x,v}(\tau)$. The exact computation of the reachable set is a difficult problem. However, there are various approaches available in the literature for computing an over-approximation of the reachable set represented as $\bar{\text{Reach}}_\tau(X_0, v)$; for example, [15, Definition VIII.3], [1], and [2].

Consider an unknown 2D environment $P \subset \mathbb{R}^2$, and state space of robot as $X = P \times O \subset \mathbb{R}^n$, where $P \subset \mathbb{R}^2$ denotes the position in the map and $O \subset \mathbb{R}^{n-2}$ denotes all other states such as orientation, velocities, etc. Using existing SLAM algorithms, we represent the environment using a uniform occupancy grid $\mu\mathbb{Z}^2$ with quantization size $\mu \in \mathbb{R}^+$. This grid is described by a matrix M and the indices corresponding to $c \in \mu\mathbb{Z}^2$ can be obtained using the index function $I_d : \mu\mathbb{Z}^2 \rightarrow \mathbb{N}^2$. Now we define free region P_f , obstacle sets P_o and unexplored region P_u regions,

$$\begin{aligned} P_o &= \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } M(I_d(c))=1} c + \llbracket -\frac{\mu}{2}, \frac{\mu}{2} \rrbracket, \\ P_u &= \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } M(I_d(c))=-1} c + \llbracket -\frac{\mu}{2}, \frac{\mu}{2} \rrbracket, \\ P_f &= \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } M(I_d(c))=0} c + \llbracket -\frac{\mu}{2}, \frac{\mu}{2} \rrbracket. \end{aligned}$$

We denote state spaces corresponding to P_f , P_o , and P_u as $X_f = P_f \times O$, $X_o = P_o \times O$ and $X_u = P_u \times O$, respectively.

Problem 3.1: Given robot model as in (1) along with bounds of state and input spaces X and U , uncertainties in the systems, and a closed environment $P \subset \mathbb{R}^2$; design an automated exploration framework to incrementally explore the unknown region P_u while preserving guarantees on the safety of the robot and the environment until no safe targets are identified.

Safe regions have no obstacles and are observable or registered on the map. Note that the assumption on closedness of the environment is needed to ensure the termination of the proposed algorithm. However, the proposed framework can generally be used to explore non-closed environments. The following section discusses the proposed framework to solve Problem 3.1.

IV. PROPOSED EXPLORATION FRAMEWORK

This paper proposes an effective and practical framework for autonomous exploration, which includes two modules: (i) a Safe Frontier Target Detector (SFTD) and (ii) a Symbolic Motion Planner (SMP). In the safe frontier detection module,

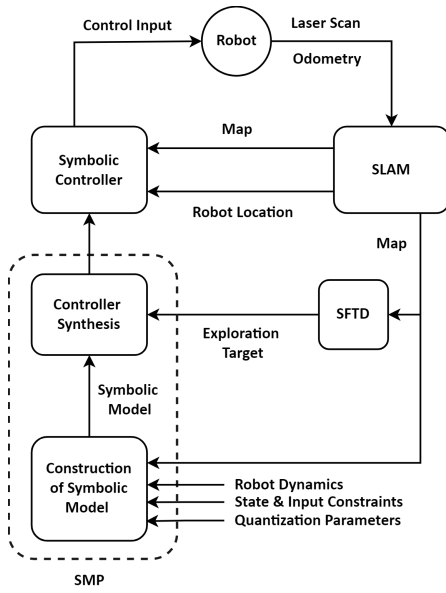


Fig. 1: Exploration framework schematic

given map P by SLAM algorithms and current robot location, we identify safe and feasible targets in the explored map and then rank these targets based on the volume of the unexplored region. Based on the mathematical model of the system and constraints, a proposed symbolic motion planner synthesizes a controller that navigates the robot to the identified exploration target with safety guarantees. The schematic of the proposed exploration strategy framework is shown in Figure 1.

A. Safe Frontier Target Detection (SFTD)

In this Subsection, we propose a novel, safe target detection algorithm based on [23] to introduce safety by modifying it to find a region near the frontier that is free of obstacles and is fully known. Due to the possibility that one of the destinations may not have a safe path leading to it, we generate multiple targets ranked according to how much of the surrounding area is unexplored.

