2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Safe Multi-Robot Exploration using Symbolic Control

Manas Sashank Juvvi‡, David Smith Sundarsingh‡, Ratnangshu Das‡, Pushpak Jagtap

***Abstract*— Multi-robot exploration is a complex problem that involves multiple robots working in a shared unknown environment. In such scenarios, the safety of the robots is of paramount importance alongside the completion of the exploration task. In this paper, we propose a modular exploration framework that (i) identifies safe frontier targets for multiple robots while taking into account the system dynamics of each robot to ensure collision avoidance with previously unknown obstacles and (ii) ensures that the robots reach their exploration targets while avoiding any obstacles discovered and each other. We employ a scalable approach to generate symbolic controllers for the multi-robot system, utilizing distance functions. We also provide formal guarantees on the safety of the exploration targets and the completion of each exploration run, with the robots avoiding collisions with each other and the obstacles. We test our approach on simulation experiments and a real-world implementation to validate it.**

I. INTRODUCTION

The domain of multi-robot exploration, where multiple intelligent robots collectively navigate and explore territories, has garnered substantial attention. This holds immense promise to bring about major changes in fields such as robotics, environmental surveillance [1], search and rescue operations [2], and many more.

Exploration is primarily done in two ways: decision-theoretic approaches or frontier-based approaches. Decision-theoretic exploration [3], [4] relies on maximizing the potential information gain associated with different decisions and then subsequently generating trajectories for exploration. Modified versions of this approach employed in the context of a learning framework for multi-robot systems have been reported in [5], [6]. However, these techniques require some knowledge of the environment, are computationally expensive, and sometimes fall back to frontier exploration methods when they fail [7]. Frontier-based exploration [8], on the other hand, involves identifying unexplored frontiers and directing exploration efforts toward these regions. Several modifications to the approach have also been reported in the literature, which include maintaining the memory of all frontiers to avoid revisits [9], employing multiple Rapidly-exploring Randomized Trees (RRT) for frontier detection [10] and dynamic frontiers [11] for tackling dynamic obstacles. Moreover, frontier exploration methods are usually preferred due to their intuitive nature and ease of use. However, the robots are directed toward frontiers without knowing what lies ahead and can lead to situations where robots come close to unobserved obstacles. Due to the robots' state or input constraints, they may be unable to navigate safely, risking collisions.

The majority of the existing exploration approaches largely rely on heuristics and prioritize the speed and extent of exploration over the safety of the robot, the environment, or both. The environments, however, often pose significant challenges due to their inherent uncertainty, including unpredictable terrain variations and the presence of other dynamic robots. Therefore, ensuring the safety of both the robots and their surroundings is incredibly important. There are a few approaches that consider safety as a factor. Notably, authors in [12] use control barrier functions as a safety shield over an existing planner. However, the approach is limited to control-affine systems and control inputs are unbounded. Also, [13] 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.

In this paper, we develop a framework for generating maps using multi-robot systems while ensuring formal safety guarantees considering all the following factors: (i) the dynamics and dimensions of the robot, (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. We use the frontier-based exploration technique and leverage the symbolic control approach [14], which consists of creating a symbolic model of the robot and refining the controller synthesized for the symbolic model to a controller for the original robot, thereby guaranteeing safe navigation to the target. The contributions of the paper are highlighted below:

- An exploration-target detection algorithm, leveraging the knowledge of robot dynamics and constraints to generate safe targets.
- An automated, scalable, correct-by-construction motion planning module that takes into account all the practical constraints of the system and navigates each robot to its corresponding targets.

II. PRELIMINARIES AND PROBLEM DEFINITION

A. Notations

The symbols $\mathbb{N}$, $\mathbb{N}_0$, $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{R}^+$, and $\mathbb{R}_0^+$ denote the set of natural, non-negative integer, integer, real, positive, and non-negative real numbers, respectively. For $a, b \in (\mathbb{R} \cup$

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

‡Authors contributed equally.

D. S. Sundarsingh, M. S. Juvvi, R. Das, and P. Jagtap are with Robert Bosch Centre for Cyber-Physical Systems, IISc, Bangalore, India {davids,manasjuvvi,ratnangshud,pushpak}@iisc.ac.in

$\{-\infty, \infty\})^n$, where $a \leqslant b$ component-wise, the closed hyper-interval is denoted by $[\![a, b]\!] := \mathbb{R}^n \cap ([a_1, b_1] \times \cdots \times [a_n, b_n])$. A relation $R \subseteq A \times B$ with the map $R : A \to 2^B$ is defined as $b \in R(a)$ iff $(a, b) \in R$ and it is strict if $R(a) \neq \varnothing$, for all $a \in A$. The inverse of the relation is defined as $R^{-1} := \{(b, a) \in B \times A | (a, b) \in R\}$ and can be written as $a \in R^{-1}(b)$. $[1; N] := \{1, \ldots, N\}$, where $N \in \mathbb{N}$. x_q represents the q^{th} element of the vector $x \in \mathbb{R}^n$, where $q \in [1; n]$ and the infinity norm of the vector is $\|x\| := \mathbf{max}_{q \in [1;n]} |x_q|$. The vector $\mathbf{1}_n$ is an n-dimensional vector having all entries as 1. Consider N sets A_i, $i \in [1; N]$, the Cartesian product of the sets is given by $A = \prod_{i \in [1;N]} A_i := \{(a_1, \ldots, a_N) | a_i \in A_i, \forall i \in [1; N]\}$. Given N functions $f_i : X_i \to A_i$, the Cartesian product of the functions is $f : X \to A := \prod_{i \in [1;N]} f_i = (f_1(x_1), \ldots, f_N(x_N))$. The composition of two maps H and R is $H \circ R(x) := H(R(x))$. If $\eta \in \mathbb{R}^n$, then $\eta\mathbb{Z}^n = \{c \in \mathbb{R}^n | \exists_{l \in \mathbb{Z}^n} \forall_{q \in [1;n]} c_q = l_q \eta_q\}$.

