Robust Sequential Trajectory Planning Under Disturbances and Adversarial Intruder
Mo Chen, Somil Bansal, Jaime F. Fisac, and Claire J. Tomlin, Fellow, IEEE

Abstract—Provably safe and scalable multivehicle trajectory planning is an important and urgent problem. Although this problem has been studied in the past, there has not been a method that guarantees both goal satisfaction and safety for vehicles with general nonlinear dynamics, while taking into account disturbances and potential adversarial agents, to the best of our knowledge. Hamilton–Jacobi (HJ) reachability is the ideal tool for guaranteeing goal satisfaction and safety under such scenarios, which has been successfully applied to many small-scale problems; however, its direct application in most cases becomes intractable when there are more than two vehicles due to the exponentially scaling computational complexity with respect to system dimension. In this paper, we take advantage of the guarantees provided by HJ reachability and eliminate the computational burden by assigning a strict priority ordering to vehicles under consideration. Under this sequential trajectory planning (STP) scheme, vehicles reserve “space–time” portions in the airspace. The space–time portions guarantee dynamic feasibility, collision avoidance, and optimality of trajectories given the priority ordering. With a computation complexity that scales quadratically when accounting for both disturbances and an intruder, and linearly when accounting for only disturbances, the STP can tractably solve the multivehicle trajectory planning problem for vehicles with general nonlinear dynamics in a practical setting. We demonstrate our theory in representative simulations.

Index Terms—Collision avoidance, multi-robot systems, optimal control, safety-critical systems, trajectory planning, unmanned aerial vehicles, unmanned airspace traffic management.

I. INTRODUCTION

RECENTLY, there has been an immense surge of interest in the use of unmanned aerial systems (UASs) in urban environments. Civil applications of UASs will involve unmanned aerial vehicles (UAVs) flying in urban environments, potentially in close proximity of humans, other UAVs, and other important assets. As a result, government agencies, such as the Federal Aviation Administration and National Aeronautics and Space Administration of the United States, are urgently trying to develop new scalable ways to organize an airspace in which potentially thousands of UAVs can fly simultaneously in the same region [1], [2].

One essential question that needs to be answered is how a group of vehicles in the same vicinity can reach their destinations while avoiding situations which are considered dangerous, such as collisions. In some previous studies that address the collision avoidance problem, specific control strategies for the vehicles are assumed. These approaches include those using velocity obstacles [3]–[6], which anticipate the future positions of vehicles based on their current speeds. Reactive collision avoidance methods include those using virtual potential fields [7], [8]; these methods discourage collisions by modifying vehicle control actions so that there is an effective repulsive force between the vehicles. Collision avoidance among vehicles with kinematics Dubins car dynamics has also been analyzed by Pallottino et al. [9], in which they propose reserving local regions around each vehicle to allow for predetermined maneuvers; this framework is formalized using a hybrid automaton. Zhou et al. [10] proposed a Voronoi-based method that guarantees collision avoidance under decentralized planning for holonomic vehicles.

Other analyses of multivehicle systems include methods for real-time trajectory generation [11], path planning for vehicles with linear dynamics in the presence of obstacles with known motion [12], and cooperative path planning via waypoints which do not account for vehicle dynamics [13] or which indirectly account for dynamics via smoothing through a mixed integer program [14]. Other related work includes those which consider only the collision avoidance problem without path planning. These results include those that assume the system has a linear model [15]–[17], rely on a linearization of the system model [18], [19], assume a simple positional state space [20], and many others [21]–[23].

Flexible planning provably safe and dynamically feasible trajectories without making strong assumptions on the vehicles’ dynamics and other vehicles’ motion is essential for dense groups of UAVs to safely fly in each other’s vicinity. In addition, in a practical setting, any trajectory or path planning scheme that also addresses collision avoidance must guarantee both goal satisfaction and safety of UAVs despite disturbances, such as weather effects and communication faults [2]. Furthermore, unexpected scenarios, such as UAV malfunctions or even UAVs, with malicious intent need to be accounted for.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.
The problem of trajectory planning and collision avoidance under disturbances in safety-critical systems has been studied using Hamilton–Jacobi (HJ) reachability analysis, which provides guarantees on goal satisfaction and safety of optimal system trajectories [24]–[29]. Reachability-based methods are particular suitable in the context of UAVs, because of the hard guarantees that are provided. In reachability analysis, one computes the reachable set, defined as the set of states from which the system can be driven to a target set. Many numerical tools are available for computing various definitions of reachable sets [30]–[33], and reachability analysis has been successfully used in applications involving systems with no more than two vehicles [25], [34]–[36].

One of the main challenges of managing the next generation of airspace is the density of vehicles that needs to be accommodated [2]. Such a large-scale system has a high-dimensional joint state space, making a direct application of dynamic programming-based approaches such as reachability analysis intractable. In particular, reachable set computations involve solving a HJ partial differential equation (PDE) or variational inequality (VI) on a grid representing a discretization of the state space, causing computational complexity to scale exponentially with system dimension.

A. Contributions and Outline

In this paper, we propose the sequential trajectory planning (STP) method to tackle the multivehicle trajectory planning problem. Our approach is similar to the approaches of [37] and [38], but provides hard guarantees on both the goal satisfaction and safety of all vehicles even in the presence of disturbances and a single intruder vehicle that could potentially be adversarial. In addition, our method scales only linearly with the number of vehicles when there is no intruder, and quadratically with the number of vehicles when there is a single intruder. On a high level, the STP assigns a strict priority ordering to vehicles under consideration. Higher priority vehicles plan trajectories without accounting for lower priority vehicles. Lower-priority vehicles treat higher priority vehicles as moving obstacles. Under this assumption, time-varying HJ reachability [27], [29] can be used to obtain optimal and provably safe trajectories for each vehicle, starting from the highest priority vehicle.

In a sense, the STP method reserves a portion of “space–time” in the airspace for each vehicle. The reserved space–time portion is recorded so that lower priority vehicles can take it into account. Besides planning around the reserved space–time portions of higher priority vehicles, no other communication between the vehicles is needed at execution time, even when disturbances and an intruder are present. Besides reducing computational complexity, a priority-based approach makes practical sense as priorities can be assigned in a first-come-first-serve basis by, for example, an air navigation service provider [2], based on flight requests from various companies and agencies sharing the airspace. More sophisticated prioritization of vehicles may include trajectory planning in “batches” of vehicles, as suggested by [39]. The details of choosing a priority are not the focus of this paper.

Almost all computation is done offline to produce a value function, corresponding to an appropriate differential game, for each vehicle. The gradient of the value function can be stored in a lookup table, which is used online to synthesize the optimal controller. Controller synthesis amounts to evaluating an analytic expression, is the only online computation, and therefore, can be done in real time.

In the absence of disturbances and intruders, and assuming each vehicle has perfect information about other vehicles’ positions, each vehicle may plan and commit to an exact trajectory, with the reserved space–time being the collision set around the trajectory at every point in time. This basic concept of the STP is formally presented in Section IV.

When the vehicles are affected by disturbances, exact trajectories cannot be known a priori, and thus, the basic STP algorithm cannot be directly applied. Fortunately, reachability analysis allows us to determine, at no additional computation cost, all possible states of each vehicle over time under the worst case disturbance, given a control strategy. In addition, we can also determine suitable portions of space–time for each vehicle depending on the available information about the control strategies of higher priority vehicles. STP under disturbances and three different assumptions on the information available about the control strategy of other vehicles is formally presented in Section V.

In scenarios where there could potentially be single, possibly adversarial intruder in the airspace, each vehicle needs extra space around other vehicles in order to be able to perform avoidance maneuvers. Assuming the intruder may be present for some maximum duration, we use reachability analysis to determine precisely the amount of space–time needed for each vehicle to be able to avoid the intruder under the presence of disturbances, making our proposed method sufficiently robust to most practical scenarios. STP in the presence of a single intruder is formally presented in Section VI.

II. PROBLEM FORMULATION

Consider N vehicles which participate in the STP process and denote these vehicles as the STP vehicles Q_i, i = 1, ..., N. We assume their dynamics are given by

\[
\begin{align*}
\dot{x}_i &= f_i(x_i, u_i, d_i), & t \leq t^{\text{STA}}_i \\
u_i &\in U_i, & d_i \in D_i, & i = 1, ..., N
\end{align*}
\]

(1)

where \(x_i \in \mathbb{R}^{n_i}\) represents the state of vehicle \(Q_i\), \(u_i \in U_i\) the control of \(Q_i\), and \(d_i \in D_i\) the disturbance experienced by \(Q_i\). For convenience, we partition the state \(x_i\) into the position component \(p_i \in \mathbb{R}^{n_p}\) and the nonposition component \(h_i \in \mathbb{R}^{n_i-n_p}: x_i = (p_i, h_i)\).

We assume that the control functions \(u_i(\cdot), d_i(\cdot)\) are drawn from the set of measurable functions. For convenience, we will use the sets \(U_i\) and \(D_i\) to, respectively, denote the set of functions from which the control and disturbance functions \(u_i(\cdot)\) and \(d_i(\cdot)\) are drawn. We further assume that the flow field \(f_i: \mathbb{R}^{n_i} \times U_i \times D_i \to \mathbb{R}^{n_i}\) is uniformly continuous, bounded, and Lipschitz continuous in \(x_i\) for fixed \(u_i\) and \(d_i\). With this assumption, given \(u_i(\cdot) \in U_i\) and \(d_i(\cdot) \in D_i\), there exists a unique trajectory solving (1) [40]. Finally, we assume that the
disturbances $d_i(\cdot)$ are drawn from the set of nonanticipative strategies $[25] \Gamma$
\[
\Gamma := \{ \mathcal{N} : \mathbb{U}_i \rightarrow \mathbb{D}_i : u_i(r) = \hat{u}_i(r) \text{ a. e. } r \in [t, s] \}
\Rightarrow \mathcal{N}[u_i](r) = \mathcal{N}[\hat{u}_i](r) \text{ a. e. } r \in [t, s] \quad (2)
\]
Each vehicle $Q_i$ has an initial state $x_i^0$ and aims to reach its target $L_i$ by some scheduled time of arrival $t_{i}^{STA}$. The target in general represents some set of desirable states, for example the destination of $Q_i$. In some situations, we may find that it is infeasible for $Q_i$ to get to $L_i$ at or before $t_i^{STA}$. Whenever unsure, we may first determine the earliest feasible $t_i^{STA}$ as described in Section VI.

