IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 11, 2021, accepted February 22, 2021, date of publication February 26, 2021, date of current version March 9, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3062662

Geometric Verification of Vector Fields for Autonomous Aircraft Navigation

GIOVANNI MIRAGLIA, (Member, IEEE), AND LOYD R. HOOK, (Member, IEEE)
Department of Electrical and Computer Engineering, The University of Tulsa, Tulsa, OK 74104, USA

Corresponding author: Giovanni Miraglia (giovmi@utulsa.edu)

This work was supported under NASA cooperative agreement number NND13AB04A.

ABSTRACT New generations of small advanced aircraft may soon transform how people and freight are transported. Thus, ensuring safe and reliable operation of these systems, in all plausible scenarios, is of utmost priority. In this paper, we address this problem by proposing a method to verify the safety of an autonomous aircraft controller following a navigational policy. The policy is encoded using discrete vector fields that are meant to drive the vehicle to a goal state and away from obstacles or restricted airspace. The solution presented in this paper provides the ability to verify the policy, catching unsafe scenarios which may be missed by random Monte Carlo methods or general reachability analysis. After illustrating the main theoretical principles, a possible practical implementation is described and compared with analysis based on Monte Carlo simulations. The comparison shows an insightful example where Monte Carlo simulations fail to detect several corner cases that are uncovered by the proposed method. In the conclusion of the paper, we provide insights about possible future research to implement the proposed solution obtaining higher accuracy and efficiency.

INDEX TERMS Navigation, unmanned aerial vehicles, autonomous agents, reachability analysis.

I. INTRODUCTION

Aircraft navigational flight planning typically consists of manually laying out long consecutive straight line segments encoded with vertices called waypoints. These segments are planned to avoid dangerous areas, such as obstacles or unapproved airspace. For increasingly autonomous airplanes, such as is being envisioned for Advanced Air Mobility (AAM), deviating from the planned path may require a complete replan. Without a replan, the aircraft risks flying into hazardous areas on a rejoin. In fact, these types of deviations may be relatively common due to a myriad of factors including weather, aircraft to aircraft conflict, or safety maneuvers. Therefore, for increasingly autonomous aircraft, a "complete plan" encoded in a "control policy" may be the best option.

For a complete plan, all of the potential states of an aircraft have an associated control policy. For instance, one popular method to encode this policy is with a feedback motion plan [1], [2], which in these cases, provides a calculated safe heading for the aircraft to capture for every possible two-dimensional location on a map (see Fig. 2). Having a pre-calculated complete plan also provides the ability to pre-verify that the plan is safe and suitable for the mission. Verification activities can include simulation or other analysis, but the intent to verify safe appropriate resultant trajectories remains, regardless of the approach.

Because of the very large number of possible aircraft states that need to be checked, a Monte Carlo approach may be used. However, because of the inherent random nature of the Monte Carlo approach, some unsafe resultant trajectories may be missed. Other methodologies, such as Hamilton-Jacobi (HJ) numerical reachability analysis [3], are more exhaustive, but their computational and space complexity is high and their accuracy is affected by both time and space discretization.

This paper proposes a new ''exhaustive'' geometric method of analysis and verification, which relies on the derivation of sets of trajectories and their intersection with unsafe areas. The proposed method is first compared to HJ reachability calculations for a small discretized cell containing a single commanded heading. Then, it is compared with Monte Carlo simulation results over large trajectories spanning multiple cells across the entire control policy. The results show that the generic HJ formulation can underestimate or overestimate the reachable set for a single cell. This is problematic because when integrated across the entire policy, these inaccuracies compound to obscure critical insights. Monte Carlo

The associate editor coordinating the review of this manuscript and approving it for publication was Seung-Hyun Kong.

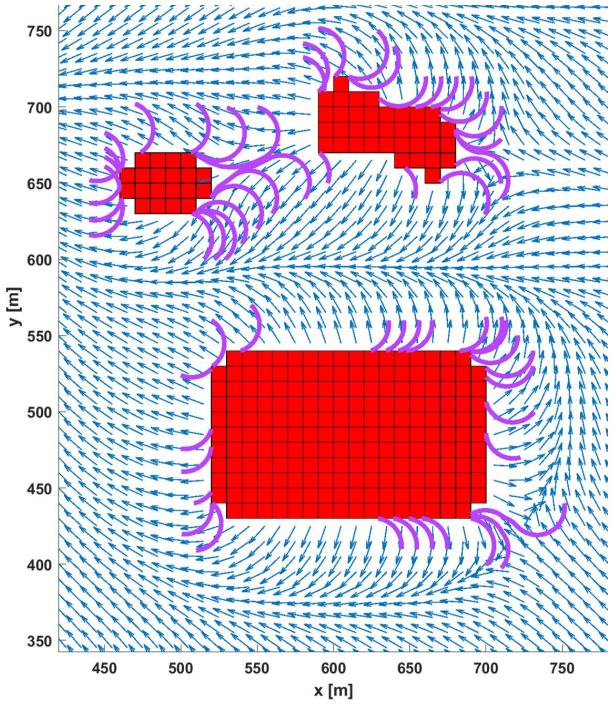

FIGURE 1. Examples of corner cases missed by Monte Carlo analysis for trajectories resulting from a navigational policy encoded with a vector field of commanded headings.

simulations are also found to miss unsafe "corner cases" that the proposed geometric method uncovers. Fig. 1 shows a set of these missed corner cases, where a resulting trajectory would intersect an obstacle causing an obvious safety concern. Uncovering these unsafe trajectories in verification allows them to be added to the set of states to be automatically avoided when the system is in operation.

II. BACKGROUND

A. AUTONOMOUS NAVIGATION

Autonomous navigation first requires the definition of goals, obstacles, and areas of restricted airspace. A navigational planner uses this information, along with vehicle maneuvering constraints, to generate a collision-free path that could be feasibly followed by the aircraft. Typically, aircraft constraints such as turn radius and climb rate are included in the evaluation of feasibility. After a path has been calculated, it must be encoded in a way that will allow controllers to direct the aircraft suitably along the path. Waypoint controllers calculate aircraft position deviation from a straight line connecting two waypoints and adjust the aircraft heading to reduce this deviation [4]. As the aircraft progresses, waypoints are successively achieved and the controller adjusts which waypoints are used to calculate the deviation.

Typically in nominal conditions, deviations from the path will remain relatively small and progress will be made toward the goal. However, in certain situations relatively large unplanned deviations may be necessary. In these cases, there may not be sufficient information encoded in the waypoints to enable the controller to safely rejoin the planned path.

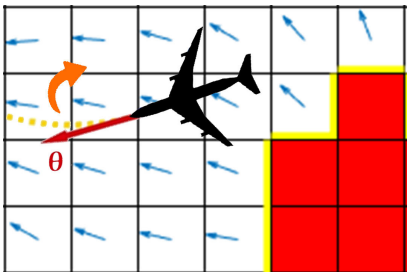

FIGURE 2. Example of the airplane following a feedback motion plan π encoded in a discrete vector field (red cells are part of the obstacle region O).

For instance, an unplanned maneuver to deconflict with an approaching aircraft may take our vehicle to a position where obstacles are between it and the planned path. In the absence of further information, the controller will attempt a normal rejoin and a collision may occur. Situations such as these require that a plan carry more information than a typical set of waypoints contain. Further, additional verification criteria must also consider resultant trajectories generated from any feasible vehicle state, not just those near the planned path.

An alternative to using waypoints is to use vector fields in a feedback motion plan [1]. The vector fields encode a *navigational policy*, which is an appropriate action for each possible location on a map. Using this method, obstacles and restricted zones can be accounted for regardless of their proximity to the nominal path. So in essence, an appropriate plan is calculated for every possible position on a map. The drawback of this approach is that it is difficult to calculate an *optimal* policy if vehicle dynamic constraints are considered. However as has been shown, a *suitable* policy is often easily calculated; with suitability being defined as a plan which, given appropriate initial conditions, always progresses toward the goal and avoids intersections with obstacles or restricted airspace.

B. POLICY VERIFICATION

Another advantage of developing a full navigational policy encoded using vector fields is that its suitability can be verified offline. In order to accomplish this, however, a model for the vehicle and controller which follows the plan must also be developed. In this paper, the vehicle is modeled kinematically assuming a constant forward velocity and a constant maximum turn rate, which taken together produce a minimum turn radius. The controller is modeled as a type of bang-bang controller, which has three possible control commands resulting in either minimum radius turns in each direction, left or right, or a straight trajectory. The controller produces the control commands to minimize the error between the vehicle's heading angle and the vector direction encoded at the vehicle's location in navigational policy as in Fig. 2.

A goal of this work is to allow both the policy and the controller to be verified together to ensure safe operation and progress toward the goal. This is critically important because if navigational planners and controllers are both autonomous and do not rely on human intervention, then they will not be

approved to fly without ensuring the safety of the system. So, autonomous systems which can fulfill planning and control requirements must be developed simultaneously with their methods of verification.

Two general strategies collectively make up the state of the art in the verification of navigational control policies. The first is known here as exhaustive verification. This method attempts to analyze every possible situation mathematically to determine the set of states and control inputs that may lead to critical end states. Typically, these critical end states are either unsafe states, such as those which end in a collision, or goal states that the vehicle is trying to achieve. Generally, exhaustive verification methods utilize reachability analysis, where techniques to determine whether a particular end state is *reachable* given the initial state and the control mechanism. However, exhaustive verification is oftentimes difficult or impossible to achieve; therefore, numerical methods are utilized to approximate the solution. Nevertheless, the main drawback is that often these methods have high computational and space complexity.