Given a map P , the algorithm aims to get a set of k possible target regions $\mathcal{T} = \{T_1, \dots, T_k\}$, where $T_i = F_i + [-T_{win}, T_{win}] \subset P_f$ with $F_i \in P_f$ and $T_{win} \in \mathbb{N}$ is the target window size. We choose $T_{win} > r_{robot}$, where r_{robot} is the minimum radius of the circle that encloses the complete robot under consideration. We denote a set of frontier windows corresponding to each target by $\mathcal{T}_F = \{\mathcal{F}_{T_1}, \dots, \mathcal{F}_{T_k}\}$. The frontier window is given by $\mathcal{F}_{T_i} = F_i + [-F_{win}, F_{win}] \subset P_f \cup P_u$ for every $T_i \subset P_f$, where $F_{win} > T_{win}$. This is to accommodate the over-approximation of obstacles in the motion planning phase, to ensure collision avoidance with unforeseen obstacles in unexplored regions, and to ensure the presence of frontier. For illustration, refer to Figure 2, where the brown box represents the target window and the green box represents the frontier window, and the yellow region shows the over-approximation of the obstacle and unexplored regions.

First, the map is split into smaller submaps. The submaps

containing frontiers are then ranked based on the size of unexplored regions. The submap with the least unexplored region is given the highest rank as this will ensure the completion of exploration in one submap before moving into another. The frontier targets T_i are then computed in each submap and are the regions near the frontier that have no obstacles. The target with the least number of unexplored cells in its corresponding frontier window is chosen in each submap to ensure that most cells near the frontier target are known. The targets are then stored in $\mathcal{T}$ in their ranked order and given to the SMP for the generation of a strategy that navigates the robot to $T_i \in \mathcal{T}$.

B. Symbolic Motion Planner (SMP)

Given an occupancy grid map $P = P_f \cup P_o \cup P_u$ generated by a SLAM algorithm, and ranked target set $\mathcal{T}$ from the SFTD, this subsection proposes a symbolic motion planner that guarantees the safety of the robot and environment while navigating to the target. For the construction of a symbolic motion planner, we consider an obstacle space $X_{obs} = (P_o \cup P_u + [-r_{robot}, r_{robot}]) \times O$. Note that since the proposed algorithm considers the robot as a point mass, we inflated the actual obstacle set $P_o \cup P_u$ by the size of the robot to ensure collision avoidance with obstacles. Next, we list some necessary preliminaries required for the main result that provides a formal guarantee on the proposed symbolic motion planner.

1) *Transition Systems and Equivalence Relation:* We present the notion of *transition system* [20] which will be used as a unified framework to represent the mathematical model (1) and the corresponding symbolic model.

Definition 4.1 (Transition system): A transition system is a tuple $S = (X, X^0, U, \rightarrow)$, where X is a set of states, $X^0 \subseteq X$ is a set of initial states, U is a set of inputs, $\rightarrow \subseteq X \times U \times X$ is a transition relation.

We denote by $x \xrightarrow{u} x'$ an alternative representation for transition $(x, u, x') \in \rightarrow$, where state x' is called a u -successor (or simply successor) of state x , for some input $u \in U$. We denote by $Post_u(x)$ the set of all u -successors of state x , and by $U(x)$ the set of all admissible inputs $u \in U$ for which $Post_u(x)$ is non-empty. Now, we provide the notion of feedback refinement relation between two transition systems, introduced in [15], which is later used to construct a symbolic model and formally verified motion planning algorithm.

Definition 4.2 (Feedback refinement relation): Consider two transition systems $S = (X, X^0, U, \rightarrow)$ and $\hat{S} = (\hat{X}, \hat{X}^0, \hat{U}, \hat{\rightarrow})$ with $\hat{U} \subseteq U$. A strict relation $Q \subseteq X \times \hat{X}$ is a feedback refinement relation from S to $\hat{S}$, denoted by $S \preceq_Q \hat{S}$, if following conditions hold for every pair $(x, \hat{x}) \in Q$:

- (i) $\hat{U}(\hat{x}) \subseteq U(x)$,
- (ii) $u \in \hat{U}(\hat{x}) \Rightarrow Q(Post_u(x)) \subseteq Post_u(\hat{x})$,

Intuitively, the above relation says that all admissible inputs of $\hat{S}$ can be used in transition system S such that all transitions in S are associated with corresponding transitions in $\hat{S}$.