On its way to $L_i$, $Q_i$ must avoid a set of static obstacles $O_i^{static} \subset \mathbb{R}^n_i$. The interpretation of $O_i^{static}$ could be a tall building, a region around an airport, or any set of states that are forbidden for each STP vehicle. In addition to the static obstacles, each vehicle $Q_i$ must also avoid the danger zones with respect to every other vehicle $Q_j$, $j \neq i$. The danger zones in general can represent any joint configurations between $Q_i$ and $Q_j$ that are considered to be unsafe. In this paper, we define the danger zone of $Q_i$ with respect to $Q_j$ to be
\[
Z_{ij} = \{(x_i, x_j) : \|p_i - p_j\|_2 \leq R_c\} \quad (3)
\]
whose interpretation is that $Q_i$ and $Q_j$ are considered to be in an unsafe configuration when they are within a distance of $R_c$ of each other. For concreteness, we will call $Z_{ij}$ the collision set, and if $(x_i, x_j) \in Z_{ij}$, then $Q_i$ and $Q_j$ are said to have collided. Note that there may exist an inevitable danger zone given the position-based danger zone $Z_{ij}$ (for example the states from which wind can push an aerial vehicle into $Z_{ij}$ irrespective of control applied by the vehicle). Reachability analysis presented in this paper by definition guarantees that vehicles avoid such inevitable danger zones. Finally, if the vehicles are not particles (as often is the case), the danger zone $Z_{ij}$ can be defined to include the effective sizes of the vehicles.

Given the set of STP vehicles, their targets $L_i$, the static obstacles $O_i^{static}$, and the vehicles' danger zones with respect to each other $Z_{ij}$, we would like, for each vehicle $Q_i$, to synthesize a controller which guarantees that $Q_i$ reaches its target $L_i$ at or before the scheduled time of arrival $t_i^{STA}$, while avoiding the static obstacles $O_i^{static}$ as well as the danger zones with respect to all other vehicles $Z_{ij}$, $j \neq i$. In addition, we would like to obtain the latest departure time $t_i^{LDT}$ such that $Q_i$ can still arrive at $L_i$ on time. Note that as long as it is feasible a vehicle to reach its target in the absence of all other vehicles, producing a safe and timely trajectory is always feasible using our proposed algorithms, since the latest departure time $t_i^{LDT}$ is obtained. Indeed, if the environment is expected to be crowded, a vehicle can simply depart early enough (and potentially arrive very early) to “avoid traffic.” In practice, if departing at or before the latest departure time $t_i^{LDT}$ is not possible, then arriving on time is infeasible.

In general, the above optimal trajectory planning problem must be solved in the joint space of all $N$ STP vehicles. However, due to the high-joint dimensionality, a direct dynamic programming-based solution is intractable. Therefore, we propose to assign a priority to each vehicle and perform STP given the assigned priorities. Without the loss of generality, let $Q_j$ have a higher priority than $Q_i$ if $j < i$. Under the STP scheme, higher priority vehicles can ignore the presence of lower priority vehicles and perform trajectory planning without taking into account the lower priority vehicles’ danger zones. A lower priority vehicle $Q_j$, on the other hand, must ensure that it does not enter the danger zones of the higher priority vehicles $Q_j$, $j < i$; each higher priority vehicle $Q_j$ induces a set of time-varying obstacles $O_j^\tau(t)$, which represents the possible states of $Q_j$ such that a collision between $Q_i$ and $Q_j$ could occur.

It is straightforward to see that if each vehicle $Q_i$ is able to plan a trajectory that takes it to $L_i$ while avoiding the static obstacles $O_i^{static}$ and the danger zones of higher priority vehicles $Q_j$, $j < i$, and then, the set of STP vehicles $Q_i$, $i = 1, \ldots, N$ would all be able to reach their targets safely. Without the STP scheme, the additional structure provided by the vehicle priorities allows us to reduce the complexity of the joint trajectory planning problem. As we will see, under the STP scheme, trajectory planning can be done sequentially in descending order of vehicle priority in the state space of only a single vehicle. Thus, the STP provides a solution whose complexity scales linearly with the number of vehicles in the presence of disturbances, as opposed to exponentially with a direct application of dynamic programming approaches. In the presence of a single intruder, the computation complexity scaling becomes quadratic. As mentioned earlier, priorities of vehicles may be assigned in a first-come-first-serve basis through, for example, an air navigation service provider [2], which manages a region of the low-altitude airspace.

In Sections IV–VI, we will explore the STP under different assumptions. We begin with the basic STP algorithm in which disturbances are ignored and perfect information of vehicles’ positions is assumed. This simplification allows us to clearly establish the basic STP algorithm. Next, we show how the basic STP approach can be made robust to disturbances as well as an imperfect knowledge of other vehicles’ positions. Finally, we further robustify the STP approach by considering how the set of STP vehicles may respond to the presence of an intruder vehicle, which may be adversarial. All of our methods use time-varying reachability analysis to provide goal satisfaction and safety guarantees.

III. TIME-VARYING REACHABILITY BACKGROUND

STP is based on reachability theory. In particular, we will be using reachability analysis to compute either a backward reachable set (BRS) $\mathcal{V}$, a forward reachable set (FRS) $\mathcal{W}$, or a sequence of BRSs and FRSs, given some target set $L$, time-varying obstacle $G(t)$, and the Hamiltonian function $H$ which captures the system dynamics as well as the roles of the control and disturbance. Intuitively, the BRS is the set of states such that the system trajectories that start from this set can reach some given target set without colliding with obstacles, despite the worst case disturbance. Similarly, FRS is the set of states that can be reached by a system starting from a target set. The BRS $\mathcal{V}$ in a time interval $[t, t_f]$ or FRS $\mathcal{W}$ in a time
interval \([t_0, t]\) will be denoted by

\[
\begin{align*}
\mathcal{V}(t, t_f) & \quad (\text{backward reachable set}) \\
\mathcal{W}(t_0, t) & \quad (\text{forward reachable set}).
\end{align*}
\] (4)

The computation of BRS (or FRS) can be formulated as a differential game between the control and disturbance (or as an optimal control problem when no disturbance is not involved), which can be solved using the principle of dynamic programming. Ultimately, this results in solving an HJPDE. For further details on this formulation, we refer the interested readers to [41]. Traditionally, only static obstacles had been considered in the reachability formulation. However, in the STP scheme, a lower priority vehicle must avoid a set of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target. Several formulations of reachability are able to perform optimal trajectory planning with hard guarantees of moving obstacles (i.e. time-varying obstacles) on its way to the target.

\[
\max \{\min \{D_i V(t, x) + H(t, x, \nabla V(t, x)), l(x) - V(t, x)\}, -g(t, x) - V(t, x) = 0, \quad t \leq t_f\}
\]

\[
V(t_f, x) = \max \{l(x), -g(t_f, x)\}
\]

(5)

where \(D_i V(t, x)\) and \(\nabla V(t, x)\) denote the time and space derivatives of the value function \(V\), respectively. In a similar fashion, the FRS is computed by solving the following initial value double-obstacle HJ VI:

\[
D_i W(t, x) + H(t, x, \nabla W(t, x)) = 0, \quad t \geq t_0 \quad W(t_0, x) = \max \{l(x), -g(t_0, x)\}.
\]

(6)

In both (5) and (6), the function \(l(x)\) is the implicit surface function representing the target set \(L = \{x : l(x) \leq 0\}\). Similarly, the function \(g(t, x)\) is the implicit surface function representing the time-varying obstacles \(G(t) = \{x : g(t, x) \leq 0\}\). The BRS \(\mathcal{V}(t, t_f)\) and FRS \(\mathcal{W}(t_0, t)\) are given by

\[
\mathcal{V}(t, t_f) = \{x : V(t, x) \leq 0\} \\
\mathcal{W}(t_0, t) = \{x : W(t, x) \leq 0\}.
\]

(7)

Some of the reachability computations will not involve an obstacle set \(G(t)\), in which case we can simply set \(g(t, x) = \infty\) which effectively means that the outside maximum is ignored in (5). In addition, note that unlike in (5), there is no inner minimization in (6). As we will see later, we will be using the BRS to determine all states that can reach some target set within the time horizon \([t, t_f]\), whereas we will be using the FRS to determine where a vehicle could be at some particular time \(t\). In addition, (6) has no outer maximum, since the FRSSs that we will compute will not involve any obstacles.

The Hamiltonian, \(H(t, x, \nabla V(t, x))\), depends on the system dynamics and the role of control and disturbance. Whenever \(H\) does not depend explicitly on \(t\), we will drop the argument. In addition, the Hamiltonian is an optimization that produces the optimal control \(u^*(t, x)\) and optimal disturbance \(d^*(t, x)\), once \(V\) is determined. For BRSs, whenever the existence of a control (“\(\exists u\)”) or disturbance is sought, the optimization is a minimum over the set of controls or disturbance. Whenever a BRS characterizes the behavior of the system for all controls (“\(\forall u\)”) or disturbances, the optimization is a maximum. We will introduce precise definitions of reachable sets, expressions for the Hamiltonian, expressions for the optimal controls as needed for the many different reachability calculations we use.


\[
\begin{align*}
\text{IV. STP WITHOUT DISTURBANCES AND} \\
\text{WITH PERFECT INFORMATION}
\end{align*}
\]

In this section, we introduce the basic STP algorithm assuming that there is no disturbance affecting the vehicles and that each vehicle knows the exact position of higher priority vehicles. Although in practice such assumptions do not hold, the description of the basic STP algorithm will introduce the notation needed for describing the subsequent, more realistic versions of the STP. We also show simulation results for the basic STP algorithm. The majority of the content in this section is taken from [42].

\[
\begin{align*}
A. \text{Theory}
\end{align*}
\]

Recall that the STP vehicles \(Q_i, i = 1, \ldots, N\), are each assigned a strict priority, with \(Q_j\) having a higher priority than \(Q_i\) if \(j < i\). In the absence of disturbances, we can write the dynamics of the STP vehicles as

\[
\dot{x}_i = f_i(x_i, u_i), \quad 1 \leq i^{\text{STA}} \leq u_i \in U_i, \quad i = 1, \ldots, N.
\]

(8)

In STP, each vehicle \(Q_i\) plans the trajectory to its target set \(L_i\) while avoiding static obstacles \(Q_i\) and the obstacles \(O_j\) induced by higher priority vehicles \(Q_j, j < i\). Path planning is done sequentially starting from the first vehicle and proceeding in descending priority, \(Q_1, Q_2, \ldots, Q_N\), so that each of the trajectory planning problems can be done in the state space of only one vehicle. During its trajectory planning process, \(Q_i\) ignores the presence of lower priority vehicles \(Q_k, k > i\), and induces the obstacles \(O_j(t)\) for \(Q_k, k > i\).