The second strategy involves the simulation of a random sampling of initial states to find a probabilistic distribution of end states. These methods are called *Monte Carlo* methods and they are typically used when an exhaustive analysis is too difficult or is computationally intractable. Although Monte Carlo can be very effective in determining the probability of certain events, sufficiently rare events are oftentimes missed by Monte Carlo and thus the method is not ideal to prove the safety of a system on its own.

For the verification of navigational policies encoded using discrete vector fields and the proposed aircraft controller model, precise techniques for exhaustive analysis have not been developed and collision scenarios can be made sufficiently rare as to be missed by Monte Carlo analysis. This work attempts to rectify this situation by developing quasi-exhaustive and precise verification techniques for navigational plans defined for use within a control system for an autonomous fixed-wing aircraft.

Specifically, the manuscript presents a novel analytic tool for the safety analysis of a navigational plan and control system built around discrete vector fields. First, the theoretical principles behind the method are summarized with related work in Section III and the analytical formulation of the problem in Section IV. The paper continues in Section V with a description of the method and in Section VI with details on a proposed implementation, pointing out its flaws. Section VII details results comparing our method with analysis based on Monte Carlo simulations showing its ability to detect corner cases that a probabilistic analysis can easily miss. Finally, in Section VIII we offer conclusions along with possible future directions for the work.

III. RELATED WORK

A. AIRCRAFT GUIDANCE, NAVIGATION, AND CONTROL

High-level control of an aircraft is generally thought to encompass three related, but distinct, components which are oftentimes grouped as guidance, navigation, and control (GNC). Guidance refers to the determination of the desired path needed to achieve a certain goal state, navigation refers to the determination of the orientation and velocity state required to follow that path, and control refers to the low-level manipulation of forces and aircraft moments required to achieve the orientation and velocity states. In [5], the authors provide an overview of the methodologies used for the guidance, navigation, and control of fixed-wing UAVs. They consider fixed-wing UAVs that generate their forward speed using a propulsion system such as propellers driven by electric motors or gas engines. A more detailed exploration of path following for fixed-wing aircraft can be found in [6]. Here, the authors provide control solutions based on theoretical stability and convergence analyses, which can be applied to almost all regular 3D paths. The proposed solutions are validated using hardware-in-the-loop simulations. Encoding a GNC solution with a vector field, similar to the one analyzed in this work, can be found in [7]. Specifically, the authors present a Gradient vector field (GVF) path following and circular obstacle avoidance that allows a fixed-wing UAV to avoid detected obstacles without the need to replan the whole flight path.

Verification of proposed GNC solutions is also presented in many related works with approaches ranging from simulation to more formal mathematically derived verification methods. For example in [8], real-time hardware-in-the-loop simulations are utilized to validate a hierarchical path planning and control algorithm for a small fixed-wing UAV. Also in [9], the authors present a safety framework for the online verification of the safety of each planned trajectory. The proposed solution uses formal methods to handle uncertainty and disturbances. The framework can be integrated into existing motion planning architectures and enables the fail-safe operation of self-driving vehicles. In [10], the authors use reachability analysis to compute the reachable sets of each motion primitive and subsequently to define the satisfaction relation of motion primitives with formulae in linear temporal logic.

Reachability analysis, specifically, is of interest in the formulation of this paper as we try to answer the question: given a feedback motion plan, which region of the space is unsafe? The state of the art of numerical analysis for this type of question is based on the use of reachability.

B. REACHABILITY ANALYSIS

Reachability can either be analyzed in the forward time direction or in the backward time direction. In forward reachability, a set of initial states is propagated forward for a specific time interval, finding the total set of possible future states which can be reached in that time horizon. Conversely, in the case of backward reachability, starting from a set of final states, the computation is performed backward for a specific time interval to find the set of possible initial states that could have lead to the final states. In a safety analysis performed using backward reachability, we start with states

inside an unsafe region and simulate the evolution of the states backward in time. If the set of states reached intersects with the set of possible initial states, then the analysis flags a potential safety hazard. In Fig. 3 one can notice that for a safe system, the intersection between the backward reachable set of the unsafe region and the set of possible initial states is empty.

In [11], the author defines and examines eight types of forward and backward reachability constructs. It is shown that forward and backward algorithms can be interchanged if well-posed backward trajectories can be defined. Furthermore, it is demonstrated that backward reachable tubes are the most broadly applicable formulation of reachability for analyzing system safety. Finally, a key insight from the analysis conducted in this paper is that backward reachability formulation is more likely to suffer from numerical stability problems.

In [12], a variety of methods for the reachability analysis of continuous and hybrid systems are presented, focusing on the following topics: geometrical objects for representing sets, approximation schemes, and combinations of graph-search algorithm and partition refinement. The work in [13] presents results of using methods to under approximate Reach-Avoid sets for navigating in space around obstacles with linear time-invariant dynamics. The work in [14] focuses on under-approximated boundaries for the backward reachable set of a simply connected target region computed using linear programming approaches. In [15], the authors use Lagrangian methods to compute forward reachable sets to determine the region of attraction around an equilibrium point.

1) GEOMETRIC REACHABILITY METHODS

In [16], systems with mixed state and input constraints are considered. After formulating a reachability controller synthesis problem for this class of systems, the authors show how standard geometric tools can address the problem for the special case where disturbance constraints are independent of the state and control. The work in [17] focuses on the reachability problem for environments with dynamic geometry. Geometric constructions based on constraints with one variant parameter are considered. The proposed solution finds a continuous path between two geometric configurations if one exists. In [18], the kinodynamic multi-robot planning problem in cluttered 3-D workspaces for a quad-rotor is considered. Offline reachability analysis on position invariant geometric trees is leveraged to find a collision-free geometric solution that is kinodynamically feasible for the multi-robot team.

C. SAFETY ANALYSIS BASED ON REACHABILITY

Reachability analysis, as a tool, has several uses in the verification of GNC systems. Specifically relevant to this work is the use of reachability to verify the safety of the system and the resultant trajectories which are an output of our approach. This particular use has a broad history and several safety analyses based on reachability have been proposed in the past.

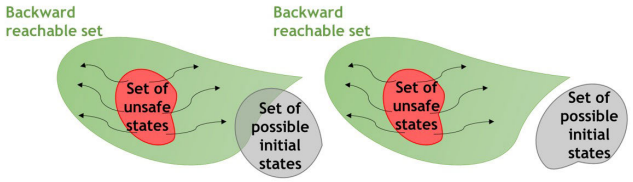

FIGURE 3. Example of an unsafe system (left) and a safe system (right).

For instance, the work in [19] addresses formal verification of automotive collision avoidance systems. Here the authors present a framework that can be used to formally verify that a collision avoidance system makes correct decisions that are robust to measurement errors. In [20], the authors address the online verification of partial motion plans for aerial robots; accounting for uncertainty in both the environment's and the robot's state. The solution proposed in this paper consists in generating robust control invariant sets based on loiter circles, where the aerial robot follows a circular pattern. The invariant sets are generated exploiting techniques from reachability analysis and considering an infinite time horizon, unlike most of the other methods based on reachability. In [21], invariant safe sets are computed for self-driving automobiles in linear time and with provable formal safety guarantees.

The Hamilton-Jacobi (HJ) formulation of reachability is one of the most general formulations that is applied to nonlinear systems in the presence of disturbances [3], [22], [23]. The drawback of the HJ formulation is that the computational complexity is exponential with respect to the dimension of the continuous state. For this reason, various decomposition methods [24], [25] for specific classes of problems have been proposed to achieve dimensionality reduction.

D. SAFETY ANALYSIS BASED ON MONTE CARLO SIMULATIONS

A different approach for the analysis of hybrid and switched systems is to use Monte Carlo simulations [26], [27]. Monte Carlo simulations are an effective tool to analyze complex nonlinear systems having high dimensional state space that are intractable using other methods. Consequentially, Monte Carlo simulations are a widely used validation tool. For instance in [19], Monte Carlo simulations are used to examine a 6-DOF model of an autonomous underwater vehicle. Simulations are used to analyze the effects of model uncertainties and to tune the guidance algorithm effectively. Another example of validation based on Monte Carlo simulations can be found in [28], where aircraft 4-D trajectories are constrained inside specific temporal and spatial boundaries, referred to as target windows (TWs). In this case, the authors propose evaluation methods based on both Monte Carlo analysis and reachability theory. Reachability theory is used to characterize the aircraft maneuvering freedom and perform conflict resolution. The proposed methodology was then validated using Monte Carlo simulations, which allowed the probability of meeting TW constraints and the likelihood of conflict to be estimated.

Validation based on Monte Carlo simulations is much more general and simpler than other methods. Nevertheless, the shortcoming in this type of testing is that this is a probabilistic approach in which a specific trajectory is tested with a certain probability. This means that there is a chance to miss corner cases in the analysis. For the reader interested in more details about using Monte Carlo simulations to simulate rare events, [27] provides a general reference that focuses on the theory of "Importance Sampling" and "Splitting" techniques as well as several applications.

IV. PROBLEM FORMULATION

The verification method presented in this paper is based on the concept of reachability. However, rather than using a generic formulation such as the Hamilton-Jacobi, specific functions are derived exploiting geometric principles [29], [30]. The method is time-independent and allows for the analysis of the whole 3-D configuration space by propagating 2-D sets. The problem is formulated below to support this analysis.

Given a workspace $W \in \mathbb{R}^2$ represented with a grid-map having resolution d and an obstacle region O, our objective is to categorize each cell of the grid map as either safe or unsafe according to the following definition:

Definition 1: A cell is safe if there does not exist a configuration from which the obstacle region is reachable with the given plan π and vehicle's controller and kinematic model.