As a result, one can easily refine the controller synthesized for the symbolic model $\hat{S}$ using feedback refinement relation Q to make it compatible with S . Further details about feedback refinement relation can be found in [15].

2) *Construction of Symbolic Model:* Before discussing the construction of symbolic models for the model (1), we assume that the value of control inputs of the robot remains constant for a sampling time $\tau \in \mathbb{R}^+$ and the sampled robot model is given by the tuple $S_h = (X_h, X_h^0, U_h, \xrightarrow{h})$ as defined in Definition 4.1. Note that we abuse the notation above by identifying u with the constant input curve with domain $[0, \tau]$ and value u .

To design a navigation controller for sampled robot model, we first construct a symbolic model of S_h that satisfies formal behavioural inclusions in terms of feedback refinement relation. Given a quantization parameter $\eta \in (\mathbb{R}^+)^n$, the symbolic model of S_h given by the tuple $\hat{S}_h = (\hat{X}_h, \hat{X}_h^0, \hat{U}_h, \xrightarrow{h})$, where

- $\hat{X}_h$ is a cover of X_h and elements of the cover $\hat{X}_h$ are nonempty, closed hyper-intervals referred to as cells. For the computation of the abstraction, we consider subset $\hat{X}_h \subseteq \hat{X}_h$ of congruent hyper-rectangles aligned on a uniform grid parameterized with quantization parameter $\eta \in (\mathbb{R}^+)^n$, i.e., $\hat{x} \in \hat{X}_h$ implies that there exists $c \in \eta\mathbb{Z}^n$ with $\hat{x} = c + [-\frac{\eta}{2}, \frac{\eta}{2}]$. The remaining cells $\hat{X}_h \setminus \hat{X}_h$ are considered as "overflow" symbols [14],
- $\hat{X}_h^0 \subseteq \hat{X}_h$,
- $\hat{U}_h$ is a finite subset of U_h ,
- for $\hat{x} \in \hat{X}_h \setminus X_{obs}$ and $\hat{u} \in \hat{U}_h$, define $A := \{\hat{x}' \in \hat{X}_h \mid (\hat{x}' \cap \text{Reach}_\tau(\hat{x}, \hat{u})) \neq \emptyset \text{ and } \text{Reach}_\tau(\hat{x}, \hat{u}) \cap X_{obs} = \emptyset\}$. If $A \subseteq \hat{X}_h$, then $\text{Post}_u(\hat{x}) = A$, and otherwise $\text{Post}_u(\hat{x}) = \emptyset$. Moreover, $\text{Post}_u(\hat{x}) = \emptyset$ for all $\hat{x} \in (\hat{X} \setminus \hat{X}) \cup X_{obs}$.

Theorem 4.3: If $\hat{S}_h$ is a symbolic model of S_h (a sampled robot model) with sampling time $\tau \in \mathbb{R}^+$, state quantization parameter $\eta \in (\mathbb{R}^+)^n$, input quantization parameter $\eta_u \in (\mathbb{R}^+)^m$, and an obstacle set X_{obs} , then $S \preceq_Q \hat{S}$.

Proof: This proof follows from [15, Theorem VIII.4]. Consider condition (i) of the Definition 4.2. The input set U for each robot is discretized by a parameter η_u . At every state x , the input set $U_h(x)$ has all the admissible inputs at that state. In the abstraction of the system, the set $\hat{U}_h(\hat{x})$ is empty if $\hat{x} \in \hat{X}_h \setminus \hat{X}_h$ and any $\hat{u} \in \hat{U}_h(\hat{x})$ such that $\text{Reach}_\tau(\hat{x}, \hat{u}) \cap X_{obs} \neq \emptyset$ is removed from $\hat{U}_h(\hat{x})$. Thus $\hat{U}_h(\hat{x}) \subseteq U_h(x)$.