From the perspective of \(Q_i\), each higher priority vehicle \(Q_j, j < i\), induces a time-varying obstacle denoted \(O_j(t)\) that \(Q_i\) needs to avoid. Therefore, each vehicle \(Q_i\) must plan its trajectory to \(L_i\), while avoiding the union of all the induced obstacles as well as the static obstacles. Let \(G_i(t)\) be the union of all the obstacles that \(Q_i\) must avoid on its way to \(L_i\):

\[
G_i(t) = O_i^{\text{static}} \cup \bigcup_{j=1}^{i-1} O_j(t).
\]

(9)

With full position information of higher priority vehicles, the obstacle induced for \(Q_i\) by \(Q_j\) is simply

\[
O_i^j(t) = \{x_i : \|p_i(t) - p_j(t)\|_2 \leq R_i\}.
\]

(10)

\[1\]Note that the index \(k\) in \(O_k\) denotes vehicles with lower priority than \(Q_i\), and the index \(j\) in \(O_j(t)\) denotes vehicles with higher priority than \(Q_i\).
Algorithm 1 Basic STP Algorithm

Suppose we are given initial conditions $x_0^i$, vehicle dynamics (8), target sets $L_i$, and static obstacles $O_i$, $i = 1, \ldots, N$. For each $i$ in ascending order starting from $i = 1$ (descending order of priority),

1) determine the total obstacle set $G_i(t)$, given in (9). In the case $i = 1$, $G_i(t) = O_i$ for all $t$;
2) compute the BRS $V_i^{basic}(t, t_i^{STA})$ defined in (11). The latest departure time $t_i^{LDT}$ is then given by $\arg \max \{t | V_i^{basic}(t, t_i^{STA}) \}$;
3) determine the trajectory $x_i(\cdot)$ using vehicle dynamics (8), with the optimal control $u_i^{basic}(\cdot)$ given by (13);
4) given $x_i(\cdot)$, compute the induced obstacles $O_i(t)$ for each $k > i$. In the absence of disturbances, $O_i(t)$ is given by (10).

Each higher priority vehicle $Q_j$ plans its trajectory while ignoring $Q_i$. Since trajectory planning is done sequentially in descending order or priority, the vehicles $Q_j$, $j<i$ would have planned their trajectories before $Q_i$ does. Thus, in the absence of disturbances, $p_j(t)$ is a priori known, and therefore $O_i(t)$, $j < i$ are known, deterministic moving obstacles, which means that $G_i(t)$ is also known and deterministic. Therefore, the trajectory planning problem for $Q_i$ can be solved by first computing the BRS $V_i^{basic}(t, t_i^{STA})$, defined as follows:

$$V_i^{basic}(t, t_i^{STA}) = \{ y : \exists u_i(\cdot) \in U_i, x_i(\cdot) \text{ satisfies (8)},$$
$$\forall s \in [t, t_i^{STA}], x_i(s) \notin G_i(s),$$
$$\exists s \in [t, t_i^{STA}], x_i(s) \in L_i, x_i(t) = y \}. \tag{11}$$

The BRS $V(t, t_i^{STA})$ can be obtained by solving (5) with $L = L_i$, $G(t) = G_i(t)$, and the Hamiltonian

$$H_i^{basic}(x_i, \lambda) = \min_{u_i \in U_i} \lambda \cdot f_i(x_i, u_i). \tag{12}$$

Note that $V(t, t_i^{STA})$, by definition, does not contain any states from which it is inevitable to avoid the danger zone $Z_{ij}$ (and $G_i$ in general). Given $V(t, t_i^{STA})$, the optimal control for reaching $L_i$ while avoiding $G_i$ is then given by

$$u_i^{basic}(t, x_i) = \arg \min_{u_i \in U_i} \lambda \cdot f_i(x_i, u_i). \tag{13}$$

from which the trajectory $x_i(\cdot)$ can be computed by integrating the system dynamics, which in this case are given by (8).

In addition, the latest departure time $t_i^{LDT}$ can be obtained from the BRS $V(t, t_i^{STA})$ as $t_i^{LDT} = \arg \max \{t | V_i^{basic}(t, t_i^{STA}) \}$.

In summary, the basic STP algorithm is given in Algorithm 1.

Note that Step 1, which determines the total obstacle set, can be updated recursively by adding a new set of induced obstacles for each next vehicle: $G_{i+1}(t) = G_i(t) \cup O_{i+1}(t)$. In addition, in implementation, Step 4 can be simplified by storing $G_i(t)$ as a lookup table with the maximum dimensionality across all vehicle state spaces. When a vehicle plans its trajectory, irrelevant dimensions of $G_i(t)$ can be ignored.

This observation keeps the computational complexity of our algorithm linear with respect to the number of vehicles.

As mentioned earlier, the basic STP algorithm, as well as all subsequent variants of STP algorithms, will always return a feasible trajectory that arrives at the target on time, as long as a feasible trajectory exists in the absence of other vehicles. This is because a vehicle can simply depart early enough to avoid being blocked by higher priority vehicles. In fact, the latest departure time $t_i^{LDT}$ quantifies exactly when each vehicle needs to depart to arrive on time.

B. Numerical Simulations

We now illustrate the basic STP algorithm using a four-vehicle example. In this example, we will use the following dynamics for each vehicle:

$$\dot{x}_i = v_i \cos \theta_i$$
$$\dot{y}_i = v_i \sin \theta_i$$
$$\dot{\theta}_i = \omega_i$$

where $x_i = (p_{x,i}, p_{y,i}, \theta_i)$ is the state of vehicle $Q_i$, $p_i = (p_{x,i}, p_{y,i})$ is the position, $\theta_i$ is the heading, $v_i$ is the speed, and $\omega_i$ is the turn rate. In this example, we assume that the vehicles have constant speed $v_i = 1 \forall i$, and the control of each vehicle $Q_i$ is given by $u_i = \omega_i$ with $|\omega_i| \leq \bar{\omega} \forall i$.

We have chosen these dynamics for clarity of illustration; the STP can handle more general systems of the form in which the vehicles have different control bounds and dynamics.

For this example, the target sets $L_i$ of the vehicles are circles of radius $r$ in the position space; each vehicle is trying to reach some desired set of positions. In terms of the state space $x_i$, the target sets are defined as $L_i = \{x_i : \| p_i - c_i \| \leq r \}$, where $c_i$ are centers of the target circles. For the simulation of the basic STP algorithm, we used $r = 0.1$. The vehicles have target centers $c_i$, initial conditions $x_i^0$, and scheduled times of arrivals $t_i^{STA}$ as follows:

$$c_1 = (0.7, 0.2), \quad x_1^0 = (-0.5, 0, 0), \quad t_1^{STA} = 0$$
$$c_2 = (-0.7, 0.2), \quad x_2^0 = (0.5, 0, \pi), \quad t_2^{STA} = 0.2$$
$$c_3 = (0.7, -0.7), \quad x_3^0 = (-0.6, 0.6, 7\pi/4), \quad t_3^{STA} = 0.4$$
$$c_4 = (-0.7, -0.7), \quad x_4^0 = (0.6, 0.6, 5\pi/4), \quad t_4^{STA} = 0.6 \tag{15}$$

The setup for this example is shown in Fig. 1, which also shows the static obstacles as the black rectangles around the center of the domain. The joint state space of this four-vehicle system is 12-D, making the joint trajectory planning...
This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY

Fig. 2. BRSs at $t = t_{LDT}^i$ for vehicles 1, 2, 3, 4, sliced at initial headings $\theta_0^i$. Black arrows indicate direction of obstacle motion. Due to the turn rate constraint, the presence of static obstacles $O_{static}^i$ and time-varying obstacles induced by higher priority vehicles $O_{j}^i(t)$ carve “channels” in the BRS, dividing it up into multiple “islands.”

and collision avoidance problem intractable for direct analysis using HJ reachability. Therefore, we apply the STP algorithm described in Algorithm 1 and repeatedly solve the double-obstacle HJ VI in (5) to obtain the optimal control for each vehicle to reach its target while avoiding higher priority vehicles. In addition, due to the flexibility of the HJ VI with respect to time-varying systems, the different scheduled times of arrival $t_{STA}^i$ can be trivially incorporated.

Figs. 2–4 show the simulation results. Since the state space of each vehicle is 3-D, each of the BRSs $\mathcal{V}_{basic}^i(t, t_{STA}^i)$ is also 3-D. To visualize the results, we slice the BRSs at the initial heading angles $\theta_0^i$. Fig. 2 shows the 2-D BRS slices for each vehicle at its latest departure times $t_{LDT}^i = -1.12, -0.94, -1.48,$ and $-1.44$ determined from our method. The obstacles in the domain $O_{static}^i$ and the obstacles induced by higher priority vehicles $O_{j}^i(t)$ inhibit the evolution of the BRSs, carving out thin “channels” that separate the BRSs into different “islands.” One can see how these “channels” and “islands” form by examining the time evolution of the BRS, shown in Fig. 3 for vehicle $Q_3$.

Finally, Fig. 4 shows the resulting trajectories of the four vehicles. Most interestingly, the subplot labeled $t = -0.55$ shows all four vehicles in close proximity without collision: each vehicle is outside of the danger zone of all other vehicles (although the danger zones may overlap). This close proximity is an indication of the optimality of the basic STP algorithm given the assigned priority ordering. Since no disturbances are present, getting as close to other vehicles’ danger zones as possible without entering the danger zones intuitively results in short transit times.

The actual arrival times of vehicles $Q_i, i = 1, 2, 3, 4$ are 0, 0.19, 0.34, 0.31, respectively. It is interesting to note that for some vehicles, the actual arrival times are earlier than
the scheduled times of arrivals $t_i^{STA}$. This is because in order to arrive at the target by $t_i^{STA}$, these vehicles must depart early enough to avoid major delays resulting from the induced obstacles of other vehicles; these delays would have led to a late arrival if vehicle $Q_i$ departed after $t_i^{LDT}$.

Computations were done on a desktop computer with a Core i7 5820K processor and two GeForce GTX Titan X graphics processing units. The average computation time per vehicle is approximately 1 s using CUDA and GPU parallelization. Note that for the simulations in this and subsequent sections, almost all of the computation is done offline. Only a lookup table query to obtain the gradient of the value function is required. For control affine systems, controller synthesis amounts to evaluation of an analytic expression.

V. STP WITH DISTURBANCES AND INCOMPLETE INFORMATION

Disturbances and incomplete information significantly complicate STP. The main difference is that the vehicle dynamics satisfy (1) as opposed to (8). Committing to exact trajectories is, therefore, no longer possible, since the disturbance $d_i(\cdot)$ is a priori unknown. Thus, the induced obstacles $O_f^i(t)$ are no longer just the danger zones centered around positions.

A. Theory

We present three methods to address the above issues. The methods differ in terms of control policy information that is known to a lower priority vehicle and have their respective advantages and disadvantages depending on the situation. The three methods are as follows:

1) Centralized Control: A specific control strategy is enforced upon a vehicle; this can be achieved, for example, by some central agent such as an air traffic controller.

2) Least Restrictive Control: A vehicle is required to arrive at target on time, but has no other control policy restrictions. In this case, the least restrictive control can be safely assumed by lower priority vehicles.

3) Robust Trajectory Tracking: A vehicle declares a nominal trajectory which can be robustly tracked under disturbances.

In general, the above-mentioned methods can be used in combination in a single trajectory planning problem, with each vehicle independently having different control policies. Lower priority vehicles would then plan their trajectories while taking into account the control policy information known for each higher priority vehicle. For clarity, we will present each method as if all vehicles are using the same method of trajectory planning.

In addition, for simplicity of explanation, we will assume that no static obstacles exist. In the situations where static obstacles do exist, the time-varying obstacles $G_i(t)$ simply become the union of the induced obstacles $O_f^i(t)$ in (10) and the static obstacles. The material in this section is taken partially from [43].