Our formulation relies on the assumption of a discrete feedback motion plan having the following characteristics:

- The plan indicates the desired heading direction, (called the commanded heading θ_c,) the vehicle should travel at each free grid-map position.
- The resolution of the grid-map d is smaller than the vehicle's minimum turning radius r.
- Bang-bang control is used to correct the error between the vehicle's actual heading θ and the commanded heading θ_c. Therefore, the vehicle can either travel straight with speed v or turn with its minimum turning radius $r = \frac{v}{\omega}$ (ω is the turn rate).

Under these assumptions, in each cell, there are five possible trajectories. If an initial angle θ_0 is equal to the commanded angle θ_c the vehicle travels in a straight line. Instead, if $\theta_0 \neq \theta_c$, the trajectory begins a turn in the direction that minimizes the angular distance required to reach the commanded heading. When θ_c is achieved, the trajectory continues in a straight line from that point. The last case is when the vehicle does not converge to the desired angle before leaving the cell. In this last case, the exit angle is different from θ_c, and the path consists of an arc without a straight line segment. In summary, three types of paths are possible:

- straight line (S path);
- turn followed by a straight line segment (CS path);
- turn only (C path);

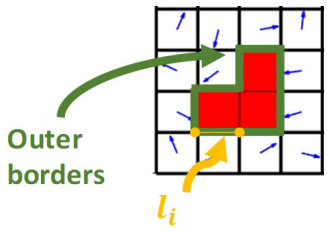

FIGURE 4. Example of outer borders of obstacle region *O*.

S stands for *straight line* while C stands for *circular motion*. Since there are two types of circular motion, the complete set of possible paths is: $\{S, R, L, RS, LS\}$, where R and L stand for right and left respectively.

The configuration space is $C = W \times \mathbb{S}^1$, therefore a configuration $q \in C$ is the tuple (x, y, θ). Similarly we can define $C_O = O \times \mathbb{S}^1$, which is the obstacle region in the configuration space.

Section III-B introduced the concepts of the Backward Reachable Set (BRS) and Forward Reachable Set (FRS). Related concepts are the Backward Reachable Tube (BRT) and the Forward Reachable Tube (FRT). The difference is that the BRS and FRS are sets of configurations for a specific time t^*, whereas BRT and FRT are sets of configurations for a time interval $[0, t^*]$ [11].

In our case, we define the BRT $\mathcal{B}(C_O, r, \pi)$ of C_O as the set of all the configurations from which C_O is reachable with a given plan π and a minimum turning radius r. A configuration q_0 is in $\mathcal{B}(C_O, r, \pi)$, if starting from q_0 and following the feedback motion plan π, there exists a time t_* in which the vehicle's configuration q_* is in C_O. It is important to notice that $\mathcal{B}(C_O, r, \pi)$ does not show any time dependency. The reason is that we consider an infinite time horizon. In other words, we are not interested in the time necessary to reach the target set; instead, we just want to know if the vehicle will reach it or not.

What stated above can be summarized as follows:

$$\mathcal{B}(C_O, r, \pi) = \{q_0 \in C | \exists t_* \in [0, \infty) \text{ s. t. } q(t_*) \in C_O\} \tag{1}$$

A set of adjacent cells can be represented using only the external borders, as illustrated in Fig. 4. A generic outer border is l_i, while the set of all the outer borders is $L = \bigcup_{i=1}^{n} l_i$. It is possible to reformulate the definition for BRT by observing that C_O is reachable from q_0 if the trajectory starting from q_0 intersects L. Exploiting this observation, the BRT can be defined as follows:

$$\mathcal{B}(C_O, r, \pi) = \{q_0 \in C | \exists t_* \in [0, \infty) \text{ s.t. } q(t_*) \in L = \bigcup_{i=1}^{n} l_i\} \tag{2}$$

This second formulation allows for the representation of C_O, which is a 3-D subspace, with the union of the 2-D sets $l_i \times \mathbb{S}^1$ (coordinate and heading angles at the borders).

V. GEOMETRIC ANALYSIS OF THE PROBLEM

This section summarizes the main geometric properties used to derive the piece-wise functions utilized in our analysis. Recalling that in a grid-map representation, the obstacle region O can be characterized using only the outer borders l_i, we start by illustrating how to compute the boundaries of the set containing every configuration $q = (x, y, \theta)$ of a cell from which a specific border $l_i \times \mathbb{S}^1$ is reached. We define this set as *Cellular Backward Reachable Tube* $\mathcal{B}(l_i, r, \theta_c)$. The boundaries of $\mathcal{B}(l_i, r, \theta_c)$ are given by the functions $\Theta_{\min}(x, y)$ and $\Theta_{\max}(x, y)$, where the dependency from r and θ_c is omitted because they are both constant in a cell for a given plan. From $\Theta_{\min}(x, y)$ and $\Theta_{\max}(x, y)$ it is possible to fix one of the coordinates and obtain the functions $\Theta_{\min}(x)$ and $\Theta_{\max}(x)$ that are specific for a border. For instance we can consider the top border of a cell as the target border and the left border as the starting border. In this case, the functions $\Theta^T_{L_{\min}}(x)$ and $\Theta^T_{L_{\max}}(x)$ delimit the range of heading angles for each position along the left border from which the top border is reachable. In other words, $\Theta^T_{L_{\min}}(x)$ and $\Theta^T_{L_{\max}}(x)$ are the boundaries of $\mathcal{B}(T, r, \theta_c)$ along the left border.

The opposite problem is the derivation of the functions $\Phi^T_{L_{\min}}(x)$ and $\Phi^T_{L_{\max}}(x)$ that are used to compute the boundaries of the *Forward Reachable Tube* $\mathcal{F}$ of the left border in the top border. In this case, the objective is to compute the set of configurations in the top border that are reached by trajectories starting from the left border. It is important to notice that the set delimited by $\Phi^T_{L_{\min}}(x)$ and $\Phi^T_{L_{\max}}(x)$ is the forward mapping of the set delimited by $\Theta^T_{L_{\min}}(x)$ and $\Theta^T_{L_{\max}}(x)$.

In the method proposed in this paper, the safety analysis of a feedback motion plan is performed by propagating backward the intersection between backward and forward reachable tubes in a border. For instance, for a border l_*, we can find the intersection set $\mathcal{I}$ between the set $\mathcal{F}(l_i)$ delimited by $\Phi^{l_*}_{l_{i\min}}(x)/\Phi^{l_*}_{l_{i\max}}(x)$ and the set $\mathcal{B}(l_j)$ delimited by $\Theta^{l_*}_{l_{j\min}}(x)/\Theta^{l_*}_{l_{j\max}}(x)$. The elements of $\mathcal{I}$ lie on trajectories that start from l_j and end in l_i.

The intersection sets are propagated from one border toward the neighboring borders starting from the outer borders of the obstacle region O. The sets being propagated consist of configurations from which the obstacle region is reachable. Once the algorithm converges, it labels cells of the grid-map as safe if in every border the BRT is an empty set. On the other hand, a cell is labeled as *unsafe* if any of the four borders has at least one configuration from which the obstacle region is reachable.

In the following subsections, we start by introducing the geometric principles exploited to derive the functions Θ and Φ mentioned above. For the sake of brevity, we illustrate these principles at a very high level. For the detailed derivation of the functions, the reader can refer to [29] and [30]. The analysis starts with the computation of the BRT from a chosen border to all points internal to a single cell, which then is used as a benchmark to compare with the well-established method based on the Hamilton-Jacobi formulation. After the illustration and comparison of the BRT for all points inside the cell, we simplify the problem considerably by exploiting the fact that, by definition, there are no safety concerns internal to a free cell. This allows us to only consider the intersection of the reachable tubes at the borders of the cells, allowing calculation of forward and backward *border-to-border mapping*. Finally, we present the core idea behind the proposed method, which consists in finding and propagating the forward and backward border-to-border intersections between cells throughout the entire map.

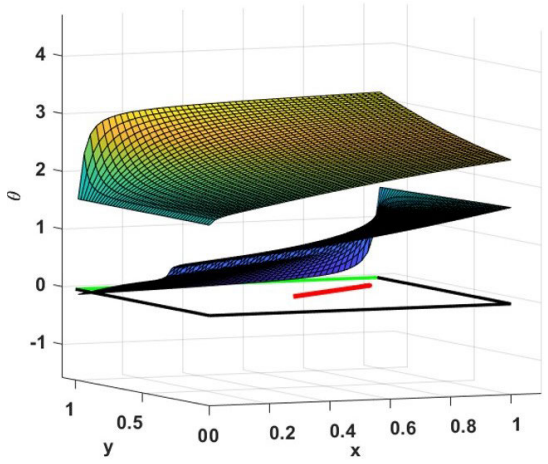

FIGURE 5. Boundaries of backward reachable tube of top border.

A. CELLULAR BACKWARD REACHABILITY

In [29], a full description of the calculations and processes required to calculate the BRT from a chosen exit border was provided. These functions allow calculations of minimum and maximum surfaces, $\Theta_{\min}(x, y, \theta_c)$ and $\Theta_{\max}(x, y, \theta_c)$, which bound the reachable tube. In Fig. 5, an example of the cellular BRT computed for the top border is shown. The example cell has a horizontal width of 1 in each direction and a commanded heading angle of $\theta_c = \frac{\pi}{2}$. The horizontal axes of the figure correspond to the x and y coordinates, while the vertical axis indicates a vehicle orientation, in radians, wrapped so that $-\pi < \theta \leq \pi$. The green line indicates the top border of the cell corresponding to the line $y = 1$ between $0 < x \leq 1$. The two surfaces shown in the figure are the functions $\Theta_{\min}$ and $\Theta_{\max}$. These surfaces span the entire horizontal extent of the cell.