For condition (ii) in Definition 4.2, consider an input $\hat{u} \in \hat{U}(\hat{x})$ such that $Q(\text{Post}_u(x)) \not\subseteq \text{Post}_u(\hat{x})$ where $Q(\text{Post}_u(x))$ is the set of all $\hat{x} \in \hat{X}_h$ which correspond to all $x' \in \text{Post}_u(x)$ according to the quantifier Q . Since by definition $\text{Post}_u(\hat{x}) = \text{Reach}_\tau(\hat{x}, \hat{u}) \cap \hat{X}_h$ and this is the over-approximated set of the solutions as defined in Section III, the only condition when $Q(\text{Post}_u(x)) \not\subseteq \text{Post}_u(\hat{x})$ is if $\text{Reach}_\tau(\hat{x}, \hat{u}) \cap X_{obs} \neq \emptyset$ but if so, $\hat{u} \notin \hat{U}_h(\hat{x})$ by definition on the abstraction. This leads to a contradiction. ■

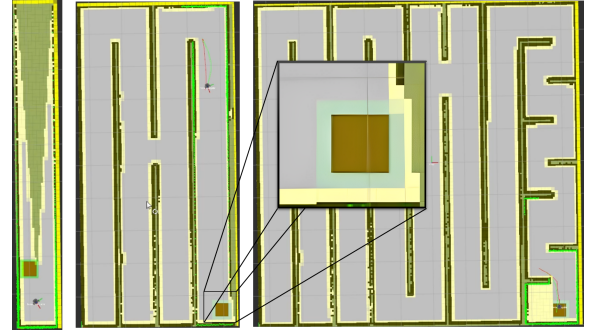


Fig. 2: Stages of exploration with target structure

3) *Controller Synthesis and Refinement:* For an obtained symbolic model $\hat{S}$, we synthesize a controller that will navigate the robot to the best-ranked target T_i while avoiding obstacle set X_{obs} . The controller $\hat{C} : \hat{X} \times \hat{U}$ is synthesized using a variant of Dijkstra's shortest path algorithm for hyper-graphs taken from [6]. One can easily refine this controller for S using the following theorem.

Theorem 4.4: If $S_h \preceq_Q \hat{S}_h$ and $\hat{C}_h$ is controller for $\hat{S}_h$ for a given target T_i , then the refined controller $C_h := \hat{C}_h \circ Q$ solves the reachability of T_i for sampled mathematical model S_h .

Proof: The proof is similar to [15, Theorem VI.3]. ■

Intuitively, the refined controller C for S can naturally be obtained from the abstract controller $\hat{C}$ by using the feedback refinement relation Q as a *quantizer* to map x to $\hat{x} \in Q(x)$. The refined controller is given as $C := \hat{C} \circ Q$. The robot uses the controller as a lookup table during navigation. Once the state x of the robot in the concrete system is known, the corresponding control input at that state, which will eventually move the robot to the target, is $u \in \hat{C}(Q(x))$. This input is applied to the system, and the process continues until the robot reaches the target T_i .

Remark 4.5: Note that if we fail to find the controller for reaching the best-ranked target T_i . We try to synthesize the controller for the next best target. Moreover, if we cannot find a controller for any of the targets, we conclude that it is not safe to move towards any target, and exploration will be terminated.

Remark 4.6: The above framework allows for synthesizing robust navigation controllers concerning various disturbances, including modelling uncertainties, input disturbances, and measurement errors. We kindly refer readers to [15, Section VI.B] for a detailed discussion.

Remark 4.7: The obtained controller C solves the safe reachability problem for the sampled system. However, one can ensure the safety guarantee of the inter-sampling trajectory by expanding the obstacle set X_{obs} by a magnitude computed using the global Lipschitz continuity property of map f . We refer the reader to [9] for a more detailed discussion.