1) Centralized Control: The highest priority vehicle $Q_1$ first plans its trajectory by computing the BRS (with $i = 1$)

$$
\gamma_{dstb}^i(t, t_i^{STA}) = \{y : \exists u_i(\cdot) \in U_i, \forall d_i(\cdot) \in D_i, x_i(t) \text{ satisfies (1),} \forall s \in [t_i^{STA}, t_i(t)], x_i(s) \notin G_i(s), x_i(t) = y \exists s \in [t_i^{STA}, t_i(t)], x_i(s) \in L_i\}.
$$

From the BRS, we can obtain the optimal control

$$
u_{dstb}^i(t, x_i) = \arg \min_{u_i \in U_i} \max_{d_i \in D_i} \lambda \cdot f_i(x_i, u_i, d_i).
$$

Here, as well as in the other two methods, the latest departure time $t_i^{LDT}$ is then given by arg sup, $x_i(t) \in \gamma_{dstb}^i(t, t_i^{STA})$.

If there is a central agent directly controlling each of the $N$ vehicles, then the control law of each vehicle can be enforced.

In this method, lower priority vehicles can safely assume that higher priority vehicles are applying the enforced control law. In particular, the optimal controller for getting to the target, $u_{dstb}^i(t, x_i)$, can be enforced. In this case, the dynamics of each vehicle becomes

$$\begin{align}
\dot{x}_i &= f_i^{sc}(t, x_i, d_i) = f_i(x_i, u_{dstb}^i(t, x_i), d_i) \\
d_i &\in D_i, \quad i = 1, \ldots, N, \quad t \in [t_i^{LDT}, t_i^{STA}] 
\end{align}
$$

where $u_i$ no longer appears explicitly in the dynamics.

From the perspective of a lower priority vehicle $Q_j$, a higher priority vehicle $Q_{j'}$, $j < i$, induces a time-varying obstacle that represents the positions that could possibly be within the collision radius $R_c$ of $Q_j$ under the dynamics $f_j^{sc}(x_j, t_j, d_j)$. Determining this obstacle involves computing an FRS of $Q_j$ starting from $x_j(t_j^{LDT}) = x_j^0$. The FRS $\mathcal{W}_j^{sc}(t_j, t_j^{LDT}, t)$ is defined as follows:

$$\begin{align}
\mathcal{W}_j^{sc}(t_j, t_j^{LDT}, t) = \{y : \exists d_j(\cdot) \in D_j, x_j(\cdot) \text{ satisfies (19),} \\
x_j(t_j^{LDT}) = x_j^0, x_j(t) = y \}. 
\end{align}
$$

This FRS can be computed using (6) with the Hamiltonian

$$H_j^{cc}(t, x_j, \lambda) = \max_{d_j \in D_j} \lambda \cdot f_j^{cc}(t, x_j, d_j).
$$

The FRS $\mathcal{W}_j^{sc}(t_j, t_j^{LDT}, t)$ represents the set of possible states at time $t$ of a higher priority vehicle $Q_j$ given all possible disturbances $d_j(\cdot)$ and gives that $Q_j$ uses the feedback controller $u_{dstb}^j(t, x_j)$. In order for a lower priority vehicle $Q_i$ to guarantee that it does not go within a distance of $R_c$ to $Q_j$, $Q_i$ must stay a distance of at least $R_c$ away from the FRS $\mathcal{W}_j^{sc}(t_j, t_j^{LDT}, t)$ for all possible values of the nonposition states $h_j$. This gives the obstacle induced by a higher priority vehicle $Q_j$ for a lower priority vehicle $Q_i$ as follows:

$$O_f^i(t) = \{x_i : \exists y \in \mathcal{P}_j(t), \|p_i - y\| \leq R_c\} 
$$

In practice, we define the target set to be a small region around the vehicle’s initial state for computational reasons.
Algorithm 2 Centralized Control Algorithm

Given initial conditions $x_0^i$, vehicle dynamics (1), target set $L_i$, and static obstacles $O_i^{static}$, $i = 1,\ldots,N$, for each $i$,

1) determine the total obstacle set $G_i(t)$, given in (9). In the case $i = 1$, $G_i(t) = O_i^{static}$;
2) compute the BRS $\mathcal{Y}^{dist}_i(t, t_i^{STA})$ defined in (16). The latest departure time $t_i^{LDT}$ is then given by $\arg\sup_j x_j^0 \in \mathcal{Y}^{dist}_i(t, t_i^{STA})$;
3) compute the optimal control $u_i^{dist}(t, x_i)$ corresponding to $\mathcal{Y}^{dist}_i(t, t_i^{STA})$ given by (18). Given $u_i^{dist}(t, x_i)$, compute the FRS $\mathcal{W}_i^{fsc}(t_i^{LDT}, t)$ in (20);
4) finally, compute the induced obstacles $O_i^k(t)$ for each $k > i$ in the centralized control method, $O_i^k(t)$ is computed using (22) where $P_i(t)$ is given by (23).

where the set $P_i(t)$ is the set of states in the FRS $\mathcal{W}_i^{fsc}(t_i^{LDT}, t)$ projected onto the states representing position $p_j$, and disregarding the nonposion dimensions $h_j$:

\[
P_i(t) = \{ p_j : \exists h_j, (p_j, h_j) \in B_j(t) \}
\]

\[
B_j(t) = \mathcal{W}_j^{fsc}(t_i^{LDT}, t).
\]

Finally, taking the union of the induced obstacles $O_i^j(t)$ as in (9) gives us the time-varying obstacles $G_i(t)$ needed to define and determine the BRS $\mathcal{Y}^{dist}_j(t, t_i^{STA})$ in (16). Repeating this process, all vehicles will be able to plan trajectories that guarantee the vehicles’ timely and safe arrival. The centralized control algorithm can be summarized in Algorithm 2.

2) Least Restrictive Control: Here, we again begin with the highest priority vehicle $Q_1$ planning its trajectory by computing the BRS $\mathcal{Y}^{dist}_j(t, t_i^{STA})$ in (16). However, if there is no centralized controller to enforce the control policy for higher priority vehicles, weaker assumptions must be made by the lower priority vehicles to ensure collision avoidance. One reasonable assumption is that all higher priority vehicles follow the least restrictive control that would take them to their targets. This control would be given by $u_j^{fsc}(t, x_j)$

\[
\begin{cases}
u_j^{dist}(t, x_j) \text{ in (18)} & \text{if } x_j(t) \in \partial \mathcal{Y}_j^{dist}(t_i^{STA}), \\
U_j & \text{otherwise}
\end{cases}
\]

Such a controller allows each higher priority vehicle to use any controller it desires, except when it is on the boundary of the BRS, $\partial \mathcal{Y}_j^{dist}(t_i^{STA})$, in which case the optimal control $u_j^{dist}(t, x_j)$ given by (18) must be used to get to the target safely and on time. This assumption is the weakest assumption that could be made by lower priority vehicles given that the higher priority vehicles will get to their targets on time.

Suppose a lower priority vehicle $Q_i$ assumes that higher priority vehicles $Q_j, j < i$, use the least restrictive control strategy $u_j^{fsc}(t, x_j)$ in (25). From the perspective of the lower priority vehicle $Q_i$, a higher priority vehicle $Q_j$ could be in any state that is reachable from $Q_i$’s initial state $x_j(t_j^{LDT}) = x_j^0$ and from which the target $L_j$ can be reached. Mathematically, this is defined by the intersection of an FRS $\mathcal{W}_j^{fsc}(t_j^{LDT}, t)$ from the initial state $x_j(t_j^{LDT}) = x_j^0$ and the BRS $\mathcal{Y}_j^{dist}(t, t_i^{STA})$ defined in (16) from the target set $L_j$, $\mathcal{W}_j^{fsc}(t_j^{LDT}, t) \cap \mathcal{Y}_j^{dist}(t_i^{STA}, t)$. In this situation, since $Q_i$ cannot be assumed to be using any particular feedback control, $\mathcal{W}_j^{fsc}(t_j^{LDT}, t)$ is defined as

\[
\mathcal{W}_j^{fsc}(t_j^{LDT}, t) = \{ y : \exists x_j(t) \in P_j(t), \|p_i - y\|_2 \leq R_i \}
\]

Algorithm 3 Least Restrictive Control Algorithm

Given initial conditions $x_0^i$, vehicle dynamics (1), target set $L_i$, and static obstacles $O_i^{static}$, $i = 1,\ldots,N$, for each $i$,

1) determine the total obstacle set $G_i(t)$, given in (9). In the case $i = 1$, $G_i(t) = O_i^{static}$;
2) compute the BRS $\mathcal{Y}_i^{dist}(t, t_i^{STA})$ defined in (16). The latest departure time $t_i^{LDT}$ is then given by $\arg\sup_j x_j^0 \in \mathcal{Y}_i^{dist}(t, t_i^{STA})$;
3) compute the FRS $\mathcal{W}_i^{fsc}(t_i^{LDT}, t)$ in (26). Given $\mathcal{W}_j^{fsc}(t_j^{LDT}, t)$ and $\mathcal{Y}_i^{dist}(t, t_i^{STA})$, compute the positions that $Q_i$ could be in. The set of these positions is given by (29);
4) compute the induced obstacles $O_i^k(t)$ for each $k > i$ using (28).

Finally, taking the union of the induced obstacles $O_i^j(t)$ as in (9) gives us the space-varying obstacles $G_i(t)$ needed to define and determine the BRS $\mathcal{Y}_i^{dist}(t, t_i^{STA})$ in (16). Repeating this process, all vehicles will be able to plan trajectories that guarantee the vehicles’ timely and safe arrival. The centralized control algorithm can be summarized in Algorithm 2.

2) Least Restrictive Control: Here, we again begin with the highest priority vehicle $Q_1$ planning its trajectory by computing the BRS $\mathcal{Y}_i^{dist}(t, t_i^{STA})$ in (16). However, if there is no centralized controller to enforce the control policy for higher priority vehicles, weaker assumptions must be made by the lower priority vehicles to ensure collision avoidance. One reasonable assumption is that all higher priority vehicles follow the least restrictive control that would take them to their targets. This control would be given by $u_j^{fsc}(t, x_j)$

\[
\begin{cases}
u_j^{dist}(t, x_j) \text{ in (18)} & \text{if } x_j(t) \in \partial \mathcal{Y}_j^{dist}(t_i^{STA}), \\
U_j & \text{otherwise}
\end{cases}
\]

Such a controller allows each higher priority vehicle to use any controller it desires, except when it is on the boundary of the BRS, $\partial \mathcal{Y}_j^{dist}(t_i^{STA})$, in which case the optimal control $u_j^{dist}(t, x_j)$ given by (18) must be used to get to the target safely and on time. This assumption is the weakest assumption that could be made by lower priority vehicles given that the higher priority vehicles will get to their targets on time.