All the possible vehicle configurations between these two surfaces, $\Theta_{\min} < \theta(x, y) \leq \Theta_{\max}$, will inevitably exit the top border. More strongly, the top border is reachable if and only if the vehicle's configuration is bound by the functions. The same process is utilized to derive the BRT for the other borders as well. It is based on the same geometric principles, therefore we can exploit the same functions used for the top border and apply proper transformations (i.e. rotation and translation).

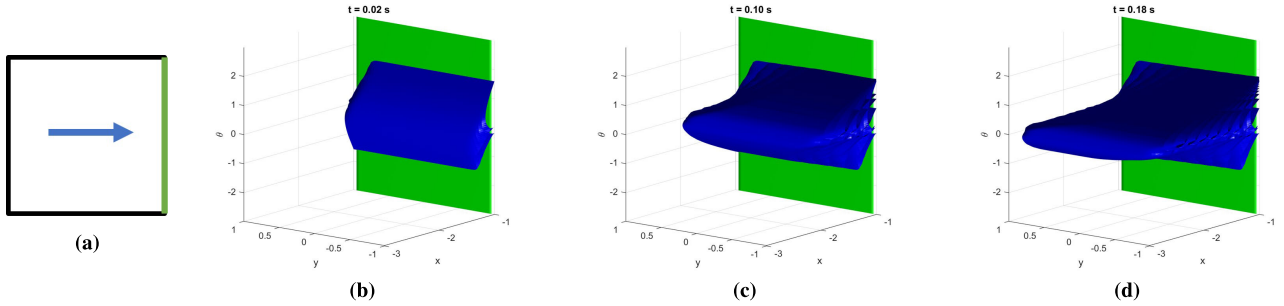

FIGURE 6. An example of the evolution of the boundaries of the backward reachable set for the Bang-bang controller described in Section IV. Time step of 0.02s, spatial resolution of 0.12m, angle resolution of 0.36°. The vehicle was traveling with a speed of 10m/s and a minimum turning radius of 5m. In (a) there is the cell with commanded angle of 0°. In this example the target set is the right border.

1) COMPARISON WITH HAMILTON-JACOBI FORMULATION

In section III, it was mentioned that one of the most popular and rigorous methods for reachability analysis is based on the use of the Hamilton-Jacobi formulation. This tool is very powerful and general. It can compute reachable sets for non-linear dynamic systems accounting also for disturbances. The data illustrated in this subsection were obtained using the implementation available in [31], which to the best of our knowledge is the state of the art of Hamilton-Jacobi reachability analysis. Even though this tool is very powerful, it is a numerical method and as such, it is affected by discretization error. More specifically, in this case, there are two sources of error, the discretization of the configuration space and the time step used to grow the boundaries of the reachable set.

In the example considered in this subsection, the target set was the right border (Fig. 6a) and the objective was to find the boundaries of $\mathcal{B}(R, r, \theta_c)$ for an infinite time horizon. Fig. 6 illustrates the growth of the backward reachable set using the tool available in [31]. In Fig. 7a, there is the complete BRT obtained using the HJ analysis. In this case, the boundaries are highlighted in orange. Instead, in Fig. 7b, there are the boundaries of BRT computed using our analytical formulation, which was derived in [29] and illustrated in the previous section. Fig. 8 shows the error affecting the HJ analysis. Fig. 8b illustrates the error for the upper boundary, while Fig. 8a illustrates the error for the lower boundary. In both figures, the error is color-coded. Red indicates an underapproximation of the reachable tube, while green an overapproximation.

This example shows that even the state of the art of numerical reachability analysis can underestimate the backward reachable tube in some regions. For the specific class of problems considered in this example, the advantages of deriving closed-form solutions for the boundaries of the reachable tube are two:

- constant computation complexity (i.e. for a given position we can compute minimum and maximum in constant time);
- no error due to time or space discretization (we cannot eliminate the rounding error that affects every digital system).

B. BORDER-TO-BORDER BACKWARD MAPPING

As mentioned in the previous section, from the generic functions for the boundaries of the BRT, it is possible to derive the ones specific for each border by fixing one of the coordinates. We define the set delimited by these functions as *border-to-border* backward reachable tube. For instance $\Theta^T_{B_{\min}}(x, \theta_c, r)$ and $\Theta^T_{B_{\max}}(x, \theta_c, r)$ delimit the BRT of the top border (T) in the bottom border (B), which means that every configuration of the bottom border that is in this set has a trajectory ending in the top border.

C. BORDER-TO-BORDER FORWARD MAPPING

The *border-to-border* BRT allows us to determine the set of entering heading angles required for a trajectory to enter the cell at one border and exit another. The opposite problem finds the exit heading angles for trajectories entering at another border. The solution to this problem is similar to the solution for the BRT. We start considering circles intersecting the corners of the starting border and any point of the target border. Fig. 9 illustrates examples of forward mapping from the bottom border to the top border. It is important to notice that only trajectories that do not exit the cell before reaching the top border must be considered. Furthermore, as shown in Fig. 9b we have to account also for cases in which multiple trajectories converge to the same end configuration. Accounting for these constraints and using the same geometric principles exploited for the backward reachability, it is possible to derive the analytic functions $\Phi^B_{\mathrm{A}\min}$ and $\Phi^B_{\mathrm{A}\max}$, which provide the boundaries of the FRT in a border B for trajectories starting from a border A. Like the functions for the backward reachability, the detailed derivation can be found in [29], [30].

D. FORWARD AND INVERSE MAPPING

To better illustrate the relationship between forward and backward reachable tubes, we refer to the example shown in Fig. 10, where trajectories starting from the bottom border and ending in the top border are considered. The forward mapping from $\mathcal{B}$ to $\mathcal{F}$ is computed with the functions $p(x, \theta_c)$ and $a(x, \theta_c)$, which provide the final location and orientation

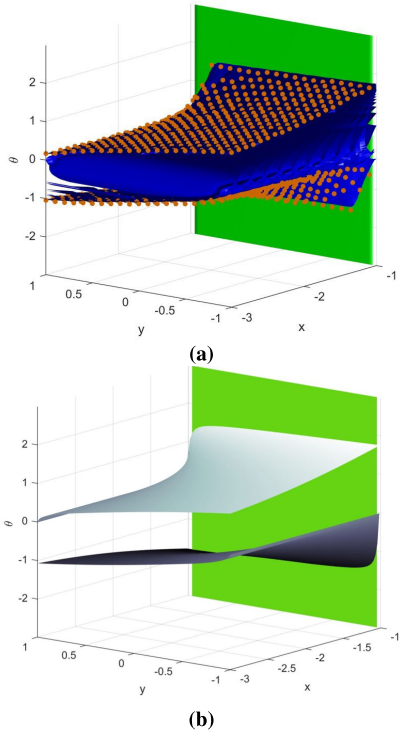

FIGURE 7. Boundaries of backward reachable tube obtained with Hamilton-Jacobi Formulation (a) and using functions $\Theta(x, y)$ (b).

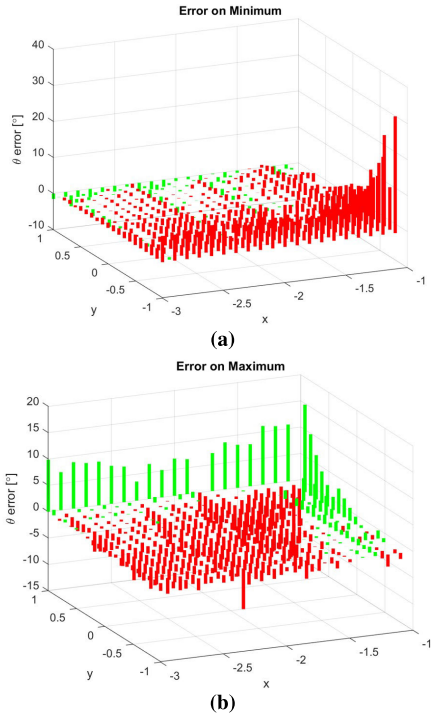

FIGURE 8. Error in backward reachable tube obtained with Hamilton-Jacobi analysis. Green indicates overapproximation, while red indicates underapproximation.

respectively.[1] In this particular example, every trajectory starting from the bottom border and ending in the top border

[1]The minimum turning radius r and the cell size d are constant, therefore they are not included in the parameters of the functions.