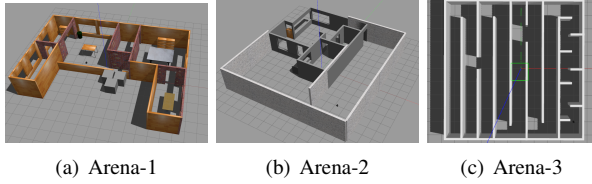


Fig. 3: Different environments considered for comparison

V. EXPERIMENTAL RESULTS

A. Simulation and Experimental Setup

To demonstrate the applicability of the proposed framework, we implement it on different robots:

(i) Differential drive robot (R_1):

$$\begin{aligned}\dot{\xi}_1(t) &= v_1(t) \cos(\xi_3(t)) + \llbracket -0.01, 0.01 \rrbracket, \\ \dot{\xi}_2(t) &= v_1(t) \sin(\xi_3(t)) + \llbracket -0.01, 0.01 \rrbracket, \\ \dot{\xi}_3(t) &= v_2(t) + \llbracket -0.01, 0.01 \rrbracket,\end{aligned}$$

where $(\xi_1(t), \xi_2(t))$ is the position and $\xi_3(t)$ is the orientation of the robot. The inputs are linear velocity $v_1(t) \in [-0.22, 0.22]$ and angular velocity $v_2(t) \in [-0.11, 0.11]$, respectively.

(ii) Single-track car model (R_2):

$$\begin{aligned}\dot{\xi}_1(t) &= v_1(t) \cos(\alpha(t) + \xi_3(t)) \cos(\alpha(t))^{-1} + \llbracket -0.01, 0.01 \rrbracket, \\ \dot{\xi}_2(t) &= v_1(t) \sin(\alpha(t) + \xi_3(t)) \cos(\alpha(t))^{-1} + \llbracket -0.01, 0.01 \rrbracket, \\ \dot{\xi}_3(t) &= v_1(t) \tan(v_2(t)) + \llbracket -0.01, 0.01 \rrbracket,\end{aligned}$$

where $\alpha(t) = \arctan(\tan(v_2(t))/2)$, $(\xi_1(t), \xi_2(t))$ is the position and $\xi_3(t)$ is the orientation of the vehicle in the 2-dimensional plane. The control inputs $v_1(t) \in [-0.6, 0.6]$ and $v_2(t) \in [-0.6, 0.6]$ are the rear wheel velocity and the steering angle, respectively.

We use a Turtlebot3 (R_1) and a car-like robot (R_2) in the simulation environment. In a real-world setting, we use Turtlebot3. The considered robots are equipped with a 360° laser sensor RPLIDAR A2 (range $\sim 8m$) and basic onboard odometry. We used a computer with AMD Ryzen 9 5950x and 128 GB RAM to perform simulations. The experimental system runs on Ubuntu 20.04 and the Robot Operating System (ROS) Noetic distribution. Karto-SLAM [10] was used to construct the 2D grid map P . For comparison purposes, we use the same robot parameters over different techniques and arenas. The simulation arenas are created using the Gazebo simulator. To investigate the influence of the map structure on the efficiency of autonomous exploration algorithms, we consider three different maps shown in Figure 3. For constructing symbolic model for R_1 and R_2 as discussed in Section IV-B, we use state quantization parameters $\eta_{R_1} = [0.1, 0.1, 0.2]$, $\eta_{R_2} = [0.05, 0.05, 0.2]$ and input quantization parameters $\eta_{R_1}^v = [0.02, 0.01]$, $\eta_{R_2}^v = [0.2, 0.2]$, respectively. The implementation of the proposed symbolic planner is computationally expensive. Thus, we ignore the planner computation time for quantifiable comparison with other real-time planners.

B. Experimental Results

The proposed approach is compared with existing and open-source autonomous exploration techniques such as frontier-based exploration [23] and multiple RRTs for exploration [22]. To compare the repeatability and generality of the algorithms, we conducted a total of 90 sets of exploration runs on three maps with five starting points in each.