Suppose a lower priority vehicle $Q_i$ assumes that higher priority vehicles $Q_j, j < i$, use the least restrictive control strategy $u_j^{fsc}(t, x_j)$ in (25). From the perspective of the lower priority vehicle $Q_i$, a higher priority vehicle $Q_j$ could be in any state that is reachable from $Q_i$’s initial state $x_j(t_j^{LDT}) = x_j^0$ and from which the target $L_j$ can be reached. Mathematically, this is defined by the intersection of an FRS $\mathcal{W}_j^{fsc}(t_j^{LDT}, t)$ from the initial state $x_j(t_j^{LDT}) = x_j^0$ and the BRS $\mathcal{Y}_j^{dist}(t, t_i^{STA})$ defined in (16) from the target set $L_j$, $\mathcal{W}_j^{fsc}(t_j^{LDT}, t) \cap \mathcal{Y}_j^{dist}(t_i^{STA}, t)$. In this situation, since $Q_i$ cannot be assumed to be using any particular feedback control, $\mathcal{W}_j^{fsc}(t_j^{LDT}, t)$ is defined as

\[
\mathcal{W}_j^{fsc}(t_j^{LDT}, t) = \{ y : \exists x_j(t) \in P_j(t), \|p_i - y\|_2 \leq R_i \}
\]

This FRS can be computed using (6) with the Hamiltonian

\[
H_j^{fsc}(x_j, \lambda) = \max_{u_j \in U_j} \lambda \cdot f_j(x_j, u_j, d_j).
\]

In turn, the obstacle induced by a higher priority $Q_j$ for a lower priority vehicle $Q_i$ is as follows:

\[
O_i^j(t) = \{ x_i : \exists y \in P_j(t), \|p_i - y\|_2 \leq R_i \}
\]

where

\[
P_j(t) = \{ p_j : \exists h_j, (p_j, h_j) \in B_j(t) \}
\]

\[
B_j(t) = \mathcal{W}_j^{fsc}(t_j^{LDT}, t) \cap \mathcal{Y}_j^{dist}(t, t_i^{STA}).
\]

The least restrictive control method can be summarized in Algorithm 3.

Remark 1: The centralized control method can be thought of as the “most restrictive control” method, in which all vehicles must use the optimal controller at all times, while the least restrictive control method allows vehicles to use any suboptimal controller that allows them to arrive at the target on time. These two methods can be considered two extremes of a spectrum in which varying degrees of optimality are assumed for higher priority vehicles. Vehicles can also choose a control strategy in the middle of the two extremes and, for example, use a control within some range around the optimal control or use the optimal control unless some condition is met. The induced obstacles and the BRS can then be similarly computed using the corresponding control strategy.

3) Robust Trajectory Tracking: Even though it is impossible to commit to and track an exact trajectory in the presence of disturbances, it may still be possible to robustly track a feasible nominal trajectory with a bounded error at all times.
The tracking error bound can be used to determine the induced obstacles. Here, computation is done in two phases: the planning phase and the disturbance rejection phase. In the planning phase, we compute a nominal trajectory \( x_{r,j}(\cdot) \) that is feasible in the absence of disturbances. In the disturbance rejection phase, we compute a bound on the tracking error.

It is important to note that the planning phase does not make full use of a vehicle’s control authority, as some margin is needed to reject unexpected disturbances while tracking the nominal trajectory. Therefore, in this method, planning is done for a reduced control set \( \mathcal{U}^p \subset \mathcal{U} \). The resulting trajectory reference will not utilize the vehicle’s full control capability; additional maneuverability is available at execution time to counteract external disturbances.

In the disturbance rejection phase, we determine the error bound independently of the nominal trajectory by finding a robust controlled-invariant set in the joint state space of the vehicle and a tracking reference that may “maneuver” arbitrarily in the presence of an unknown bounded disturbance. Taking a worst case approach, the tracking reference can be viewed as a virtual evader vehicle that is optimally avoiding the actual vehicle to enlarge the tracking error. We, therefore, can model trajectory tracking as a pursuit–evasion game in which the actual vehicle is playing against the coordinated worst case action of the virtual vehicle and the disturbance.

Let \( x_j \) and \( x_{r,j} \) denote the states of the actual vehicle \( Q_j \) and the virtual evader, respectively, and define the tracking error \( e_j = x_j - x_{r,j} \). When the error dynamics are independent of the absolute state as in (31) (and also \([25, (7)]\)), we can obtain error dynamics of the form

\[
\dot{e}_j = f(e_j, u_j, u_{r,j}, d_j),
\]

\[
u_j \in \mathcal{U}_j, u_{r,j} \in \mathcal{U}_j^p, d_j \in \mathcal{D}_j, \quad t \leq 0. \tag{31}
\]

To obtain bounds on the tracking error, we first conservatively estimate the error bound around any reference state \( x_{r,j} \), denoted as \( \mathcal{E}_j \):

\[
\mathcal{E}_j = \{ e_j : \| p_{e_j} \|_2 \leq R_{EB} \}, \tag{32}
\]

where \( p_{e_j} \) denotes the position coordinates of \( e_j \) and \( R_{EB} \) is a design parameter. We next solve a reachability problem with its complement \( \mathcal{E}_j^c \), the set of tracking errors violating the error bound, as the target in the space of the error dynamics. From \( \mathcal{E}_j \), we compute the following BRS:

\[
\gamma_j^{EB}(t, 0) = \{ y : \forall u_j(\cdot) \in \mathcal{U}_j, \exists x_{r,j}(\cdot) \in \mathcal{U}_j^p, \exists d_j(\cdot) \in \mathcal{D}_j, e_j(\cdot) \text{ satisfies (31)}, e_j(t) = y, \exists s \in [t, 0], e_j(s) \in \mathcal{E}_j^c \} \tag{33}
\]

where the Hamiltonian to compute the BRS is given by:

\[
H_j^{EB}(e_j, \lambda) = \max_{u_j \in \mathcal{U}_j, u_{r,j} \in \mathcal{U}_j^p} \min_{d_j \in \mathcal{D}_j} \lambda \cdot f(e_j, u_j, u_{r,j}, d_j). \tag{34}
\]

Letting \( t \to -\infty \), we obtain the infinite-horizon controlled-invariant set \( \Omega_j := \lim_{t \to -\infty} (\gamma_j^{EB}(t, 0))^c \). If \( \Omega_j \) is nonempty, then the tracking error \( e_j \) at flight time is guaranteed to remain within \( \Omega_j \subseteq \mathcal{E}_j \) provided that the vehicle starts inside \( \Omega_j \) and subsequently applies the feedback control law

\[
\kappa_j(e_j) = \arg \max_{u_j \in \mathcal{U}_j, u_{r,j} \in \mathcal{U}_j^p} \min_{d_j \in \mathcal{D}_j} \lambda \cdot f(e_j, u_j, u_{r,j}, d_j). \tag{35}
\]

The induced obstacles by each higher priority vehicle \( Q_j \) can thus be obtained by

\[
\mathcal{O}_j^H(t) = \{ x_j : \exists y \in \mathcal{P}_j(t), \| p_i - y \|_2 \leq R_c \}
\]

\[
\mathcal{P}_j(t) = \{ p_j : \exists h_j, (p_j, h_j) \in \mathcal{B}_j(t) \}
\]

\[
\mathcal{B}_j(t) = \Omega_j + x_{r,j}(t), \tag{36}
\]

where the “+” in (36) denotes the Minkowski sum.\(^3\) Intuitively, if \( Q_j \) is tracking \( x_{r,j}(t) \), then it will remain within the error bound \( \Omega_j \) around \( x_{r,j}(t) \) \( \forall t \). This set is mathematically given by the “tube” obtained by augmenting the error bound \( \Omega_j \) around the reference trajectory \( x_{r,j}(\cdot) \). This is precisely the set \( \mathcal{B}_j(t) \) (and \( \mathcal{P}_j(t) \) for the position states). The induced obstacles can then be obtained by augmenting a danger zone around this set. Finally, we can obtain the total obstacle set by \( \Omega_j \):

\[
\tilde{\Gamma}_j(t) = \tilde{\Gamma}_j(t) + \Omega_j. \tag{37}
\]

Finally, given \( \Omega_j \), we can guarantee that \( Q_j \) will reach its target \( \mathcal{L}_j \) if \( \mathcal{Q}_j \subseteq \mathcal{L}_j \); thus, in the trajectory planning phase, we modify \( \mathcal{L}_j \) to be \( \tilde{\mathcal{L}}_j := \{ x_j : \Omega_j + x_j \subseteq \mathcal{L}_j \} \) and compute a BRS, with the control authority \( \mathcal{U}_j^p \), that contains the initial state of the vehicle. Mathematically

\[
\gamma_j^{ST}(t, t_j^{STA}) = \{ y : \forall u_j(\cdot) \in \mathcal{U}_j^p, x_j(\cdot) \text{ satisfies (8),} \}
\]

\[
\forall s \in [t, t_j^{STA}], x_j(s) \notin \tilde{\Gamma}_j(t), \exists s \in [t, t_j^{STA}], x_j(s) \in \tilde{\mathcal{L}}_j, x_j(t) = y. \tag{38}
\]

The BRS \( \gamma_j^{ST}(t, t_j^{STA}) \) can be obtained by solving (5) using the Hamiltonian

\[
H_j^{ST}(x_j(\cdot), \lambda) = \min_{u_j \in \mathcal{U}_j^p} \lambda \cdot f_j(x_j(\cdot), u_j). \tag{39}
\]

The corresponding optimal control for reaching \( \tilde{\mathcal{L}}_j \) is

\[
u_j^{ST}(t) = \arg \min_{u_j \in \mathcal{U}_j^p} \lambda \cdot f_j(x_j(\cdot), u_j). \tag{40}
\]

The nominal trajectory \( x_{r,j}(\cdot) \) can thus be obtained by using vehicle dynamics (8), with the optimal control \( u_j^{ST}(\cdot) \) given by (40). From the resulting nominal trajectory \( x_{r,j}(\cdot) \), the overall control policy to reach \( \mathcal{L}_j \) can be obtained via (35).

Note that in this paper, the reduced control authority \( \mathcal{U}_j^p \subset \mathcal{U} \) and the parameter \( R_{EB} \) are selected by trial and error such that the tracking error bound \( \Omega_j \) is nonempty.

\(^3\)The Minkowski sum of sets \( A \) and \( B \) is the set of all points that are the sum of any point in \( A \) and \( B \).
Algorithm 4 Robust Trajectory Tracking Algorithm

Given initial conditions \( x_i^0 \), vehicle dynamics (1), target sets \( \mathcal{L}_i \), and static obstacles \( O_i^{\text{static}} \), \( i = 1, \ldots, N \), for each \( i \),
1) determine the total obstacle set \( G_i(t) \), given in (9). In the case \( i = 1, G_i(t) = O_i^{\text{static}} \) \( \forall t \);
2) decide on a reduced control authority \( \mathcal{U}_i^p \) for the planning phase, and choose a parameter \( R_{\text{EBB}} \) to conservatively bound the tracking error;
3) compute the BRS \( V_i^{\text{BB}}(t, 0) \) using (33) and make sure that \( \Omega_i \neq \emptyset \). Given \( R_{\text{EBB}} \), the error bound on the tracking error is given by \( \Omega_i \);
4) using \( \Omega_i \), determine the augmented obstacle set \( \bar{G}_i(t) \), given in (37);
5) compute the BRS \( V_i^{\text{MT}}(t, \tau^{\text{STA}}) \) as described in (38) using the reduced target set \( \mathcal{L}_i \), \( \bar{G}_i(t) \) as obstacles, and the control authority \( \mathcal{U}_i^p \). The latest departure time \( t_i^{\text{LDT}} \) is then given by \( \text{arg max}_t x_i^0 \in V_i^{\text{MT}}(t, \tau^{\text{STA}}) \);
6) compute the nominal trajectory \( x_{r,i}(\cdot) \) for \( Q_i \) in the absence of disturbances, which can be obtained using the vehicle dynamics in (8) and optimal control given in (40);
7) the induced obstacles \( O_k^i(t) \) for each \( k > i \) can be computed using \( \Omega_i \) and \( x_{r,i}(\cdot) \) via (36).