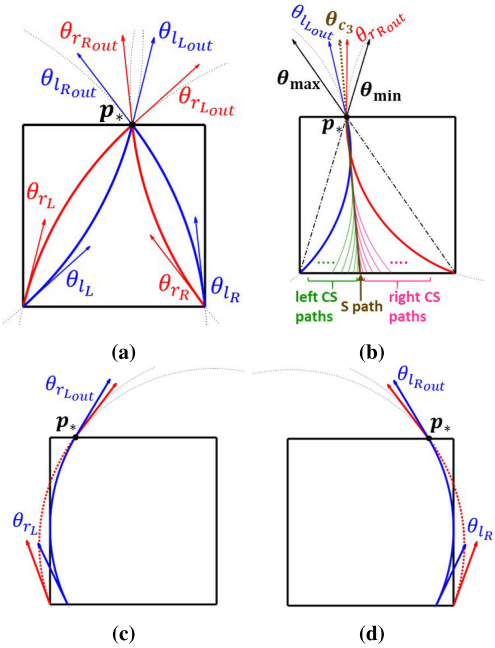

FIGURE 9. Examples of forward mapping from bottom to top border. (a) Examples of minimum and maximum angles associated with right and left turns starting from edges of the bottom border. (b) Examples of trajectories ending in the same configuration of the top border.(c) Example of right turn "limited" by the left border. (d) Example of left-turn "limited" by right border.

consists in a right turn without a straight line portion. In this case, with a right-handed coordinate system having center in the bottom left corner, the forward mapping is performed using the following functions:

$$p(x,\theta) = x + r\cos\left(\theta - \frac{\pi}{2}\right) - \sqrt{r^2 - \left(d - r\sin\left(\theta - \frac{\pi}{2}\right)\right)^2} \tag{3}$$

$$a(x,\theta) = \arctan2\left(d - r\sin\left(\theta - \frac{\pi}{2}\right), -\sqrt{r^2 - \left(d - r\sin\left(\theta - \frac{\pi}{2}\right)\right)^2}\right) - \frac{\pi}{2} \tag{4}$$

The inverse functions that map configurations of the top border backward to the bottom border can be written in a similar fashion:

$$p^{-1}(x,\theta) = x + r\cos\left(\theta - \frac{\pi}{2}\right) - \sqrt{r^2 - \left(d - r\sin\left(\theta - \frac{\pi}{2}\right)\right)^2} \tag{5}$$

$$a^{-1}(x,\theta) = \arctan2\left(d + r\sin\left(\theta - \frac{\pi}{2}\right), \sqrt{r^2 - \left(d - r\sin\left(\theta - \frac{\pi}{2}\right)\right)^2}\right) + \frac{\pi}{2} \tag{6}$$

Fig. 10 illustrates how two generic subsets of the top border are mapped backward to the bottom border using the inverse mapping. It is clear that a configuration can be mapped

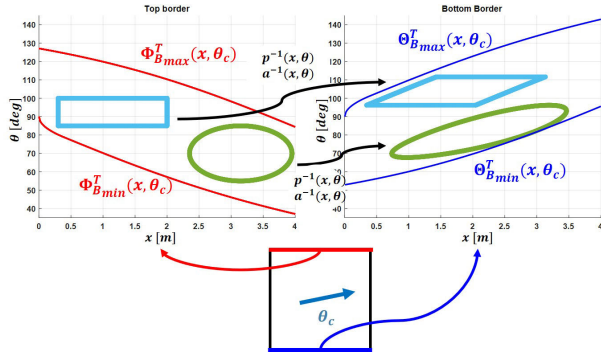

FIGURE 10. Inverse mapping for right turns.

backward only if it is in the set $\mathcal{F}(B)$ whose boundaries are computed using the function Φ. On the other hand, the forward mapping of a configuration of the bottom border is in the top border only if it is part of the set $\mathcal{B}(T)$, whose boundaries are computed with the function Θ. In general, for a generic set S in a border b_i, only the intersection between S and the set $\mathcal{F}(b_j)$ can be mapped backward to the border b_j. This property is the essence of the method proposed in this manuscript. In particular, starting from the borders of the obstacle region, in which every configuration is considered part of the BRT $\mathcal{B}(O)$, the method consists in mapping backward the intersections between $\mathcal{B}(O)$ and the forward reachable tubes of the neighboring borders. The process continues until only empty sets are being propagated, which means that the algorithm converged to the solution.

E. INTERSECTIONS BETWEEN SETS

In the previous subsection, the inverse mapping of sets entirely contained in the forward reachable tube was illustrated. Here, we consider the inverse mapping of the intersection between forward and backward reachable tubes, which is the basic operation performed in our method to compute the backward reachable tube $\mathcal{B}(O)$ for the borders of every cell in the grid-map.

An internal border is shared between two cells; therefore, it is possible to consider two directions. For instance, for a horizontal border, it is possible to consider trajectories that go from the bottom cell toward the top cell (upward direction) or trajectories that go from the top cell toward the bottom one (downward direction). Fig. 11 illustrates an example where the upward direction is considered. On the left side, there is the workspace, which consists of two free cells and one occupied by the obstacle region. On the right side, there are three sets for each figure. These sets are relative to the border shared between the two free cells. The sets in Fig. 11a are the backward reachable sets of the other three borders (i.e. top, left, and right) of the upper cell. Instead, in Fig. 11b, the sets are the forward reachable sets of the other three borders of the bottom cell. Fig. 12 shows the overlapping of the sets illustrated in Fig. 11a and Fig. 11b. Each intersection represents all the trajectories starting from a specific border of the bottom cell and ending in a specific border of the top

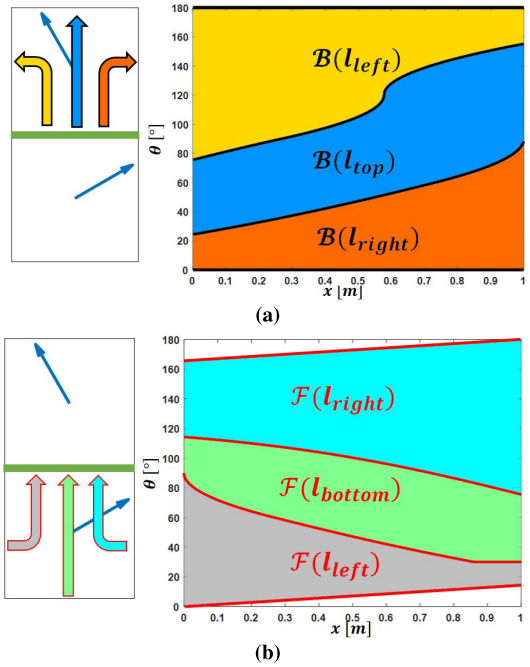

FIGURE 11. (a) Backward reachable tube in the bottom border of the upper cell. (b) Forward reachable tube in the top border of the bottom cell.

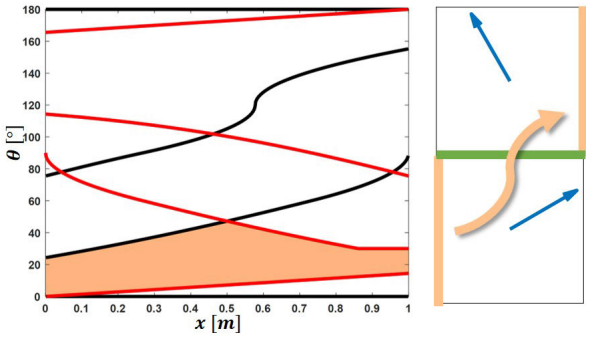

FIGURE 12. Intersections of forward and backward reachable tubes.

cell. For instance, the highlighted intersection set refers to all the trajectories starting from the left border of the lower cell and ending in the right border in the upper cell.

F. BACKWARD MAPPING OF INTERSECTIONS

The example in Fig. 12 shows that a 2-D set of a border is associated to a group of trajectories (and therefore 3-D configurations). Using a naive expression, we can say that a 2-D set in a border is a compressed representation of a 3-D set of configurations that are inside the cell. This is the key idea exploited in the safety analysis proposed in this paper.

According to Definition 1 a cell is labeled as safe if for all of its configurations the corresponding trajectory does not reach the obstacle region. Going back to the example in Fig. 12, it is clear that we do not need to test every 3-D configuration in the cell. Instead, we just need to test the configurations at the borders. Therefore, we can rephrase the definition of a safe cell as follows.

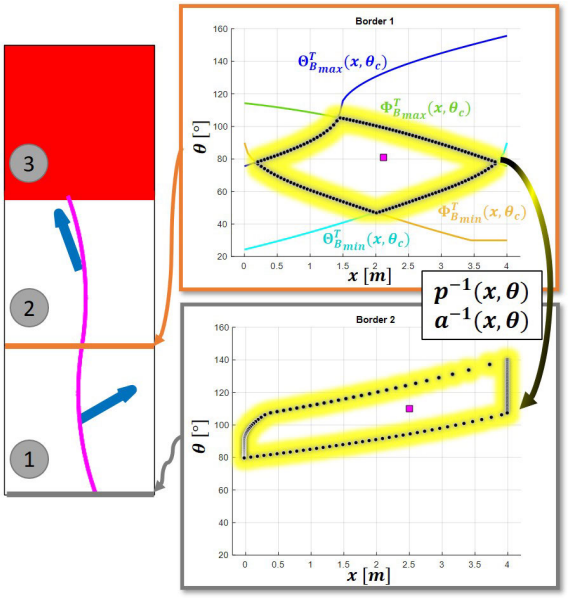

FIGURE 13. Backward reachability in a corridor.

Definition 2: A cell is safe if in every border b_i the backward reachable tube of the obstacle region $\mathcal{B}(O)_{b_i}$ is an empty set.

The meaning of this second definition is straightforward. If $\mathcal{B}(O)_{b_i}$ is an empty set for all the four borders, there is no configuration in that cell from which the obstacle region is reachable. Consequentially, we can determine the safety of the whole plan by computing the backward reachable tube of the obstacle region only for the borders of the grid-map. This computation is performed starting from the outer borders of the obstacle region. For each border the intersections between $\mathcal{B}(O)_{b_i}$ and $\mathcal{F}(b_j)$ are mapped backward to the neighboring border b_j.

This process is illustrated using the example in Fig. 13, where the top cell is part of the obstacle region. In this scenario, the objective is to compute the backward reachable tube of the red cell at the beginning of the corridor. Therefore, we want to compute all the configurations in the bottom border of the lowest cell having a trajectory ending in the red cell.