B. Multi-Robot Systems

Consider a collection of $N \in \mathbb{N}$ robots and let $I = [1; N]$. Each robot's state evolution is given by the following discrete-time control system:

$$\mathsf{x}_i(k+1) = \mathsf{f}_i(\mathsf{x}_i(k), \mathsf{u}_i(k)),\ i \in I,\ k \in \mathbb{N}_0, \qquad (1)$$

where $\mathsf{x}_i(k) \in X_i \subset \mathbb{R}^{n_i}$ is the state of the i^{th} robot and $\mathsf{u}_i(k) \in U_i \subset \mathbb{R}^{m_i}$ is the input to the robot. The map f_i is defined as $\mathsf{f}_i : X_i \times U_i \to X_i$. The state evolution of the multi-robot system is given by:

$$\mathsf{x}(k+1) = \mathsf{f}(\mathsf{x}(k), \mathsf{u}(k)),\ k \in \mathbb{N}_0,\ x(0) \in X^0, \qquad (2)$$

where X^0 is the set of initial states, $\mathsf{x}(k) \in X := \prod_{i \in I} X_i \subset \mathbb{R}^n$ is the state of the multi-robot system, $n = \Sigma_{i \in I} n_i$, $\mathsf{u}(k) \in U := \prod_{i \in I} U_i \subset \mathbb{R}^m$ is the input to the system and $m = \Sigma_{i \in I} m_i$. The function f is given by $\mathsf{f} : X \times U \to X$ and $\mathsf{f}(\mathsf{x}(k), \mathsf{u}(k)) := \prod_{i \in I} \mathsf{f}_i(\mathsf{x}_i(k), \mathsf{u}_i(k))$.

The reachable set from a set $\mathcal{X} \subseteq X$ under an input $u \in U$ is given by $Reach(\mathcal{X}, u) := \bigcup_{x \in \mathcal{X}} \mathsf{f}(x, u)$, which is the set of all states that the system "reaches" when an input u is applied at all the states in $\mathcal{X}$ in a one-time step. This reachable set is difficult to compute, so we use the over-approximated reachable set, $\overline{Reach}(x, u)$. Several approaches are available in the literature for computing this over-approximated set; for example, [15], [16], [17] and [18].

C. Transition Systems

We now introduce the notion of transition systems [14] which will be used as a unified representation for the robot model (1) and their corresponding symbolic models.

Definition 2.1: A transition system is a tuple $\Sigma = (X, X^0, U, F)$, where X is the set of states (possibly infinite), $X^0 \subseteq X$ is the set of initial states, U is the set of inputs (possibly infinite), and the map $F : X \times U \rightrightarrows X$ is the transition relation.

The set of admissible inputs for $x \in X$ is denoted by $U^a(x) := \{u \in U \mid F(x, u) \neq \varnothing\}$. We use $x' \in F(x, u)$ to represent the u-successor (or successor) of state x.

The transition system representation of the i^{th} robot is given by the tuple $\Sigma_i = (X_i, X_i^0, U_i, F_i)$, where $X_i \subset \mathbb{R}^{n_i}$ is the set of the states of robot i, $X_i^0 \subseteq X_i$ is the set of initial states, $U_i \subset \mathbb{R}^{m_i}$ is the set of inputs available for robot i, and for $x_i \in X_i$, $u_i \in U_i$, $F_i(x_i, u_i) := \mathsf{f}_i(x_i, u_i)$.

The composition of the N transition systems, which represents the multi-robot system (2) is defined below.

Definition 2.2: Given a collection of $N \in \mathbb{N}$ robots represented as $\{\Sigma_i\}_{i \in I}$, where $I = [1; N]$, the composed transition system is $\Sigma = \big(X, X^0, U, F\big)$, where

- $X = \prod_{i \in I} X_i$, $X^0 \subseteq X$, $U = \prod_{i \in I} U_i$,
- for $x \in X$ and $u \in U$, $F(x, u) = \prod_{i \in I} F_i(x_i, u_i)$.

D. Map Parameters

Consider a partially known 2D environment $P \subset \mathbb{R}^2$ represented by a uniform occupancy grid $\mu\mathbb{Z}^2$ with quantization parameter $\mu \in \mathbb{R}^2$, by existing SLAM algorithms. This grid is described for robot i by a matrix M_i and the indices corresponding to $c \in \mu\mathbb{Z}^2$ in M_i can be obtained using the index function $Id_i : \mu\mathbb{Z}^2 \to \mathbb{N}^2$, where $i \in [1; N]$. A point $c \in \mu\mathbb{Z}^2$ is an obstacle for robot i if $M_i(Id_i(c)) = 1$, a free space if $M_i(Id_i(c)) = 0$, and unexplored if $M_i(Id_i(c)) = -1$. Now we define free region P_f, obstacle sets P_o and unexplored region P_u as,

$$P_o = \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } \mathbf{max}_{i \in [1;N]} M_i(Id_i(c)) = 1} c + \Big[\!\Big[-\frac{\mu}{2}, \frac{\mu}{2}\Big]\!\Big],$$

$$P_u = \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } \mathbf{max}_{i \in [1;N]} M_i(Id_i(c)) = -1} c + \Big[\!\Big[-\frac{\mu}{2}, \frac{\mu}{2}\Big]\!\Big],$$

$$P_f = \bigcup_{\forall c \in \mu\mathbb{Z}^2 \text{ s.t. } \mathbf{max}_{i \in [1;N]} M_i(Id_i(c)) = 0} c + \Big[\!\Big[-\frac{\mu}{2}, \frac{\mu}{2}\Big]\!\Big].$$

The final merged map is given by $P := P_o \cup P_u \cup P_f$. We denote state spaces of robot i corresponding to P_f, P_o, and P_u as $X_{if} = P_f \times O_i$, $X_{io} = P_o \times O_i$ and $X_{iu} = P_u \times O_i$, respectively, where $i \in [1; N]$ and $O_i \subset \mathbb{R}^{n_i - 2}$ denotes all the states of robot i except its 2-D position, such as orientation, velocities, etc. In our approach, we consider the unexplored region and the occupied region as the obstacle space for each robot given by $Obs_i = ((P_o \cup P_u) + [\![-r_{robot_i}\mathbf{1}_2, r_{robot_i}\mathbf{1}_2]\!]) \times O_i$, where r_{robot_i} is the radius of the smallest circle that completely encloses robot i. This over-approximation is provided as the symbolic controller considers the system to be a point-mass.