Nevertheless, selecting these parameters in a principled way is an interesting future direction.

The robust trajectory tracking method can be summarized in Algorithm 4.

B. Numerical Simulations

We demonstrate our proposed methods for accounting for disturbances and incomplete information using a four-vehicle example. Each vehicle has the simple kinematics model in (14) but with disturbances added to the evolution of each state

\[
\begin{align*}
\dot{x}_i &= v_i \cos \theta_i + d_{x,i} \\
\dot{y}_i &= v_i \sin \theta_i + d_{y,i} \\
\dot{\theta}_i &= \omega_i + d_{\theta,i} \\
\ddot{\theta}_i &= \omega_i \\
\| (d_{x,i}, d_{y,i}) \|_2 &\leq \delta_r, \quad | \omega_i | &\leq \bar{\omega}, \\
\end{align*}
\]

(41)

where \( d = (d_{x,i}, d_{y,i}, d_{\theta,i}) \) represents \( Q_i \)'s disturbances in the three states. The control of \( Q_i \) is \( u_i = (v_i, \omega_i) \), where \( v_i \) is the speed of \( Q_i \) and \( \omega_i \) is the turn rate; both controls have a lower and upper bound. For illustration purposes, we choose \( v = 0.5, \bar{v} = 1, \bar{\omega} = 1 \); however, our method can easily handle the case in which these inputs differ across vehicles. The disturbance bounds are chosen as \( \delta_r = 0.1, \delta_{\theta} = 0.2 \), which correspond to a 10% uncertainty in the dynamics.

For this example, we have chosen scheduled times of arrival \( \tau_i^{\text{STA}} = 0 \) \( \forall i \) for simplicity. Each vehicle aims to get to a target set of radius \( r = 0.1 \). The vehicles’ target centers \( x_i^1 \) and initial conditions \( x_i^0 \) are given by (15). These parameters are the same as the example in Section IV-B, except that the \( \tau_i^{\text{STA}} \) values are the same for all vehicles and that there are no static obstacles. The problem setup for this example is shown in Fig. 5.

With the above parameters, we obtain \( t_i^{\text{LDT}}, i = 1, 2, 3, 4 \). Note that even though \( \tau_i^{\text{STA}} \) is assumed to be same for all vehicles in this example for simplicity, our method can easily handle the case in which \( \tau_i^{\text{STA}} \) is different for each vehicle as we have already shown in Section IV-B.

For each proposed method of computing induced obstacles, we show the vehicles’ entire trajectories (colored dotted lines), and overlay their positions (colored asterisks) and headings (arrows) at a point in time in which they are in a relatively dense configuration. In all the cases, the vehicles are able to avoid each other’s danger zones (colored dashed circles) while getting to their target sets in minimum time. In addition, we show the evolution of the BRS over time for \( Q_3 \) (green boundaries) as well as the obstacles induced by the higher priority vehicles (black boundaries). Offline computations were done on a desktop computer with a Core i7 5820K processor and two GeForce GTX Titan X graphics processing units. The average computation time per vehicle is approximately 2 s using CUDA and GPU parallelization.

1) Centralized Control: Fig. 6 shows the simulated trajectories in the situation where a centralized controller enforces each vehicle to use the optimal controller \( u_{i1}^{\text{opt}}(t, x_i) \) according to (18), as described in Section V-A1. In this case, vehicles appear to deviate slightly from a straight line trajectory toward their respective targets, just enough to avoid higher priority vehicles. The deviation is small since the centralized controller is quite restrictive, making the possible positions of higher priority vehicles cover a small area. In the dense configuration at \( t = -1.0 \), the vehicles are close to each other but still outside each other’s danger zones.
Fig. 7. Evolution of the BRS and the obstacles induced by $Q_1$ and $Q_2$ for $Q_3$ in the centralized control method. Since vehicles apply the optimal control at all times, the obstacle sizes are only slightly bigger than those in Figs. 2 and 3.

Fig. 8. Simulated trajectories in the least restrictive control method. All vehicles start moving before $Q_1$ starts, because the large obstacles make it optimal to wait until higher priority vehicles pass by, leading to earlier $t_{LDT}$ values.

Fig. 9. Evolution of the BRS for $Q_3$ in the least restrictive control method. In this case, $t_{LDT} = -2.66$, significantly earlier than that in the centralized control method ($-1.94$), reflecting the impact of larger induced obstacles.

Fig. 10 shows vehicle trajectories in the situation where each vehicle robustly tracks a prespecified trajectory and is guaranteed to stay inside a “bubble” around the trajectory. Fig. 11 shows the evolution of the BRS and induced obstacles for vehicle $Q_3$. The obstacles induced by other vehicles inhibit

by $Q_1$, since lower priority vehicles make the weak assumption that higher priority vehicles are using the least restrictive control. Because the induced obstacles are so large, it is faster for lower priority vehicles to wait until higher priority vehicles pass by than to move around higher priority vehicles. As a result, vehicles never form a dense configuration, and their trajectories are relatively straight, indicating that vehicles take a short path to the target after higher priority vehicles pass by. This is also indicated by the early $t_{LDT}$ values for all vehicles, except $Q_1$, the highest priority vehicle, since it need not account for any moving obstacles.

From $Q_3$’s (green) perspective, the large obstacles induced by $Q_1$ and $Q_2$ are shown in Fig. 9 as the black boundary. As the BRS (green boundary) evolves over time, its growth gets inhibited by the large obstacles for a long time, from $t = -0.89$ to $t = -1.39$. Eventually, the boundary of the BRS reaches the initial state of $Q_3$ at $t = t_{LDT} = -2.66$.

3) Robust Trajectory Tracking: In the planning phase, we reduced the maximum turn rate of the vehicles from 1 to 0.6 and the speed range from [0.5, 1] to exactly 0.75 (constant speed). With these reduced control authorities, we determined from the disturbance rejection phase that a nominal trajectory from the planning phase can be robustly tracked within a distance of $R_{EB} = 0.075$.

Fig. 10 shows vehicle trajectories in the situation where each vehicle robustly tracks a prespecified trajectory and is guaranteed to stay inside a “bubble” around the trajectory. Fig. 11 shows the evolution of the BRS and induced obstacles for vehicle $Q_3$. The obstacles induced by other vehicles inhibit
VI. STP WITH AN INTRUDER

In Section V, we made the basic STP algorithm more robust by taking into account disturbances and considering situations, in which vehicles may not have complete information about other vehicles’ control strategies. However, if a vehicle not in the set of STP vehicles enters the system, or even worse, if this vehicle is an adversarial intruder, the original plan can lead to vehicles entering into another vehicle’s danger zone. Without an additional safety margin that takes a potential intruder into account, a vehicle trying to avoid the intruder may effectively become an intruder itself, leading to a domino effect. In this section, we propose a method to allow vehicles to avoid an intruder while maintaining the STP structure.

A. Theory

In general, the effect of an intruder on the vehicles in structured flight can be entirely unpredictable, since the intruder in principle could be adversarial in nature, and the number of intruders could be arbitrary. Therefore, for our analysis to produce reasonable results, two assumptions about the intruders must be made.

Assumption 1: At most one intruder (denoted as $Q_I$) affects the STP vehicles at any given time. The intruder is removed from the system after affecting the STP vehicles after a duration of $t^{IAT}$. The removal of the intruder can be done, for example, by forcing it out of the altitude range of the STP vehicles.

Let the time at which intruder appears in the system be $t$ and the time at which it disappears be $t$. Assumption 1 implies that $t \leq t + t^{IAT}$. Thus, any vehicle $Q_i$ would need to avoid the intruder $Q_I$ for a maximum duration of $t^{IAT}$. This assumption can be valid in situations where intruders are rare and that some fail safe or enforcement mechanism exists to force the intruder out of the altitude level affecting the STP vehicles.

The removal of the intruder can be done, for example, by forcing it out of the altitude level affecting the STP vehicles. The removal of the intruder can be done, for example, by forcing it out of the altitude level affecting the STP vehicles.

Assumption 2: The dynamics of the intruder are known and given by $\dot{x}_i = f_i(x_i, u_i, d_i)$. Assumption 2 is required for the HJ reachability analysis. Even though assumption 2 often does not hold for practical systems, a conservative model of the intruder may be used in situations, where the dynamics of the intruder are not known exactly. Note that we only assume that the dynamics of the intruder are known, but its initial state $x_0$, control $u$, and disturbance $d$ experiences are unknown.

Based on the above-mentioned assumptions, we aim to design a control policy that ensures for each STP vehicle separation with the intruder and with other STP vehicles and successful transit to the destination. However, depending on the initial state of the intruder, its control policy, and the disturbances in the dynamics of a vehicle and the intruder, a vehicle may arrive at different states after avoiding the intruder. Therefore, a control policy that ensures a successful transit to the destination needs to account for all such possible states, which is a trajectory planning problem with multiple (infinite,
to be precise) initial states and a single destination, and is hard to solve in general.

Thus, we divide the intruder avoidance problem into two subproblems. 1) We first design a control policy that ensures a successful transit to the destination if no intruder appears or successful avoidance of the intruder if it does appear. 2) After the intruder is removed from the system at $\bar{t}$, we solve a new STP problem (that is, we “replan”) for vehicles which needed to avoid the intruder. In this case, the affected vehicles will replan as lowest priority vehicles starting from the initial they happen to arrive at after avoiding the intruder. Suppose some vehicle $Q_i$ starts avoiding the intruder $Q_I$ at some time $t = \underline{t}$ and stops avoiding at $t = \bar{t}$. When $t < \underline{t}$, $Q_i$ must plan its trajectory taking into account the possibility that it may need to avoid an intruder $Q_I$. Since $Q_i$ may spend a duration of up to $I_{AT}$ performing avoidance, its induced obstacles $O_i^j(k), k > i$ need to be computed in a way that reflects this possibility. The induced obstacles computation is discussed in Section VI-A1.

We must also ensure that while avoiding the intruder, $Q_i$ does not collide with the total obstacle set $G_i(t)$. This requires computing the augmented total obstacle set $\hat{G}_i(t)$; the computation of $\hat{G}_i(t)$ and the controller that guarantees the avoidance of the augmented obstacles are discussed in Section VI-A2.