At the border shared between cell 3 and cell 2, every configuration is considered part of the BRT because this is a border of the obstacle region. Therefore, for this border, the BRT is $\mathcal{B}(O)_{b_{2_{\text{top}}}} = [0, 2\pi] \times [0, d]$. The first step is performed in cell 2 and it consists in finding the inverse mapping of every configuration of $\mathcal{B}(O)_{b_{2_{\text{top}}}}$ that is reachable from the bottom border $b_{2_{\text{bottom}}}$. Recalling that every configuration of the top border is part of the BRT $\mathcal{B}(O)_{b_{2_{\text{top}}}}$, it is clear that in the bottom border of cell 2 we have $\mathcal{B}(O)_{b_{2_{\text{bottom}}}} = \mathcal{B}(O) \cap b_{2_{\text{bottom}}} = \mathcal{B}(O)_{b_{2_{\text{bottom}}}}$, which means that in this border the subset of $\mathcal{B}(O)$ consists of all the trajectories that can reach the top border. The second step consists in finding out which configurations of $\mathcal{B}(O)_{b_{2_{\text{bottom}}}}$ can be reached from the bottom border of cell 1. Therefore we must compute the forward reachable tube $\mathcal{F}(b_{1_{\text{bottom}}})$ in $b_{1_{\text{top}}} \equiv b_{2_{\text{bottom}}}$. Then we must find the intersection set $\mathcal{I} = \mathcal{F}(b_{1_{\text{bottom}}}) \cap \mathcal{B}(O)_{b_{2_{\text{bottom}}}}$ and map its elements backward to the bottom border of cell 1, obtaining the set we were looking for.

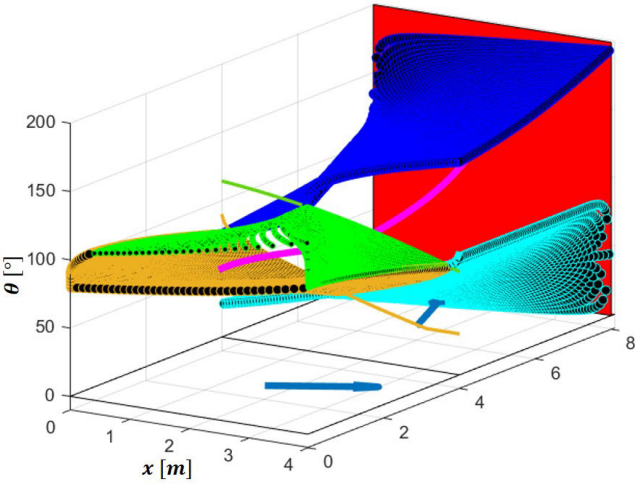

FIGURE 14. 3-D view of backward reachability in a corridor.

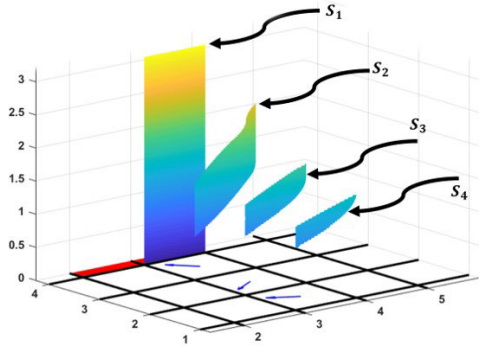

FIGURE 15. Example of inverse mapping across multiple borders.

On the right side of Fig. 13, the top figure shows the intersection between $\mathcal{F}(b_{1_{\text{bottom}}})$ (green and yellow curves) and $\mathcal{B}(b_{2_{\text{top}}})$ (light and dark blue lines), while the bottom figure shows the inverse mapping of $\mathcal{I}$. The configurations inside the two sets are part of the trajectory shown on the left.

Fig. 14 provides a 3-D view of the example illustrated in Fig. 13, where it is possible to see how the intersection set maps backward. Furthermore, it is also possible to see the evolution of the trajectory.

While Fig. 13 shows the propagation across just one border, Fig. 15 illustrates a more general example in which the propagation of $\mathcal{B}(O)$ is performed across multiple borders.

VI. ITERATIVE INVERSE MAPPING

In the simple example depicted in Fig. 13, it was possible to find a closed-form solution for the inverse mapping of the intersection set $\mathcal{I}$. That was possible because with the given commanded direction, for every configuration of $\mathcal{I}$, the trajectory was a full turn without a straight line segment. In this case, the mapping is bijective, which means that there exist the inverse functions a^{-1} and p^{-1} that we can use to perform the inverse mapping of the intersection set $\mathcal{I}$. Nevertheless, this is not always possible. In fact, for configurations reached

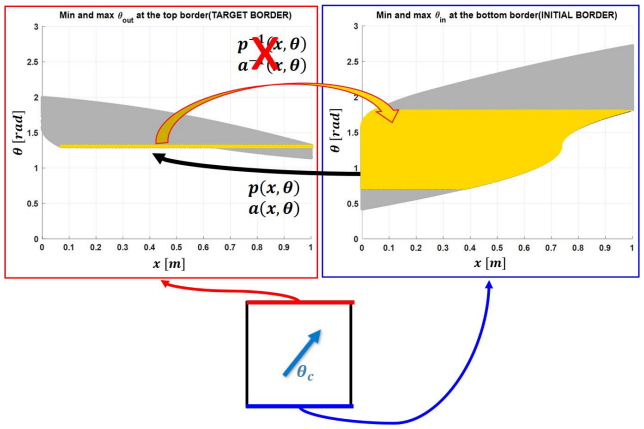

FIGURE 16. Example of inverse mapping in presence of CS paths. In the gray area there are configurations having trajectories consisting in C paths (bijective mapping), while configurations in the yellow region have CS paths (non bijective mapping).

by multiple CS paths, the mapping is not bijective. This is clarified by Fig. 16, where it is possible to see that for CS paths the inverse mapping of a line segment is a 2-D region. In these cases, an algorithmic solution is necessary to find the boundaries of the inverse mapping of $\mathcal{I}$.

Another challenge for the practical implementation of the method formulated in this paper is that an effective approach must be defined to keep track of all the subsets for each border of the grid-map. For instance, a possible solution could be to keep track of the actual functions along with the intersection points. However, the scalability of this approach is limited. A different method could be to keep track of polygons that approximate the shapes of the sets. In this case, the main challenge is to find the best approximation for the boundaries of the sets. Another challenge is to define how the propagation of the sets is performed. For instance, it is possible to use a *Depth First* approach to propagate the sets. However, it is necessary to define criteria for the termination of the propagation that account for the presence of closed-loop trajectories. These research questions are still open and they are left for future work. In the following subsection, we present a simple implementation that overcomes the abovementioned challenges.

The purpose of the implementation described below is to show the benefits of this approach, while effective solutions for the aforementioned challenges are still being investigated. This implementation consists in the iterative propagation of sets discretized using grid-maps. Details about the effects of the discretization error introduced by the use of grid-maps are illustrated in [30]. Although this solution does not fully exploit the benefits of the proposed method, it allows us to prove via experimental analysis that this approach is much more reliable than the simple Monte Carlo simulations. In fact, this analysis can spot corner cases that are difficult to detect with a probabilistic method.

A. THE ALGORITHM

In the current implementation of the method, a raster map B_i is used to represent the backward reachable tube of the

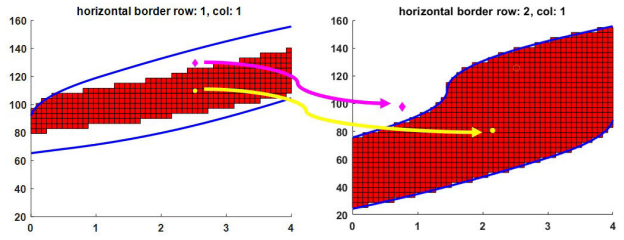

FIGURE 17. Example of inverse mapping using raster maps.

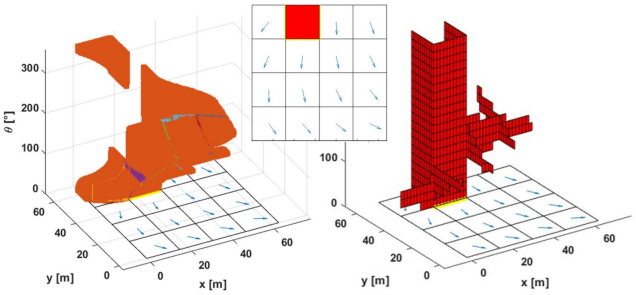

FIGURE 18. 3-D view of the backward reachable tube and 2-D raster maps. The vertical axis indicates the orientation angle and it is wrapped between 0° and 360°.

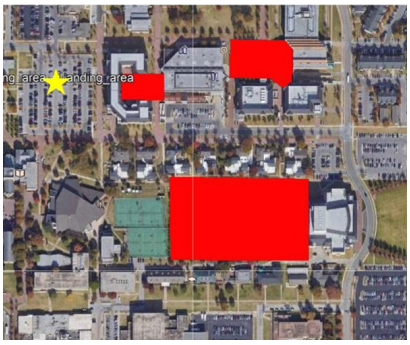

FIGURE 19. Considered scenario: The university of tulsa campus.

obstacle region in the border b_i. A cell of B_i can be either 1 or 0. In the first case, the configurations associated with that cell are considered part of the backward reachable set $\mathcal{B}(O)$. Algorithm 1 illustrates how the computation of $\mathcal{B}(O)$ is performed for every border of the grid-map. The main task is the inverse mapping of the filled cells of B_i toward the neighboring borders from which they are reached. The algorithm iterates until it converges to the final solution. In each iteration, only borders in which a new configuration was set to 1 are considered.