Remark 2.3: The map definitions show that if one of the robots identifies a grid point $c \in \mu\mathbb{Z}^2$ as an obstacle, i.e. $M_i(Id_i(c)) = 1$, $i \in [1; N]$, then that grid is an obstacle in the merged map. This places more emphasis on safety.

E. Problem Formulation

Next, we state the problem considered in this paper:

Problem 2.4: Given a multi-robot system as in (2) with $N \in \mathbb{N}$ robots and a closed environment $P \subset \mathbb{R}^2$, design an automated exploration framework that incrementally explores the environment, while guaranteeing that each robot avoids obstacles and collisions with other robots, till no safe exploration targets can be found.

We aim to solve the Problem 2.4 by generating a safe exploration target for each robot and synthesizing a symbolic controller that ensures that the robots reach the target while avoiding obstacles and collisions with each other. We propose the following four-step process:

1) We first find a set of ranked exploration targets for each robot in the multi-robot system (MRS).
2) We then synthesize controllers that navigate each robot towards the corresponding best targets while avoiding obstacles.
3) Using a distance function, we compose the individual controlled systems in order to eliminate transitions and states in the overall composed system that do not maintain the specified distance between robots.
4) Since a few transitions that solved the reachability problem for each robot might have been removed during step (3), we synthesize the symbolic controller again for the composed system that ensures that all robots reach their corresponding targets.

This final controlled system is such that each robot moves towards its assigned target without colliding with other robots or obstacles. Once the robots reach their targets, the process is repeated till there are no more safe targets available.

III. Proposed Exploration Framework

This section introduces the overall exploration approach, which involves four steps: (i) finding safe exploration targets for each robot using a Safe Frontier Target Detector (SFTD), (ii) construction of a symbolic model of each robot and synthesizing reachability controllers for them, (iii) composition of the individual controlled systems using a distance function and (iv) controller synthesis that ensures that all robots reach their corresponding targets despite enforcement of distance maintenance in step (iii). Controller synthesis is done based on the mathematical model of the system and constraints, which eliminates impractical input values.

A. Safe Frontier Target Detector (SFTD)

We first generate targets for robot $i \in [1; N]$. We adapt frontier-based exploration [8] by finding a free region near frontiers. As one of the targets may not have a safe trajectory for robot i, we will generate multiple targets and rank them based on the size of neighbouring unexplored regions.

Given a map P and the robot dynamics (1), the algorithm aims to find a set of j_i possible frontier target regions $\mathcal{F}_{iT} = \{F_i^1, \ldots, F_i^{j_i}\}$ for robot i, where

$$F_i^m = (T_i^m \cup S_i^m) + [\![-\phi_{win_i}\mathbf{1}_{n_i}, \phi_{win_i}\mathbf{1}_{n_i}]\!], m \in [1; j_i]. \quad (3)$$

Here, the target region T_i^m is defined as:

$$T_i^m = p_i^m + [\![-\tau_{win_i}\mathbf{1}_{n_i}, \tau_{win_i}\mathbf{1}_{n_i}]\!] \subset X_{if}, \quad (4)$$

where $p_i^m \in X_{if}$, $m \in \{1, \ldots, j_i\}$ and $\tau_{win_i} \in \mathbb{R}^+$ is the target window size. We choose $\tau_{win_i} \geqslant r_{robot_i}$, where r_{robot_i} is the radius of the smallest circle that encloses robot i. Given target set T_i^m, we define the safety window as:

$$S_i^m := \bigcup_{x_i \in T_i^m} \bigcup_{u_i \in U_i^a(x_i)} \overline{Reach}_i(x_i, u_i) \text{ such that } S_i^m \subset X_{if}, \quad (5)$$

and $\phi_{win_i} \in \mathbb{R}^+$ is chosen to accommodate the over-approximation of obstacles in the motion planning phase and to ensure the presence of frontiers and S_i^m ensures that the robot does not collide with a previously unknown obstacle. The frontier targets are chosen such that $F_i^m \backslash (T_i^m \cup S_i^m) \cap X_{iu} \neq \varnothing$ to ensure the presence of a frontier. For illustration, refer to Figure 1, where the brown region represents the target window, the green region represents the safety window, and the red region represents the frontier window. The next theorem shows that the safety window S_i^m will ensure that robot i will never collide with any previously unknown obstacles near the exploration target for the next transition. This ensures that the robot can move to the next target after reaching the previous one while preserving safety.

Theorem 3.1: Consider robot i as given in (1), state-space X_{if} corresponding to $P_f \subseteq P$ as defined in Section II-D, and a frontier target $F_i^m \in \mathcal{F}_{iT}$ as given in (3). If robot i starts anywhere inside $T_i^m \subset F_i^m$, it will not enter a newly explored region.

Proof: The safety window corresponding to F_i^m is given by, $S_i^m = \bigcup_{x_i \in T_i^m} \bigcup_{u_i \in U_i^a(x_i)} \overline{Reach}_i(x_i, u_i)$, such that $S_i^m \subset X_{if}$. Thus, robot i, when starting from any $x_i \in T_i^m$, under any input $u_i \in U_i^a(x_i)$ will reach a state $x_i' \in \overline{Reach}_i(x_i, u_i) \subset S_i^m$ which is fully known and obstacle-free. This allows the robot to take one transition without reaching previously unexplored regions near the target. ■

This target structure is proposed to ensure that robot i always has some space to move around without collision even if an obstacle, identified as unknown before exploration, is present very close to the target.