In Section VI-A3, we describe how $Q_i$ can guarantee collision avoidance with the intruder. A pairwise collision avoidance problem such as this has been solved in isolation in [25]. Finally, when $t > \bar{t}$, $Q_i$ has already successfully avoided the intruder, but depending on the state it happens to arrive at after avoiding the intruder, it may need to replan its trajectory to reach the target safely. The replanning process is discussed in Section VI-A4.

1) Induced Obstacle Computation: The goal of this section is to compute, for each lower priority vehicle $Q_i$, the time-varying obstacles induced by each higher priority vehicle $Q_j, j < i$, denoted by $O_i^j(t)$. As before, the total obstacle set $G_i(t)$ can then be obtained using (9). To compute the obstacle that $Q_i$ needs to avoid at time $\tau$, it is sufficient to consider the scenarios, where $\underline{t} \in [t - I_{AT}, t]$. This is because if $\underline{t} < t - I_{AT}$, then the STP vehicles would already be in the replanning phase at time $t$ and hence cannot be in conflict.

Depending on the information known to a lower priority vehicle $Q_i$ about $Q_j$’s control strategy, we can use one of the three methods described in Section V to compute the “base” obstacles $B_i(t)$; these are the obstacles that would have been induced by $Q_j$ in the absence of an intruder. The base obstacles are given by (24), (30), and (36), respectively, for the centralized control, least restrictive control, and robust trajectory tracking methods.

The induced obstacles, $O_i^j(t)$, are then given by the states that $Q_j$ can reach while avoiding the intruder, starting from some state in $B_i(t), \underline{t} \in [t - I_{AT}, t]$. These states can be obtained by computing an FRS from the base obstacles

$\mathcal{W}_j^O(t - \tau, t)$ represents the set of all possible states that $Q_j$ can reach after a duration of $\tau$ starting from inside $B_j(t - \tau)$. This FRS can be obtained by solving the Hamiltonian $H_j^O(x_j, \lambda) = \max_{u_j \in U_j} \max_{d_j \in D_j} \lambda \cdot f_j(x_j, u_j, d_j)$. (43)

Since $\tau \in [0, I_{AT}]$, the induced obstacles can be obtained as

$O_i^j(t) = \{x_j : \exists y \in \mathcal{P}_j(t), \|p_i - y\|_2 \leq R_i\}$

$P_j(t) = \{p_j : \exists h_j, (p_j, h_j) \in \bigcup_{\tau \in [0, I_{AT}]} \mathcal{W}_j^O(t - \tau, t)\}$ (44)

Note that by the definition of base obstacles, $B_i(t + t_2) \subseteq \mathcal{W}_j^{BO}(t + t_1, t + t_2) \forall t, t_2 > t_1$, where $\mathcal{W}_j^{BO}(t + t_1, t + t_2)$ denotes the FRS of $B_j(t + t_1)$ computed for a duration of $t_2 - t_1$. Therefore, we have that $\mathcal{W}_j^O(t - \tau, t) \subseteq \mathcal{W}_j^O(t - I_{AT}, t)$ $\forall \tau \in [0, I_{AT}]$. Thus, $P_j(t)$ in (44) can be equivalently written as

$P_j(t) = \{p_j : \exists h_j, (p_j, h_j) \in \mathcal{W}_j^O(t - I_{AT}, t)\}$. (45)

2) Augmented Obstacle Computation: We next need to ensure that $Q_i$ does not collide with the obstacles $\hat{G}_i(\cdot)$ computed in Section VI-A1 even when it is avoiding the intruder. In particular, we can compute a region around the obstacles $\hat{G}_i(\cdot)$ such that for all disturbances, $Q_i$ can avoid colliding with obstacles for $I_{AT}$ seconds regardless of its avoidance control, if $Q_i$ starts outside this region. Augmenting $\hat{G}_i(\cdot)$ with this region gives us the augmented obstacles, $\hat{G}_i(\cdot)$, that can then be used during the trajectory planning of $Q_i$ to ensure collision avoidance with $\hat{G}_i(\cdot)$.

Suppose that the intruder appears in the system at some time $\underline{t} = t - I_{AT} + \tau, \tau \in [0, I_{AT}]$. In this case, we need to ensure that $Q_i$ does not collide with the obstacle $\hat{G}_i(t + \tau)$ at time $t + \tau$, regardless of its control $u_i(s)$ and disturbance $d_i(s)$ for the time interval $s \in [t, t + \tau]$. It is, therefore, sufficient to avoid the $\tau$-horizon BRS of $\hat{G}_i(t + \tau)$ at time $t$. This argument applies for all $\tau \in [0, I_{AT}]$. Mathematically

$\hat{G}_i(t) = \bigcup_{\tau \in [0, I_{AT}]} \mathcal{V}_i^G(t, t + \tau)$ (46)

where $\mathcal{V}_i^G(t, t + \tau)$ represents BRS of $\hat{G}_i(t + \tau)$ computed backward for $\tau$ seconds. Formally

$\mathcal{V}_i^G(t, t + \tau) = \{y : \exists u_i(\cdot) \in U_i, \exists d_i(\cdot) \in D_i, x_i(\cdot) \text{ satisfies (1), } x_i(t + \tau) = y, x_i(s) \in \hat{G}_i(s)\}$. (47)

The Hamiltonian $H_i^G$ to compute $\mathcal{V}_i^G(\cdot)$ is given by:

$H_i^G(x_i, \lambda) = \min_{u_i \in U_i} \min_{d_i \in D_i} \lambda \cdot f_i(x_i, u_i, d_i)$. (48)

Remark 2: Note that if we use the robust trajectory tracking method to compute the base obstacles, we would need to augment the obstacles in (46) by the error bound of $Q_i, \Omega_i$, as discussed in Section V-A3.
Finally, we compute a BRS $V_i^{\text{AO}}(t, t_i^{\text{STA}})$ for trajectory planning that contains the initial state of $Q_i$, while avoiding these augmented obstacles
\[
V_i^{\text{AO}}(t, t_i^{\text{STA}}) = \{y : \exists u_i(\cdot) \in U_i, \forall d_i(\cdot) \in D_i, x_i(\cdot) \text{ satisfies (1), } \forall s \in [t, t_i^{\text{STA}}], x_i(s) \notin G_i(s), \exists s \in [t, t_i^{\text{STA}}], x_i(s) \in L_i, x_i(t) = y\}.
\]

The Hamiltonian $H_i^{\text{AO}}$ to compute BRS in (49) is given by
\[
H_i^{\text{AO}}(x_i, \lambda) = \min_{u_i \in U_i, d_i \in D_i} \max \lambda \cdot f_i(x_i, u_i, d_i).
\]

Note that $V_i^{\text{AO}}(\cdot)$ ensures goal satisfaction for $Q_i$ in the absence of intruder. The goal satisfaction controller is given by
\[
u_i^{\text{AO}}(t, x_i) = \arg \min_{u_i \in U_i, d_i \in D_i} \max \lambda \cdot f_i(x_i, u_i, d_i).
\]

Moreover, if $Q_i$ starts within $V_i^{\text{AO}}$, it is guaranteed to avoid collision for a duration of $t_i^{\text{STA}}$, starting at any $\bar{t} < t_i^{\text{STA}}$, irrespective of the control and disturbance applied during this time period.

3) Optimal Avoidance Controller: First, we define relative dynamics of the intruder $Q_i$ with state $x_i$ with respect to $Q_i$ with state $x_i$
\[
x_{i,i} = x_i - x_i, \quad \dot{x}_{i,i} = f_r(x_{i,i}, u_i, u_i, d_i, d_i).
\]

Given the relative dynamics, we compute the set of states from which the joint states of $Q_i$ and $Q_i$ can enter the danger zone $Z_{i,i}$ despite the best efforts of $Q_i$ to avoid $Q_i$. This set of states is given by the BRS $V_i^{\text{CA}}(t, t_i^{\text{LAT}})$
\[
V_i^{\text{CA}}(t, t_i^{\text{LAT}}) = \{y : \forall u_i(\cdot) \in U_i, \exists \lambda I(\cdot) \in \tilde{U}_i, \exists d_i(\cdot) \in D_i, \exists s(\cdot) \in [t, t_i^{\text{LAT}}], x_i(\cdot) \text{ satisfies (52), } \exists s(\cdot) \in [t, t_i^{\text{LAT}}], x_i(s) \in \tilde{L}_i, x_i(t) = y\}
\]

where $\tilde{L}_i = \{x_{i,i} : \|p_{i,i}\|_2 \leq R_i\}$ and the Hamiltonian for computing this BRS is given by
\[
H_i^{\text{CA}}(x_{i,i}, \lambda) = \max_{u_i \in U_i} \left(\min_{d_i \in D_i, d_i \in D_i} \lambda \cdot f_i(x_{i,i}, u_i, u_i, d_i, d_i)\right).
\]

Once the value function $V_i^{\text{CA}}(t, x_{i,i})$ corresponding to the BRS $V_i^{\text{CA}}(t, t_i^{\text{LAT}})$ is computed, the optimal avoidance control $u_i^{\text{CA}}$ can be obtained as
\[
u_i^{\text{CA}}(t, x_i, x_i) = \arg \max_{u_i \in U_i} \left(\min_{d_i \in D_i, d_i \in D_i} \lambda \cdot f_i(x_{i,i}, u_i, u_i, d_i, d_i)\right).
\]

Under normal circumstances when the intruder $Q_i$ is far away, we have $V_i^{\text{CA}}(0, x_{i,i}) > 0$; as $Q_i$ gets closer to $Q_i$, $V_i^{\text{CA}}(0, x_{i,i})$ decreases. If $Q_i$ applies the control $u_i^{\text{CA}}$ when $V_i^{\text{CA}}(0, x_{i,i}) = 0$, then the collision avoidance between $Q_i$ and $Q_i$ is guaranteed for a duration of $t_i^{\text{LAT}}$ under the worst case intruder control strategy.

In addition, obstacle augmentation (46) ensures that $Q_i$ does not collide with $G_i(\cdot)$ during the avoidance maneuver. The overall control policy for avoiding the intruder and collision with other vehicles is thus given by
\[
u_i^\text{A}(t) = \begin{cases} u_i^{\text{AO}}(t, x_i) & t \leq \bar{t} \\ u_i^{\text{CA}}(t, x_i, x_i) & t \leq \bar{t} \leq t_i \end{cases}
\]

4) Replanning After Intruder Avoidance: After the intruder disappears, goal satisfaction controllers which ensure that the vehicles reach their destinations can be obtained by solving an STP problem as described in Section V, where the starting states of the vehicles are now given by the states they end up in, denoted $\bar{x}_i^0$, after avoiding the intruder. Let the optimal control policy corresponding to this goal satisfaction controller be denoted $u_i^\ast(t, x_i)$. The overall control policy that ensures intruder avoidance, collision avoidance with other vehicles, and successful transition to the destination is given by
\[
u_i^\ast(t) = \begin{cases} u_i^\ast(t, x_i) & t \leq \bar{t} \\ u_i^A(t, x_i) & t > \bar{t} \end{cases}
\]

Note that in order to replan using an STP method, we need to determine feasible $t_i^{\text{STA}}$ for all vehicles. This can be done by computing an FRS
\[
\n_{i}^{\text{RP}}(\bar{t}, t) = \{y \in \mathbb{R}^m : \exists u_i(\cdot) \in U_i, \exists d_i(\cdot) \in D_i, \exists s(\cdot) \in [\bar{t}, t], x_i(t) = y_i^0, x_i(\cdot) \text{ satisfies (52), } \exists s(\cdot) \in [\bar{t}, t], x_i(s) \in \tilde{G}_i(\cdot)\}
\]

where $x_i^0$ represents the state of $Q_i$ at $t = \bar{t}$; $G_i(\cdot)$ takes into account the fact that $Q_i$ needs to avoid higher priority vehicles $j, j < i$ and is defined in an analogous way to (9).