Figure 4 compares the exploration techniques in terms of exploration time, the number of collisions and the final percentage of area explored in three different simulation environments. From Figure 4(a), one can readily observe that the proposed strategy is slower than the frontier-based strategy but faster than multiple RRTs exploration. This is due to the greedy approach used in the frontier-based technique, which targets the largest frontier on a map. In contrast, multiple RRTs have a random tree that finds the frontier points based on the information gain. Thus, the targets change more frequently, resulting in revisiting already explored regions. In the proposed approach, we find a safe target region instead of a point target to increase the safety of the robot and the environment. To ensure overall safety, the proposed approach considers unexplored regions as pseudo-obstacles due to the uncertainty of information in that region.

Figure 4(b) shows that the existing planners, such as DWA and TEB, work well in one environment but perform poorly in others due to numerous parameters that need to be tuned based on the environment complexity. Conversely, the proposed approach depends on three (viz., state discretization, input discretization, and sampling time) tuning parameters that are independent of the environment. We also compared the percentage of area explored (see Figure 4(c)). The results show that other algorithms perform best in relatively big and less cluttered arenas. But when it comes to narrow environments like Arena-3, for which the exploration progress is shown in Figure 2, the problem occurs because the planners fail to navigate the robot through the environment safely. From 4(b), one can easily observe that the total exploration percentage by the other two algorithms compared to our approach decreases significantly due to many collisions. Since the exploration approaches that consider safety [3], [18] have unbounded input spaces and are constrained by approximations on the environment, we cannot efficiently compare those results in the considered experiments.

The implementation of the proposed framework can be found at https://github.com/FocasLab/autonomous_exploration and video of the experiments can be found at <https://youtu.be/DTSIalccz6o>.

VI. CONCLUSION AND FUTURE WORK

In this work, we proposed a general autonomous exploration framework for 2-D exploration using ground robots that provides safety guarantees for the robot and the environment. This is achieved by considering many practical factors, such as the mathematical model of the robot, state/input constraints, modelling uncertainties, and sensor noise while designing a safe frontier target and motion planner. The

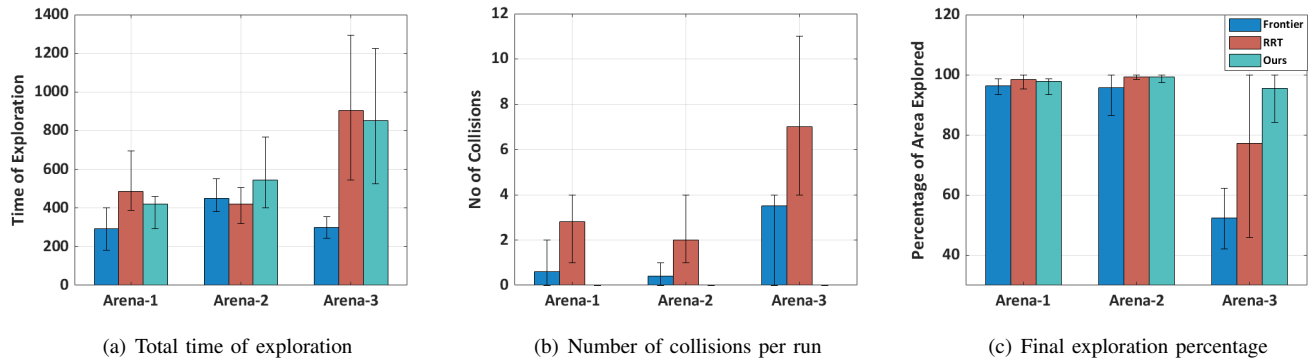


Fig. 4: Comparison of different approaches in three different arenas. Vertical lines in the bar graph indicate the minimum and maximum of collected data.

proposed motion planner uses a symbolic control approach. Despite its potential to solve many real-world issues, the method is generally computationally expensive. Thus current safe exploration framework is limited to a static environment. However, the approach is highly parallelizable. Therefore, in future work, we plan to leverage high-performance computing to solve the problem of safe exploration in the presence of dynamic obstacles, multi-robot exploration, and 3D exploration.