First, the map is split into smaller submaps based on the position of each robot in the environment. Robots are assigned their respective regions to locate targets, which are then ranked based on the size of the neighbouring unexplored region. The target with the smallest adjacent unexplored region is ranked first in order to avoid oscillation between unexplored regions during exploration. The frontier targets F_i^m, where $i \in [1; N]$, $m \in [1; j_i]$ and j_i is the number of targets in each submap, are then computed for each robot i, ranked and their target windows $T_i^m \subset F_i^m$ are then stored in $\mathcal{T}_i$ in the ranked order for each robot. This set $\mathcal{T}_i$ is given to the controller synthesis module for the generation of a strategy that navigates robot i to $T_i^m \in \mathcal{T}_i$.

B. Controller Synthesis for Each Robot for Reaching Target

In this subsection, we describe the construction of a symbolic model of robot $i \in [1; N]$ given by (1), where

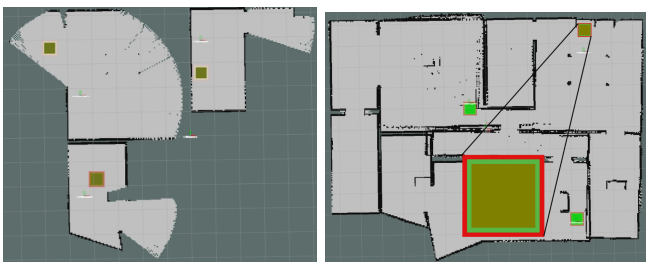


Fig. 1: Stages of exploration with target structure

$N \in \mathbb{N}$ is the number of robots in the multi-robot system, and controller synthesis for navigating the robot to a target $T_i^m \in \mathcal{T}_i$. In order to relate robot i given by the transition system $\Sigma_i = (X_i, X_i^0, U_i, F_i)$ with its symbolic model $\hat{\Sigma}_i = (\hat{X}_i, \hat{X}_i^0, \hat{U}_i, \hat{F}_i)$, we use the notion of feedback refinement relation [18].

Definition 3.2: Consider two transition systems Σ_i and $\hat{\Sigma}_i$. A strict relation $Q_i \subseteq X_i \times \hat{X}_i$ is said to be the feedback refinement relation from Σ_i to $\hat{\Sigma}_i$, denoted by $\Sigma_i \preceq_{Q_i} \hat{\Sigma}_i$, if for each $(x_i, \hat{x}_i) \in Q_i$ the following conditions hold:

- $\hat{U}_i^a(\hat{x}_i) \subseteq U_i^a(x_i)$,
- $u_i \in \hat{U}_i^a(\hat{x}_i) \implies Q_i(F_i(x_i, u_i)) \subseteq \hat{F}_i(\hat{x}_i, u_i)$.

All inputs available for the system $\hat{\Sigma}_i$ is also available for Σ_i and for any input available to $\hat{\Sigma}_i$, with the feedback refinement relation Q_i, the transition $\hat{F}_i(\hat{x}_i, u_i)$ is associated with the corresponding transition $F_i(x_i, u_i)$ of the concrete system Σ_i. Thus any controller built for the system $\hat{\Sigma}_i$ can be refined using Q_i to make it compatible with Σ_i.

We now construct a symbolic model $\hat{\Sigma}_i$ such that the symbolic model includes the corresponding behaviour of the system Σ_i through the feedback refinement relation Q_i.

Definition 3.3: The symbolic model of the system Σ_i is given by $\hat{\Sigma}_i = (\hat{X}_i, \hat{X}_i^0, \hat{U}_i, \hat{F}_i)$, where

- $\hat{X}_i$ is a cover over X_i whose elements are non-empty, closed hyper-intervals called cells. Let $\bar{\hat{X}}_i \subseteq \hat{X}_i$ be a compact set of congruent hyper-rectangles aligned on a uniform grid parameterized with a quantization parameter $\eta_i \in (\mathbb{R}^+)^{n_i}$. Each $\hat{x}_i \in \bar{\hat{X}}_i$ is given by $c_{\hat{x}_i} + [\![-\frac{\eta_i}{2}, \frac{\eta_i}{2}]\!]$, where $c_{\hat{x}_i} \in \eta_i \mathbb{Z}^{n_i}$. The cells in $\hat{X}_i \backslash \bar{\hat{X}}_i$ are called overflow cells,
- $\hat{X}_i^0 \subseteq \hat{X}_i$, $\hat{U}_i$ is a finite subset of U_i,
- for $\hat{x}_i \in \bar{\hat{X}}_i \backslash Obs_i$ and $\hat{u}_i \in \hat{U}_i$, a set $A := \{\hat{x}_i' \in \hat{X}_i | \hat{x}_i' \cap \overline{Reach}_i(\hat{x}_i, \hat{u}_i) \neq \varnothing\}$. If $A \subseteq \bar{\hat{X}}_i$, $\hat{x}_i' \notin \hat{X}_i \backslash \bar{\hat{X}}_i$, $\forall \hat{x}_i' \in A$ and $A \cap Obs_i = \varnothing$, then $\hat{F}_i(\hat{x}_i, \hat{u}_i) = A$. Otherwise, $\hat{F}_i(\hat{x}_i, \hat{u}_i) = \varnothing$.

For a detailed procedure on constructing the symbolic model, kindly refer to [19].

Theorem 3.4: [20, Theorem 4.3] If $\hat{\Sigma}_i$ is the symbolic model of Σ_i which satisfies formal behavioural inclusions in terms of the feedback refinement relation $Q_i \subseteq X_i \times \hat{X}_i$, $\eta_i \in (\mathbb{R}^+)^{n_i}$ is the quantization parameter and Obs_i is an obstacle set, then $\Sigma_i \preceq_{Q_i} \hat{\Sigma}_i$.