The FRS in (55) can be obtained by solving
\[
\max_{\mathbf{D}_i} \{\mathbf{D}_i \mathbf{W}_i(\bar{t}, x_i) + H_i^{\text{RP}}(\bar{t}, t, x_i, x_i, x_i, x_i)\}, \quad \n_{i}^{\text{RP}}(\bar{t}, t) = \max_{\mathbf{D}_i} \{\mathbf{D}_i \mathbf{W}_i(\bar{t}, x_i)\}
\]

where $\mathbf{W}_i$, $\mathbf{G}_i$, and $\mathbf{I}_i$ represent the FRS, obstacles during replanning, and the initial state of $Q_i$, respectively. The new $t_i^{\text{STA}}$ of $Q_i$ is now given by the earliest time at which $\n_{i}^{\text{RP}}(\bar{t}, t)$ intersects the target set $\tilde{L}_i, t_i^{\text{STA}} := \arg \min_{t_i} \{\n_{i}^{\text{RP}}(\bar{t}, t) \cap \tilde{L}_i \neq \emptyset\}$. Intuitively, this means that there exists a control policy which will steer the vehicle to its destination by that time, despite the worst case disturbance it might experience. Note that, in theory, $x_i^0$ could be such that it is impossible to reach the target region starting from $x_i^0$; however, for most practical systems target remains feasible given a long enough time horizon to reach the target.

Remark 3: Note that we only need to replan the trajectories of the vehicles that are affected by the intruder. In particular, if $V_i^{\text{CA}}(0, x_{i,i}(t)) > 0$ during the entire duration $t \in [\bar{t}, t]$ for a vehicle, then the vehicle would need not to apply any avoidance control, and hence replanning would not be required for this vehicle.

Remark 4: In general, an intruder can be present in the system for much longer than $t_i^{\text{LAT}}$, as long as it is not affecting...
Algorithm 5 Intruder Avoidance Algorithm (Offline Planning)

Given initial conditions $x_i^0$, vehicle dynamics (1), intruder dynamics in Assumption 2, target sets $L_i$, and static obstacles $O_i^{\text{static}}, i = 1, \ldots, N$, for each $i$,

1. determine the total obstacle set $G_i(t)$, given in (9). In the case $i = 1$, $G_i(t) = O_i^{\text{static}} \cup t$;
2. compute the augmented obstacle set $\tilde{G}_i(t)$ given by (46), where $V_{\pi}^0(0, \tau)$ is given by (47);
3. given $\tilde{G}_i(t)$, compute the BRS $V_{\pi}^{\text{ST}}(t, i^{\text{STA}})$ defined in (49);  
4. the optimal control to avoid the intruder can be obtained by computing $V_{\pi}^{\text{STA}}(t, i^{\text{STA}})$ in (53) and using (54);
5. the induced obstacles $O_i(t)$ for each $k > i$ can be computed using (44).

Intruder Avoidance Algorithm (Online Replanning)

For each vehicle $i$ that performed avoidance,

1. compute $V_{\pi}^{\text{ST}}(t, i)$ using (55). The new $t_i^{\text{STA}}$ for $Q_i$ is given by $\arg \inf_{\hat{t}} V_{\pi}^{\text{ST}}(t, i) \cap L_i \neq \emptyset$;
2. given $t_i^{\text{STA}}, \tilde{x}_i^{\pi}$, vehicle dynamics (1), target set $L_i$, and static obstacles $O_i^{\text{static}}, i = 1, \ldots, N$, use any of the three STP methods discussed in Section V for re-planning.

the STP vehicles. Thus, $L$ really corresponds to the time an intruder starts affecting a STP vehicle.

Remark 5: Note that even though we have presented the analysis for one intruder, the proposed method can handle multiple intruders as long as only one intruder is present at any given time.

We conclude this section with the STP algorithm that takes into account an intruder that may appear for a duration of $t_{\text{LAT}}$ (see Algorithm 5).

B. Numerical Simulations

To illustrate that STP is robust with respect to disturbances as well as a single intruder present for a duration of $t_{\text{LAT}}$, we use a five-vehicle example, in which one of the five vehicles is an intruder. We assume that each vehicle has the dynamics given in (41). For this example, we chose the parameters $\eta = 0.1, \bar{\eta} = 1$, and $\bar{\omega} = 1$, and disturbance bounds $d_e = 0.1$ and $d_\omega = 0.2$, which correspond to a 10% uncertainty in the dynamics.

The vehicles’ initial states, scheduled times of arrival, and target sets are the same as those described in Section V-B, except that in this example, we have increased the target radius to $r = 0.15$. For illustration purposes, we have chosen to use the robust trajectory tracking method described in Section V-A3 for the base obstacles’ computation, and hence each vehicle tracks a nominal trajectory.

Fig. 12 shows the simulation at $t = \bar{t} = -2.39$, which corresponds to the time at which the intruder “disappears” from the domain. This time is chosen to maximally highlight the impact of the intruder. Here, the intruder is shown in black, and the STP vehicles are shown in the other different colors.

By the time $t = -2.39$, vehicle $Q_2$ (red) and vehicle $Q_3$ (green) have been avoiding the intruder for some time. This is evident from the amount of deviation between the actual positions of vehicles $Q_2$ and $Q_3$ (denoted by *) and their nominal positions (denoted by o) specified by the nominal trajectories they originally planned to track; these vehicles have abandoned nominal trajectory tracking in order to ensure safety with respect to the intruder during the time interval marked by the vertical black dotted lines. Before the intruder’s presence, both vehicles track their nominal trajectories closely; however, both vehicles later deviate significantly from their nominal trajectories in order to avoid the intruder. After the intruder is gone, both vehicles replan their trajectories and arrive at their targets at a later time.

The STP vehicles are rather far apart because a large margin is needed to ensure that separation even when multiple vehicles need to avoid the intruder. In this example in particular, the lowest priority vehicle $Q_4$ needed to depart very early compared to $Q_2$ and $Q_3$ so that if an intruder were to arrive, $Q_4$ does not impede the ability of the other vehicles to perform avoidance. The early departure of $Q_4$ can be inferred from the fact that at $t = -2.39$, it is already nearly at its target.

For the same reason, the highest priority vehicle $Q_1$ has not departed from its initial state yet, and thus is not shown at $t = -2.39$. $Q_2$ and $Q_3$ needed to depart very early compared with $Q_1$ to ensure sufficient margin for avoidance maneuvers.

Fig. 13 shows the nominal (black) and actual trajectories (red and green, respectively) of vehicles $Q_2$ (top subplot) and $Q_3$ (bottom subplot). Specifically, the x and y positions
over time are shown, and the black dotted vertical lines indicate the time interval in which the intruder is present. From Fig. 13, one can clearly see that before the intruder was present, both vehicles are able to track their nominal trajectories closely. When the intruder appears, the vehicles deviate from their nominal trajectories significantly. After the intruder disappears, both vehicles replan new trajectories, and at a later time, the resulting actual trajectories eventually arrive at the same location as the nominal trajectories.

In general, when all vehicles have different dynamics, the total computation time scales quadratically with the number of vehicles, because each vehicle must consider the BRs (47) from the induced obstacle of higher priority vehicles, as discussed in Section VI-A1; the induced obstacles involve the FRS (42), which depends on the dynamics of the higher priority vehicles. In our simulation, we assumed that all vehicles have the same dynamics, and thus, the BRs (47) are equivalent up to a coordinate transformation, resulting in a computation time that scales linearly with the number of vehicles. For the simulations shown, computations were done on a desktop computer with a Core i7 5820K processor and two GeForce GTX Titan X graphics processing units. The average computation time per vehicle is approximately 5 s using CUDA and GPU parallelization.

VII. CONCLUSION AND FUTURE WORK

Guaranteed-safe multivehicle trajectory planning is a challenging problem, and previous analyses often either require strong assumptions on the motion of the vehicles or result in a large degree of conservatism. Differential game techniques, such as the HJ reachability, are ideally suited for guaranteeing goal satisfaction and safety under disturbances, but become intractable for even a small number of vehicles.

Our robust STP method assigns a strict priority ordering to vehicles to offer a tractable and practical approach to the multivehicle trajectory planning problem. Under the proposed method, a portion of “space–time” is reserved for vehicles in the airspace in descending priority order to allow for dense vehicle configurations. Unlike previous priority-based methods, our approach accounts for disturbances and an adversarial intruder. STP reduces the scaling of HJ reachability’s computational complexity from exponential to linear with respect to the number of vehicles, while maintaining hard guarantees on goal satisfaction and safety under disturbances.

In the presence of a single intruder vehicle, STP still guarantees goal satisfaction and safety with a quadratically scaling computational complexity.

In the future, we plan to investigate the ways of guaranteeing a maximum number of vehicles that need to replan, to combine reachability analysis with other trajectory and path planning methods to improve computation speed, and to better understand the scenarios, under which the STP scheme is the most useful by running large-scale simulations.

REFERENCES


Mo Chen received the B.A.Sc. degree in engineering physics from The University of British Columbia, Vancouver, BC, Canada, in 2011, and the Ph.D. degree from the Electrical Engineering and Computer Sciences Department, University of California at Berkeley, Berkeley, CA, USA, in 2017, under the supervision of Dr. C. Tomlin. He is currently a Postdoctoral Researcher with the Aeronautics and Astronautics Department, Stanford University, Stanford, CA, USA, under the supervision of Dr. M. Pavone. He will be joining Simon Fraser University, Burnaby, BC, Canada, as an Assistant Professor with the School of Computing Science in 2018. His current research interests include multiagent systems, safety-critical systems, and control theory.

Dr. Chen received the 2017 Eli Jury Award and the 2016 Demetri Angelakos Memorial Achievement Award.

Claire J. Tomlin (F’10) was an Assistant, an Associate, and a Full Professor in aeronautics and astronautics with Stanford University, Stanford, CA, USA, from 1998 to 2007. In 2005, she joined the University of California at Berkeley, Berkeley, CA, USA, where she is currently the Charles A. Desoer Professor of Engineering in electrical engineering and computer sciences. Her current research interests include control theory and hybrid systems, with applications to air traffic management, unmanned aerial vehicle systems, energy, robotics, and systems biology.

Dr. Tomlin is a MacArthur Foundation Fellow (2006). In 2010, she held the Tage Erlander Professorship of the Swedish Research Council at KTH Royal Institute of Technology, Stockholm, Sweden.