REFERENCES

- [1] M. Althoff and B. H. Krogh. Reachability analysis of nonlinear differential-algebraic systems. *IEEE Transactions on Automatic Control*, 59(2):371–383, Feb 2014.
- [2] E. Asarin, T. Dang, and A. Girard. Reachability analysis of nonlinear systems using conservative approximation. In *International Workshop on Hybrid Systems: Computation and Control*, pages 20–35. Springer, 2003.
- [3] A. Bajcsy, S. Bansal, E. Bronstein, V. Tolani, and C. J. Tomlin. An efficient reachability-based framework for provably safe autonomous navigation in unknown environments. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 1758–1765, 2019.
- [4] V. Cavinato, T. Eppenberger, D. Youakim, R. Siegwart, and R. Dub. Dynamic-aware autonomous exploration in populated environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1312–1318, 2021.
- [5] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, 1997.
- [6] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen. Directed hypergraphs and applications. *Discrete applied mathematics*, 42(2-3):177–201, 1993.
- [7] H. Gao, X. Zhang, J. Wen, J. Yuan, and Y. Fang. Autonomous indoor exploration via polygon map construction and graph-based SLAM using directional endpoint features. *IEEE Transactions on Automation Science and Engineering*, 16(4):1531–1542, 2019.
- [8] W. Gao, M. Booker, A. Adiwahono, M. Yuan, J. Wang, and Y. W. Yun. An improved frontier-based approach for autonomous exploration. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 292–297, 2018.
- [9] P. Jagtap, F. Abdi, M. Rungger, M. Zamani, and M. Caccamo. Software fault tolerance for cyber-physical systems via full system restart. *ACM Trans. Cyber-Phys. Syst.*, 4(4), aug 2020.
- [10] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient sparse pose adjustment for 2D mapping. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–29, 2010.
- [11] M. Lauri and R. Ritala. Planning for robotic exploration based on forward simulation. *Robotics and Autonomous Systems*, 83:15–31, 2016.
- [12] W. C. Lee, M. C. Lim, and H. L. Choi. Extendable navigation network based reinforcement learning for indoor robot exploration. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11508–11514, 2021.
- [13] C. Papachristos, M. S. Kamel, M. Popovic, S. Khattak, A. Bircher, H. Oleynikova, T. Dang, F. Mascari, K. Alexis, and R. Siegwart. *Autonomous Exploration and Inspection Path Planning for Aerial Robots Using the Robot Operating System*, pages 67–111. 01 2019.
- [14] G. Reissig. Computing abstractions of nonlinear systems. *IEEE Transactions on Automatic Control*, 56(11):2583–2598, 2011.
- [15] G. Reissig, A. Weber, and M. Rungger. Feedback refinement relations for the synthesis of symbolic controllers. *IEEE Transactions on Automatic Control*, 62(4):1781–1796, 2017.
- [16] C. Roesmann, W. Feiten, T. Woesch, F. Hoffmann, and T. Bertram. Trajectory modification considering dynamic constraints of autonomous robots. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, 2012.
- [17] R. Shrestha, F. P. Tian, W. Feng, P. Tan, and R. Vaughan. Learned map prediction for enhanced mobile robot exploration. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1197–1204, 2019.
- [18] A. Singletary, T. Gurriet, P. Nilsson, and A. D. Ames. Safety-critical rapid aerial exploration of unknown environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10270–10276, 2020.
- [19] E. D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science and Business Media, 2013.
- [20] P. Tabuada. *Verification and control of hybrid systems: a symbolic approach*. Springer Science and Business Media, 2009.
- [21] M. Thoresen, N. H. Nielsen, K. Mathiassen, and K. Y. Pettersen. Path planning for UGVs based on traversability hybrid A*. *IEEE Robotics and Automation Letters*, 6(2):1216–1223, 2021.
- [22] H. Umari and S. Mukhopadhyay. Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1396–1402, 2017.
- [23] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151, 1997.
- [24] D. Zhu, T. Li, D. Ho, C. Wang, and M. Q.-H. Meng. Deep reinforcement learning supervised autonomous exploration in office environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7548–7555, 2018.
- [25] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li. A comparative analysis of LiDAR SLAM-Based indoor navigation for autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):6907–6921, 2022.