Remark 3.5: By construction, the symbolic model of the system does not include any transitions that lead into the obstacle. Thus, any controller designed for this model will not navigate the robot towards the obstacle.

The next step is the synthesis of the controller for the symbolic model and the refined controller that ensures that the concrete system Σ_i accomplishes the reachability task. Consider the transition system $\Sigma_i = (X_i, X_i^0, U_i, F_i)$ and a controller $C_i : X_i \rightrightarrows U_i$, where for all $x_i \in X_i$, $C_i(x_i) \subseteq U_i^a(x_i)$. Let the domain of the controller be $dom(C_i) := \{x_i \in X_i | C_i(x_i) \neq \varnothing\}$.

We first construct a controller $\hat{C}_i$ for the symbolic model $\hat{\Sigma}_i$ using graph theoretical methods [14] that will navigate the robot i to the best-ranked target $T_i^m \in \mathcal{T}_i$.

Theorem 3.6: [18, Theorem VI.3] If $\Sigma_i \preceq_{Q_i} \hat{\Sigma}_i$ and $\hat{C}_i$ is the symbolic controller that navigates $\hat{\Sigma}_i$ to $T_i^m \in \mathcal{T}_i$, then $C_i := \hat{C}_i \circ Q_i$ solves the reachability of T_i^m for Σ_i.

The refined controller C_i obtained by refining the symbolic controller $\hat{C}_i$ using Q_i will ensure that the system Σ reaches T_i^m. Toolboxes like [19], [21], and [22] can be used for symbolic controller synthesis.

Definition 3.7: Given a controller C and a transition system Σ, the controlled system is given by the tuple $\Sigma | C = (X_C, X_C^0, U_C, F_C)$, where

- $X_C = X \cap dom(C)$, $X_C^0 \subseteq X_C$, $U_C = U$,
- for $x_C \in X_C$ and $u_C \in U_C$, $x_C' \in F_C(x_C, u_C)$ iff $x_C' \in F(x_C, u_C)$ and $u_C \in C(x_C)$.

Using Definition 3.7, we construct the symbolic controlled systems of the individual robots as $\hat{\Sigma}_i | \hat{C}_i = (\hat{X}_{\hat{C}_i}, \hat{X}_{\hat{C}_i}^0, \hat{U}_{\hat{C}_i}, \hat{F}_{\hat{C}_i})$.

C. Composition of Individual Controlled Systems

The next step involves the composition of the controlled systems and the construction of a controlled MRS that ensures that each robot maintains a prescribed distance from another using a distance function. Given a collection of controlled systems and their symbolic model $\{\hat{\Sigma}_i | \hat{C}_i\}_{i \in [1;N]}$, we compose them as shown in Definition 2.2. Since $\Sigma_i \preceq_{Q_i} \hat{\Sigma}_i$ and the controller $C_i := \hat{C}_i \circ Q_i$, we can say that $\Sigma_i | C_i \preceq_{Q_i} \hat{\Sigma} | \hat{C}_i$, as evident from [18, Theorem VI.3] and [18, Corollary VI.5], and after composition of the controlled systems, we have $\Sigma | C \preceq_Q \hat{\Sigma} | \hat{C}$ as there is no coupling between the robots. A control input $u = (u_1, \ldots, u_N) \in C(x)$ iff $u_i \in C_i(x_i)$, $\forall i \in [1; N]$, where $x = (x_1, \ldots, x_N)$ and a discretized control input $\hat{u} = (\hat{u}_1, \ldots, \hat{u}_N) \in \hat{C}(\hat{x})$ iff $\hat{u_i} \in \hat{C}_i(\hat{x}_i)$, $\forall i \in [1; N]$, where $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_N)$. The strict relation $Q \subset X \times \hat{X}$ is defined as $Q := \{(x, \hat{x}) \in X \times \hat{X} | (x_i, \hat{x}_i) \in Q_i, \ \forall i \in [1; N]\}$. The discretization parameter of the composed system is $\eta = \prod_{g \in [1;N]} \eta_g$.

Consider a distance function defined as,

$$dist(x_{ij}) := \|x_i - x_j\| - d_{ij}, \tag{6}$$

where $x_{ij} = [x_i, x_j]$, x_i, x_j are the states of the robots i and j, respectively when the MRS is in state x, $i, j \in [1; N]$ and $i \neq j$ and d_{ij} is the distance that robot i and j must maintain between each other. However, note that the controlled systems are represented by their symbolic model $\hat{\Sigma}_i | \hat{C}_i = (\hat{X}_{\hat{C}_i}, \hat{X}_{\hat{C}_i}^0, \hat{U}_{\hat{C}_i}, \hat{F}_{\hat{C}_i})$, where each symbol $\hat{x}_i \in \hat{X}_{\hat{C}_i}$ has infinite states in $\mathbb{R}^n$, making it impossible to verify that robot i maintains the distance d_{ij} at all states in $\hat{x}_i$. Hence, we modify the distance function $dist(x_{ij})$ to make it compatible with the symbolic model. The modified distance function is defined as $\hat{dist}(x_{ij}) := dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2}$, where $\eta_{ij} := \max\{\eta_i, \eta_j\}$, $(x_i, \hat{x}_i) \in Q_i$, $(x_j, \hat{x}_j) \in Q_j$, $\hat{x}_{ij} = c_{\hat{x}_{ij}} + [\![\frac{-\eta_{ij}}{2}, \frac{\eta_{ij}}{2}]\!]$ and $c_{\hat{x}_{ij}} = [c_{\hat{x}_i}, c_{\hat{x}_j}]$.