Fig. 17 illustrates an example of inverse mapping based on raster maps. The mapping is performed in two steps. In the first step, the BRT $\mathcal{B}(b_i)$ is computed in b_j. In the second step, any configuration $q_k = (x_k, \theta_k)$ of b_j that is contained in $\mathcal{B}(b_i)$ is tested. If the forward mapping of q_k falls in an occupied cell of b_i, then the cell of b_j corresponding to q_k is set to 1. In Fig. 17, the mapping of the yellow circle falls in an occupied cell, therefore the corresponding cell is set to 1. Instead, the mapping of the purple diamond falls in a free cell; therefore, its corresponding cell remains empty.

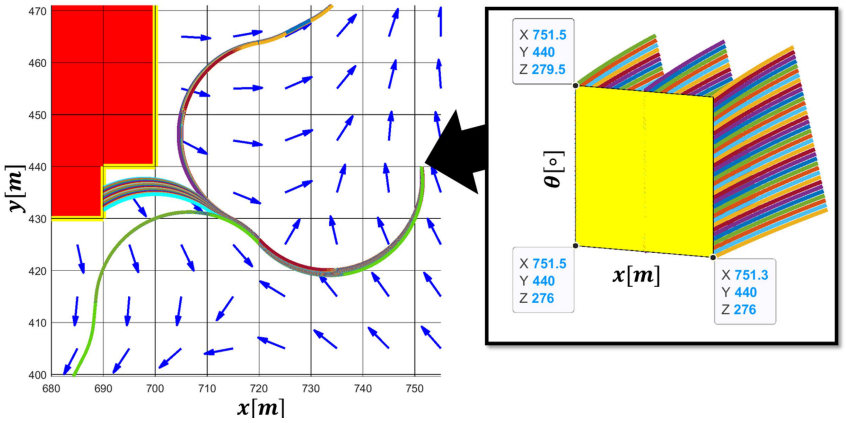

FIGURE 20. Example of corner cases. Trajectories start from the set shown on the right, where the angle range is 3.5° and position range is 0.2m.

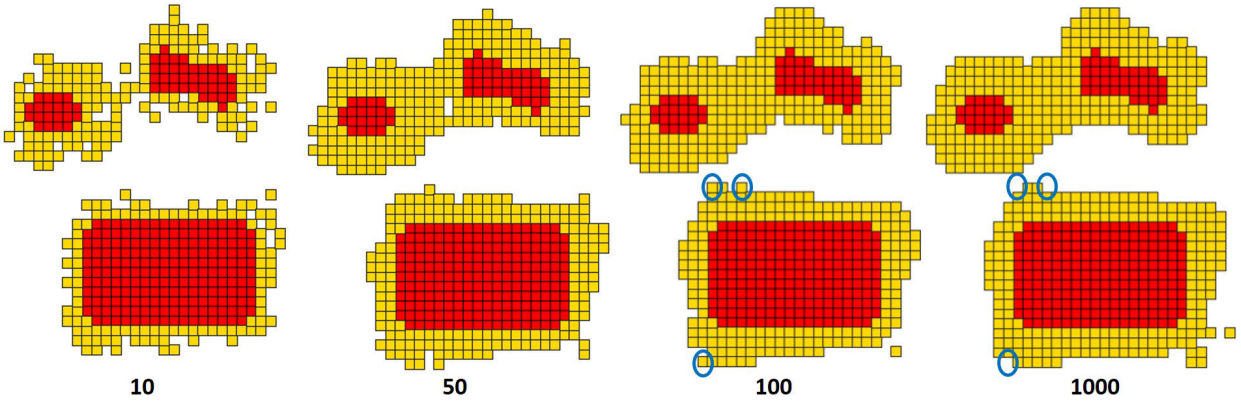

FIGURE 21. Results of Monte Carlo simulations using 10, 50, 100, and 1000 samples per cell.

Algorithm 1 Iterative Backward Propagation

Input: W_*, grid-map GM
Output: Backward Reachable Tube $\mathcal{B}(W_*)$

```
while (new configuration is added to B(W_*)) do
  for (Border B_{i_t}) do
    if B_{i_t} ≠ B_{i_{t-1}} then
      for neighbor B_j do
        propagate sets of B_{i_t} to B_j
      end for
    end if
  end for
end while
return BRT
```

The space complexity can be evaluated by assuming that the grid-map used to represent the workspace is a square having n cells per side. Neglecting the borders at the margins of the grid-map, the number of tables necessary to characterize the whole workspace is $2n(n-1)$. Assuming that angle and position at the border are discretized using m and $2m$ cells, the number of bits necessary to represent $\mathcal{B}(W_*)$ is $2m^2(2n(n-1))$. Instead, if the whole 3-D configuration space was discretized, $2m^3n^2$ bits would be necessary. For instance with $n = 20$ and $m = 200$, the number of required bits in this method is $1.52 \cdot 10^7$, while for a 3-D discretization would be $6.4 \cdot 10^9$.

In Fig. 18, there is an example showing the 3-D view of the BRT $\mathcal{B}(O)$ and the 2-D raster maps computed with the algorithm illustrated here.

VII. COMPARISON WITH MONTE CARLO SIMULATIONS

In this section, we compare the safety analysis performed using the proposed algorithm with the analysis based on Monte Carlo simulations. The plan was generated for the scenario illustrated in Fig. 19 using the method described in [32]–[34]. The motion plan was generated with an intentional bad tuning to have a large unsafe region. In the figure, red polygons are no-fly zones, whose purpose is to prevent UAVs from flying over areas with a high density of people; the yellow star is the goal location. The resolution of the grid-map used in our comparison was $10m$ and the minimum turning radius was assumed to be $18m$.

Fig. 20 shows one of the corner cases. In this scenario, there is an extremely small set of trajectories that oscillate between turn directions and intersect the restricted airspace. The plan which produces this behavior looks reasonable given the obstacle set and goal location; however, the bang-bang controller can lead to this type of trajectory when the error in the heading angle is close to π. We must point out that

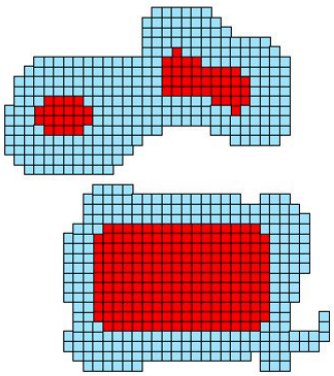

FIGURE 22. Unsafe region computed using the proposed geometric method.

situations like the one presented here are nearly impossible to predict by observation alone; and thus, automatic methods to uncover them are essential. In particular, in Fig. 20, it is possible to notice that all the trajectories start from a very small set of states, having a heading angle range of just 3.5° and a position range of just $0.2m$, which is very small if compared with the resolution of the cells ($10m$) and the minimum turning radius of the UAV ($18m$).

The example in Fig. 20 shows that, in the considered region, the set of trajectories ending in the obstacle region is very small, and that there is a ''jump'' in the distance between trajectories and obstacle region. Therefore, it is clear that finding those corner cases is extremely difficult. Cases like these highlight the need for exhaustive verification due to the difficulty of discovering them with Monte Carlo simulations.

An additional reason that urges us to find alternatives to Monte Carlo analysis is that the error does not consistently improve as the number of simulations is increased. This phenomenon is visible in Fig. 21, where some unsafe cells missed by the analysis with 1000 samples per cell were instead spotted by the analysis with 100 samples per cell.

While Fig. 21 shows the unsafe regions computed using Monte Carlo simulations with a various number of samples per cell, Fig. 22 shows the results obtained with the algorithm illustrated in Section IV. These results were obtained using a resolution of 0.2 m for the position and 3.6° for the heading angle. In Fig. 23, there is the overlapping between the results obtained with the Monte Carlo simulations and the results obtained with our algorithm. Furthermore, Fig. 23 shows one trajectory for each unsafe cell missed by the Monte Carlo analysis.

From Fig. 23 it is clear that the analysis performed using Monte Carlo simulations misses important corner cases that would cause unsafe states, while they were detected using our method. This is possible (and expected) because the proposed geometric method allows the testing of an infinite number of configurations simultaneously. Furthermore, even in the case of a simple implementation like the one presented in Section IV, the error due to discretization is consistent. Which in our case means that it is possible to set a threshold to the used resolution, above which it is guaranteed to detect

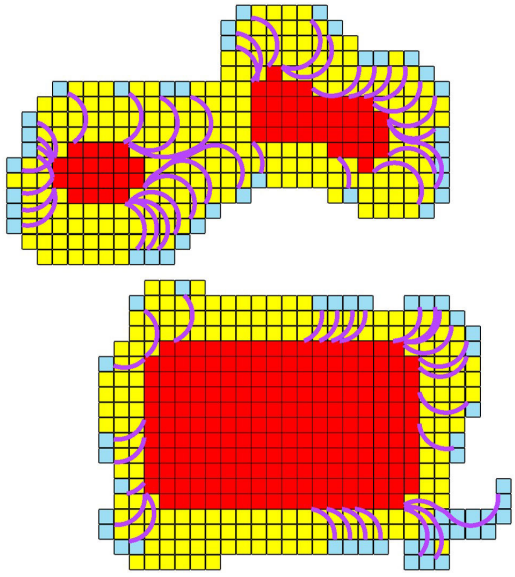

FIGURE 23. Overlapping of the results obtained with Monte Carlo simulations (yellow) and our method (light blue).

specific unsafe resultant trajectories in a cell. As we have shown above, this is certainly not true for a probabilistic method.