Lemma 3.8: Consider robots $i, j \in [1; N]$ represented by Σ_i and Σ_j respectively, their corresponding symbolic model $\hat{\Sigma}_i$ and $\hat{\Sigma}_j$ such that $\Sigma_i \preceq_{Q_i} \hat{\Sigma}_i$ and $\Sigma_j \preceq_{Q_j} \hat{\Sigma}_j$, where $Q_i \subset X_i \times \hat{X}_i$ and $Q_j \subset X_j \times \hat{X}_j$ defined by $Q_i = \{(x_i, \hat{x}_i) \in X_i \times \hat{X}_i : x_i \in \hat{x}_i\}$ and $Q_j = \{(x_j, \hat{x}_j) \in X_j \times \hat{X}_j : x_j \in \hat{x}_j\}$

respectively are strict relations, $\eta_{ij} := \max\{\eta_i, \eta_j\}$ and the distance function $dist(x_{ij})$, as defined in (6). If $dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2} \geqslant 0$, then $dist(x_{ij}) \geqslant 0$, $\forall (x_i, \hat{x}_i) \in Q_i$ and $\forall (x_j, \hat{x}_j) \in Q_j$.

Proof: Consider a symbol $\hat{x}_{ij}$. Given the discretized state-space, we know that $\|c_{\hat{x}_{ij}} - x_{ij}\| \leqslant \frac{\eta_{ij}}{2}$, $\forall x_i \in Q_i^{-1}(\hat{x}_i)$ and $\forall x_j \in Q_j^{-1}(\hat{x}_j)$. Now, $dist(c_{\hat{x}_{ij}}) - dist(x_{ij}) \leqslant \|dist(c_{\hat{x}_{ij}}) - dist(x_{ij})\| \leqslant \|c_{\hat{x}_{ij}} - x_{ij}\| \leqslant \frac{\eta_{ij}}{2}, dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2} \leqslant dist(x_{ij})$. Thus, $dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2} \geqslant 0 \Longrightarrow dist(x_{ij}) \geqslant 0$, $\forall (x_i, \hat{x}_i) \in Q_i$ and $\forall (x_j, \hat{x}_j) \in Q_j$. ∎

Remark 3.9: This lemma says that we can use the centre point of the symbol to ensure that each robot in the multi-robot system maintains a safe distance from another robot, irrespective of the state of each robot within the symbol.

Definition 3.10: Given the distance function $dist(x_{ij})$ defined as in (6) and the symbolic model of the controlled multi-robot system $\hat{\Sigma}|\hat{C} = (\hat{X}_C, \hat{X}_C^0, \hat{U}_C, \hat{F}_C)$, the safety controller $\hat{C}_S$ that ensures that the robots i and j maintain the prescribed distance d_{ij} from each other is given by,

- $dom(\hat{C}_S) \subseteq \hat{X}_C \cap \{\hat{x} | dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2} \geqslant 0, \forall i, j \in [1; N] \text{ and } i \neq j\}$, where $\eta_{ij} = \max\{\eta_i, \eta_j\}$,
- $\hat{C}_S(\hat{x}) = \hat{U}_C^a(\hat{x}) \cap \{\hat{u} | \mathbf{max}_{\hat{x}' \in \hat{F}(\hat{x}, \hat{u})} [dist(c_{\hat{x}'_{ij}}) - dist(c_{\hat{x}_{ij}})] \geqslant -\gamma \times (dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2}), \forall i, j \in [1; N] \text{ and } i \neq j\}$, where $\gamma \in (0, 1)$.

We now construct the controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S = (\hat{X}_S, \hat{X}_S^0, \hat{U}_S, \hat{F}_S)$ using the safety controller $\hat{C}_S$ as shown in Definition 3.7.

Lemma 3.11: Given the safety controller $\hat{C}_S$ defined in Definition 3.10, the composed system $\hat{\Sigma}|\hat{C}$ and the controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S = (\hat{X}_S, \hat{X}_S^0, \hat{U}_S, \hat{F}_S)$ constructed as shown in Definition 3.7, $\forall \hat{x} \in \hat{X}_S$ and any $\hat{u} \in \hat{U}_S^a(\hat{x})$, $dist(c_{\hat{x}'_{ij}}) - \frac{\eta_{ij}}{2} \geqslant 0, \forall \hat{x}' := c_{\hat{x}'} + [\![\frac{-\eta}{2}, \frac{\eta}{2}]\!] \in \hat{F}_S(\hat{x}, \hat{u})$, $\forall i, j \in [1; N]$ and $i \neq j$.

Proof: Since $\hat{U}_S^a(\hat{x}) \subseteq \hat{C}_S(\hat{x})$, for all $\hat{x} \in \hat{X}_S$, $\forall i, j \in [1; N]$ and any $\hat{u} \in \hat{U}_S^a(\hat{x})$, $\hat{x}' \in \hat{F}_S(\hat{x}, \hat{u})$ is such that,

$$
\begin{aligned}
&dist(c_{\hat{x}'_{ij}}) - dist(c_{\hat{x}_{ij}}) \geqslant -\gamma \times \left(dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2}\right), \\
&dist(c_{\hat{x}'_{ij}}) - \frac{\eta_{ij}}{2} - dist(c_{\hat{x}_{ij}}) + \frac{\eta_{ij}}{2} \\
&\geqslant -\gamma \times \left(dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2}\right), \\
&dist(c_{\hat{x}'_{ij}}) - \frac{\eta_{ij}}{2} \geqslant (1 - \gamma) \times \left(dist(c_{\hat{x}_{ij}}) - \frac{\eta_{ij}}{2}\right).
\end{aligned}
$$

Since $\forall \hat{x} \in \hat{X}_S$, $dist(c_{\hat{x}_{ij}}) - \frac{\eta_{max}}{2} \geqslant 0$ and $\gamma \in (0, 1)$, we have $dist(c_{\hat{x}'_{ij}}) - \frac{\eta_{ij}}{2} \geqslant 0$. ∎

Remark 3.12: The lemma shows that any transition in the controlled system will lead to a state where the robots still maintain the distance defined by the distance function.
The two lemmas show that the distances between the robots are always maintained in this system.

D. Distributed Controller for Multi-Robot System

During the construction of the controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S$, a few trajectories that accomplish the reachability task may have been removed in order to maintain the distance required. Hence, we synthesize a controller $\hat{C}_B$ for the obtained controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S$ that ensures navigation of each robot $i \in [1; N]$ to its corresponding target T_i^m using graph-theoretic methods. The combination of controllers C_i, where $i \in [1; N]$, $C_S := \hat{C}_S \circ Q$ and $C_B := \hat{C}_B \circ Q$ ensure that each robot of the MRS (2) reaches its target while avoiding obstacles and each other. The following theorem shows this.

Theorem 3.13: Given the controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S$, the relation $Q \subset X \times \hat{X}$, a collection of targets $\{T_i^m\}_{i \in [1;N]}$, where $m \in [1; j_i]$ and j_i is the number of targets for robot i, and the controller $\hat{C}_B$ which navigates $(\hat{\Sigma}|\hat{C})|\hat{C}_S$ towards $\prod_{i \in [1;N]} T_i^m$, where $m \in [1; j_i]$ and j_i is the number of targets for robot i, $C_B := \hat{C}_B \circ Q$ solves the reachability of $\prod_{i \in [1;N]} T_i^m$ for $(\Sigma|C)|C_S$ and the robots maintain the specified distance.

Proof: We know that $\Sigma|C \preceq_Q \hat{\Sigma}|\hat{C}$ and $C_S := \hat{C}_S \circ Q$, and from [18, Theorem VI.3,Corollary VI.5], we have $(\Sigma|C)|C_S \preceq_Q (\hat{\Sigma}|\hat{C})|\hat{C}_S$. From Lemma 3.8 and Definition 3.10, we know that the controller $\hat{C}_S$ ensures that all states in the controlled system $(\Sigma|C)|C_S$ are such that the robots maintain a prescribed distance from each other. And lemma 3.11 shows that any transition in the controlled system $(\hat{\Sigma}|\hat{C})|\hat{C}_S$ leads back to the states where the distance is maintained, and this extends to the concrete system as well due to Lemma 3.8. We now synthesize a controller $\hat{C}_B$ using graph theoretic methods [14] that navigates the system $(\hat{\Sigma}|\hat{C})|\hat{C}_S$ towards the set of targets $\{T_i^m\}_{i \in [1;N]}$, such that $\hat{x} \in \hat{X}_S$ reaches $\prod_{i \in [1;N]} T_i^m$. Given $(\Sigma|C)|C_S \preceq_Q (\hat{\Sigma}|\hat{C})|\hat{C}_S$ and Theorem 3.6, $C_B := \hat{C}_B \circ Q$ solves the reachability of $\prod_{i \in [1;N]} T_i^m$ for $(\Sigma|C)|C_S$. ∎

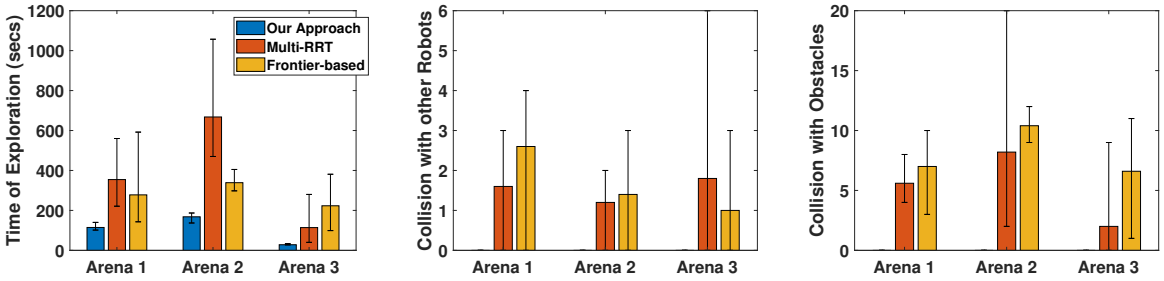
The controller C_B is then used to construct the controlled system $((\Sigma|C)|C_S)|C_B$, which only has states that ensure that robots maintain their prescribed distance from each other and transitions that accomplish reachability of the targets without collision and violation of distance requirements. Thus, by constructing this controlled system from the individual robot models (1) using the target set $\mathcal{T}_i$, where $i \in [1; N]$, obtained from the SFTD, the prescribed distance between robots $i, j \in [1; N]$, $i \neq j$ and the obstacle set Obs_i defined based on map $P \subset \mathbb{R}^2$, we can ensure that the closed-loop MRS will navigate through the environment such that each robot reaches its target without collision with obstacles while maintaining the prescribed distance with all the other robots in the MRS. After the robots reach their targets, the target set $\mathcal{T}_i$ and Obs_i get updated based on the updated map P and the process of controller synthesis continues till no more safe targets are found.

Remark 3.14: Due to the result in theorem 3.13, we must ensure that the robots start inside the region where a trajectory to the target exists and where they are at the specified distance away from each other.

IV. EXPERIMENTAL RESULTS

A. Simulation and Experimental Setup

We implemented the proposed exploration approach on a two and three-robot heterogeneous system. The mathematical models of the robots are given below:



(a) Total time of exploration

(b) Number of collisions with each other per run

(c) Number of collisions with obstacles per run

Fig. 2: Comparison of different approaches in three different arenas. Vertical lines in the bar graph indicate the minimum and maximum of collected data.

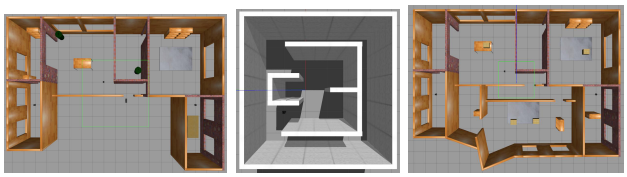


Fig. 3: Simulation environments used for comparison

(i) Differential drive robot (R_1):

$$\begin{aligned} x(k+1) &= x(k) + \upsilon(k)\cos(\theta(k)), \\ y(k+1) &= y(k) + \upsilon(k)\sin(\theta(k)), \\ \theta(k+1) &= \theta(k) + \omega(k), \end{aligned}$$

where $(x(k), y(k))$ is the position and $\theta(k)$ is the orientation of the robot. The bounded control inputs are linear velocity $\upsilon(k) \in [-0.22, 0.22]$ with a discretization of 0.02 and angular velocity $\omega(k) \in [-0.11, 0.11]$, with a discretization of 0.01, respectively.