VIII. CONCLUSION

In this paper, we presented a novel analytic method suitable for the safety analysis of feedback motion plans based on discrete vector fields. The analytic method is based on the concept of reachability. However, rather than using a general formulation such as Hamilton-Jacobi, specific functions are derived exploiting geometric principles. The practical implementation of this analytic tool presented some challenges that were overcome with a solution based on the iterative propagation of rasterized sets. With this type of analysis, it is possible to find corner cases, which are difficult to detect using Monte Carlo simulations. This was shown comparing the unsafe region that was obtained using Monte Carlo simulations and the region obtained with the iterative algorithm.

The research described in this paper sets the base for a new approach for safety analysis in the autonomous guidance, navigation, and control of fixed-wing aircraft (or, in general, vehicles whose kinematics can be approximated with a Dubin's path). Future developments can have multiple directions. From the authors' point of view, the areas that deserve more focus are the followings:

1) A different implementation of the proposed analytic tool should be derived. The new implementation should propagate the boundaries of the sets instead of their elements. Furthermore, an algorithmic solution must be formulated for the cases in which there is no closed-form solution in the inverse mapping.
2) The analytic formulation should be extended to other cases involving other types of controllers, and therefore various types of trajectories.

REFERENCES

[1] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[2] G. Miraglia and L. Hook IV, "A feedback motion plan for vehicles with bounded curvature constraints," 2019, *arXiv:1910.06460*. [Online]. Available: http://arxiv.org/abs/1910.06460

[3] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi, "Computational techniques for the verification of hybrid systems," *Proc. IEEE*, vol. 91, no. 7, pp. 986–1001, Jul. 2003.

[4] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton, NJ, USA: Princeton university press, 2012.

[5] T. A. Johansen and T. I. Fossen, *Guidance, Navigation, and Control of Fixed-Wing Unmanned Aerial Vehicles BT—Encyclopedia of Robotics*, M. H. Ang, O. Khatib, B. Siciliano, Eds. Berlin, Germany: Springer, 2020, pp. 1–9, doi: 10.1007/978-3-642-41610-1_67-1.

[6] J.-M. Kai, T. Hamel, and C. Samson, "A unified approach to fixed-wing aircraft path following guidance and control," *Automatica*, vol. 108, Oct. 2019, Art. no. 108491. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109819303437

[7] J. P. Wilhelm and G. Clem, "Vector field UAV guidance for path following and obstacle avoidance with minimal deviation," *J. Guid., Control, Dyn.*, vol. 42, no. 8, pp. 1848–1856, Aug. 2019.

[8] D. Jung, J. Ratti, and P. Tsiotras, "Real-time implementation and validation of a new hierarchical path planning scheme of UAVs via hardware-in-the-loop simulation," *J. Intell. Robotic Syst.*, vol. 54, nos. 1–3, pp. 163–181, Mar. 2009.

[9] C. Pek, M. Koschi, and M. Althoff, "An online verification framework for motion planning of self-driving vehicles with safety guarantees," in *Proc. AAET-Automatisiertes Vernetztes Fahren*, 2019, pp. 260–274.

[10] A. Rizaldi, F. Immler, B. Schürmann, and M. Althoff, "A formally verified motion planner for autonomous vehicles," in *Proc. 16th Int. Symp., ATVA*. Los Angeles, CA, USA: Springer, Oct. 2018, pp. 75–90.

[11] I. M. Mitchell, "Comparing forward and backward reachability as tools for safety analysis," in *Proc. 10th Int. Workshop, HSCC*. Pisa, Italy: Springer, Apr. 2007, pp. 428–443.

[12] E. Asarin, T. Dang, G. Frehse, A. Girard, C. Le Guernic, and O. Maler, "Recent progress in continuous and hybrid reachability analysis," in *Proc. IEEE IEEE Conf. Comput. Aided Control Syst. Design Int. Conf. Control Appl. Int. Symp. Intell. Control*, Oct. 2006, pp. 1582–1587.

[13] M. Shubert, M. Oishi, M. Baldwin, and R. S. Erwin, "Under-approximating reach-avoid sets for space vehicle maneuvering in the presence of debris," *IFAC-PapersLine*, vol. 51, no. 12, pp. 142–147, 2018. [Online]. Available: [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896318308450

[14] B. Xue, Z. She, and A. Easwaran, *Under-Approximating Backward Reachable Sets by Polytopes BT—Computer Aided Verification*, S. Chaudhuri and A. Farzan, Eds. Cham, Switzerland: Springer, 2016, pp. 457–476.

[15] A. El-Guindy, D. Han, and M. Althoff, "Estimating the region of attraction via forward reachable sets," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 1263–1270.

[16] E. C. Kerrigan, J. Lygeros, and J. M. Maciejowski, "A geometric approach to reachability computations for constrained discrete-time systems," *IFAC Proc. Volumes*, vol. 35, no. 1, pp. 323–328, 2002.

[17] M. R. Hidalgo and R. Joan-Arinyo, "The reachability problem in constructive geometric constraint solving based dynamic geometry," *J. Automated Reasoning*, vol. 52, no. 1, pp. 99–122, Jan. 2014.

[18] A. Desai and N. Michael, "Online planning for quadrotor teams in 3-D workspaces via reachability analysis on invariant geometric trees," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 8769–8775.

[19] J. Nilsson, J. Fredriksson, and A. C. E. Ödblom, "Verification of collision avoidance systems using reachability analysis," *IFAC Proc. Volumes*, vol. 47, no. 3, pp. 10676–10681, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016433100

[20] D. Althoff, M. Althoff, and S. Scherer, "Online safety verification of trajectories for unmanned flight with offline computed robust invariant sets," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 3470–3477.

[21] C. Pek and M. Althoff, "Efficient computation of invariably safe states for motion planning of self-driving vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 3523–3530.

[22] M. Chen and C. J. Tomlin, "Hamilton–Jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management," *Annu. Rev. Control, Robot., Auto. Syst.*, vol. 1, no. 1, pp. 333–358, May 2018.

[23] D. Lee, M. Chen, and C. J. Tomlin, "Removing leaking corners to reduce dimensionality in hamilton-jacobi reachability," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 9320–9326.

[24] I. M. Mitchell and C. J. Tomlin, "Overapproximating reachable sets by Hamilton-Jacobi projections," *J. Sci. Comput.*, vol. 19, nos. 1–3, pp. 323–346, 2003, doi: 10.1023/A:1025364227563.

[25] M. Chen, S. Herbert, and C. J. Tomlin, "Fast reachable set approximations via state decoupling disturbances," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 191–196.

[26] R. Y. Rubinstein and D. P. Kroese, *Simulation Monte Carlo Method*, vol. 10. Hoboken, NJ, USA: Wiley, 2016.

[27] G. Rubino and B. Tuffin, *Rare Event Simulation Using Monte Carlo Methods*. Hoboken, NJ, USA: Wiley, 2009.

[28] K. Margellos and J. Lygeros, "Toward 4-D trajectory management in air traffic control: A study based on Monte Carlo simulation and reachability analysis," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1820–1833, Sep. 2013.

[29] G. Miraglia and L. R. Hook, "Feedback motion plan verification for vehicles with bounded curvature constraints," in *Proc. IEEE 10th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Oct. 2019, pp. 0844–0854.

[30] G. Miraglia, "Planning and verification under minimum turning radius constraints," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Tulsa, Tulsa, OK, USA, 2019. [Online]. Available: https://search.proquest.com/openview/747486d39c11cf1dc6dac8274467d1e0/1pq-origsite=gscholar&cbl=51922&diss=y

[31] *Hjreachability/Helperoc*. Accessed: Sep. 16, 2020. [Online]. Available: https://github.com/HJReachability/helperOC

[32] G. Miraglia and L. Hook, "Dynamic geo-fence assurance and recovery for nonholonomic autonomous aerial vehicles," in *Proc. IEEE/AIAA 36th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2017, pp. 1–7.

[33] G. Miraglia, L. R. Hook, T. Fiorenzani, K. N. Maleki, and M. A. Skoog, "Discrete vector fields for 2-D navigation under minimum turning radius constraints," *Intell. Service Robot.*, vol. 13, no. 3, pp. 343–363, Jul. 2020, doi: 10.1007/s11370-020-00317-8.

[34] G. Miraglia and L. R. Hook, "A feedback motion plan for vehicles with bounded curvature constraints," in *Proc. IEEE 10th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Oct. 2019, pp. 1180–1186.

GIOVANNI MIRAGLIA (Member, IEEE) received the M.S. degree in electronics engineering from La Sapienza University, Rome, Italy, in 2016, and the Ph.D. degree in computer engineering from The University of Tulsa, Tulsa, OK, USA, in 2019.

He is currently a Senior Software Engineer at The MathWorks, Inc., Natick, MA, USA. His research interests include autonomous vehicle navigation systems and VnV technologies for autonomous systems. He is a member of the honor society IEEE-HKN. He currently serves as the Vice-Chair for the Boston Chapter of the IEEE Robotics and Automation Society.

LOYD R. HOOK (Member, IEEE) received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Oklahoma, in 2006 and 2011, respectively. He has worked in aircraft autonomous systems research and implementation for the past 15 years beginning at NASA Dryden/Armstrong flight research center and continuing at his current position as an Assistant Professor at the University of Tulsa. He is currently the Director of the TU Vehicle Autonomy and Intelligence Lab (TU VAIL), where his research focus is the development of safety assured autonomous vehicle systems.