(ii) Omni-direcrtional robot (R_2 and R_3):

$$\begin{aligned} x(k+1) &= x(k) + \nu(k), \\ y(k+1) &= y(k) + \upsilon(k), \end{aligned}$$

where $(x(k), y(k))$ is the position of the robot. The inputs are, linear velocities, $\nu(k), \upsilon(k) \in [-0.22, 0.22]$ with a discretization of 0.02.

We use a Turtlebot3 differential drive robot (R_1) and two omni-directional robots (R_2) and (R_3) built based on the turtlebot3 hardware in the simulation environment and a two-robot system in the real-world setting. 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. GMapping [23] was used to construct the 2D grid map.

B. Experimental Results

The proposed approach is compared with open-sourced multi-robot exploration frameworks such as Multi-Robot RRT exploration [24] and frontier-based exploration [8]. To compare the repeatability and generality of the algorithms, we conducted a total of 75 sets of exploration runs on three maps with five starting points in each.

Figure 2 compares the exploration techniques in terms of exploration time, number of collisions with other robots and with the obstacles. For comparison, we use the same robots with different approaches and arenas shown in Figure 3. Figure 3 shows an example of the progression of exploration in one of the arenas. The implementation of the proposed symbolic planner is computationally expensive. Parallel versions of the proposed approach [25] can synthesize controllers in real-time, but they require large server-grade processing units, which are not easily available. Thus, we ignore the controller synthesis time for quantifiable comparison with other real-time planners. From Figure 2(a), it can be observed that the exploration time taken is lower compared to Multi-RRT and Frontier-Based exploration due to lack of revisiting explored region. Our approach ensures no collisions during exploration, as shown in Figures 2(b) and (c) leading to a safer exploration framework. The map formed by the other approaches is highly distorted because of the collisions. Our approach considers the constraints of the robot, which ensures that the robots do not collide, reducing the exploration time significantly and leading to the generation of a less distorted map. We also conducted real-world experiments using a two-robot system which are shown in the supplementary material. The video of the implementation can be found at https://www.youtube.com/watch?v=NX40bgxskQc.

V. CONCLUSION AND FUTURE WORK

This work proposed a general safe autonomous exploration framework for multi-robot systems. This is achieved by considering the dynamics and dimensions of robots and practical constraints on states and inputs. We have also provided formal guarantees on the safety of the robots and on the accomplishment of each exploration run. Despite its potential to solve many real-world issues, the method is generally computationally expensive, limiting it to static environments. 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 and in 3D environments.

REFERENCES

[1] A. Singh, F. Ramos, H. D. Whyte, and W. J. Kaiser, "Modeling and decision making in spatio-temporal processes for environmental surveillance," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 5490–5497.

[2] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmen, *Search and Rescue Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1151–1173.

[3] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robotics and Autonomous Systems*, vol. 83, pp. 15–31, 2016.

[4] 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*, vol. 16, no. 4, pp. 1531–1542, 2019.

[5] T. Wang, J. Wang, Y. Wu, and C. Zhang, "Influence-based multi-agent exploration," 2019.

[6] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson, "MAVEN: Multi-Agent Variational Exploration," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[7] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robotics and Autonomous Systems*, vol. 83, pp. 15–31, 2016.

[8] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 1997, pp. 146–151.

[9] 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)*, 2018, pp. 292–297.

[10] 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)*, 2017, pp. 1396–1402.

[11] V. Cavinato, T. Eppenberger, D. Youakim, R. Siegwart, and R. Dubé, "Dynamic-aware autonomous exploration in populated environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1312–1318.

[12] 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)*, 2020, pp. 10 270–10 276.

[13] 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)*, 2019, pp. 1758–1765.

[14] P. Tabuada, *Verification and control of hybrid systems: a symbolic approach*. Springer Science and Business Media, 2009.

[15] A. A. Kurzhanskiy and P. Varaiya, "Reach set computation and control synthesis for discrete-time dynamical systems with disturbances," *Automatica*, vol. 47, no. 7, pp. 1414–1426, 2011.

[16] O. Maler, "Computing reachable sets: An introduction," 2008.

[17] M. Althoff, G. Frehse, and A. Girard, "Set propagation techniques for reachability analysis," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, pp. 369–395, 2021.

[18] G. Reissig, A. Weber, and M. Rungger, "Feedback refinement relations for the synthesis of symbolic controllers," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1781–1796, 2017.

[19] M. Rungger and M. Zamani, "SCOTS: A tool for the synthesis of symbolic controllers," in *Proceedings of the 19th international conference on hybrid systems: Computation and control*, 2016, pp. 99–104.

[20] D. S. Sundarsingh, J. Bhagiya, J. Chatrola, and P. Jagtap, "Autonomous exploration using ground robots with safety guarantees," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.

[21] M. Khaled and M. Zamani, "OmegaThreads: Symbolic controller design for ω-regular objectives," in *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*. New York, NY, USA: Association for Computing Machinery, 2021.

[22] P. Jagtap and M. Zamani, "QUEST: A tool for state-space quantization-free synthesis of symbolic controllers," in *Quantitative Evaluation of Systems*, N. Bertrand and L. Bortolussi, Eds. Cham: Springer International Publishing, 2017, pp. 309–313.

[23] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[24] 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)*, 2017, pp. 1396–1402.

[25] M. Khaled and M. Zamani, "pFaces: An acceleration ecosystem for symbolic control," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 252–257.