A Path Defense Approach to The Multiplayer Reach-Avoid Game

Mo Chen, Zhengyuan Zhou, and Claire J. Tomlin

***Abstract*—**

I. Introduction

Differential games have been studied extensively, and are powerful theoretical tools in robotics, aircraft control, security, and other domains [1], [2], [3]. The multiplayer reach-avoid game (to be defined precisely in Section II) is a differential game between two adversarial teams of cooperating players, where one team attempts to reach a certain target quickly while the other team aims to delay, or if possible, prevent the opposing team from reaching the target. One example of a reach-avoid game is the popular game Capture-the-Flag (CTF) [4], [5]. In robotics and automation, CTF has been explored most notably in the Cornell Roboflag competition, where two opposing teams of mobile robots are directed by human players to play the game [6]. A number of results related to motion planning and human-robot interactions have been reported from the competition [7], [8], [9], [10].

A multiplayer reach-avoid game is a complex game due to both the conflicting goals of the two teams and the cooperation among the players within each team, rendering the optimal solution for each team nontrivial to obtain and visualize. Previous work [4], [5] has shown that even in a 1 vs. 1 scenario, human agents are sometimes unable to find the optimal way to play, losing in situations in which an optimal winning strategy exists. For general multiplayer reach-avoid games, optimal solutions are extremely difficult to compute due to the intrinsic high dimensionality of the joint state space. Multiplayer differential games have been previously addressed using various techniques. In [7], where a team of defenders assumes that the attackers move towards their target in straight lines, a mixed-integer linear programming approach was used. In [11], optimal defender strategies are determined using a linear program, with the assumption that the attackers use a linear feedback control law. In complex pursuit-evasion games where players may change roles over time, nonlinear model-predictive control [12] and approximate dynamic programming [13] approaches have been investigated. In both cases, opponent strategies are estimated based on explicit prediction models.

The above-mentioned techniques tend to work well in the situations in which accurate models of the opponent team can be obtained. While those techniques have proven to be effective in their corresponding scenarios, they cannot be easily adapted to solve a general multiplayer reach-avoid game when no prior information on each side is known. The ideal framework to use for such a general multiplayer reach-avoid game is the classical Hamilton-Jacobi-Isaacs (HJI) approach [14], in which an HJI partial differential equation (PDE) is solved to obtain optimal strategies for both teams. In this case, if both teams play optimally, the result of the game is completely determined by the joint initial condition of the players. In addition, modern numerical tools [15], [16], [17] have been developed to carry out the computations and leverage the power of the HJI framework when the dimensionality of the problem is low. These numerical tools have been employed to successfully solve a variety of differential games, path planning problems, and optimal control problems, including aircraft collision avoidance [15], automated in-flight refueling [18], and two-player reach-avoid games [5]. These tools offer tremendous flexibility in terms of the player dynamics and terrain, and do not explicitly assume any specific control strategy or prediction models for the players. A notable success [15] has been witnessed in applying such tools to solve low dimensionality problems.

Despite the power that the HJI framework and the numerical tools have to offer, solving a general multiplayer reach-avoid game is computationally intractable due to the curse of dimensionality: the joint state space of two N-player teams in a two-dimensional domain is $4N$-dimensional. Numerically, when the state space is discretized, the number of nodes scales exponentially with the number of dimensions. Therefore, computing optimal solutions for a general multiplayer reach-avoid game in the HJI framework is out of reach. As a result, the inherent trade-off between optimality of the solutions and computational complexity must be considered and made. Our solution in this paper is no exception to this rule.

In [19], we considered each defender-attacker pair and compute the optimal solutions for both players using the HJI framework. We then invoke the graph-theoretical maximum matching algorithm [20], [21] to determine optimal pairings. This procedure approximates the HJI solution to the multiplayer game by combining the solutions from the N^2 two-player games between each attacker-defender pair. This procedure reduces the computation complexity from solving

This work has been supported in part by NSF under CPS:ActionWebs (CNS-931843), by ONR under the HUNT (N0014-08-0696) and SMARTS (N00014-09-1-1051) MURIs and by grant N00014-12-1-0609, by AFOSR under the CHASE MURI (FA9550-10-1-0567).

M. Chen, and C. J. Tomlin are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA {mochen72,tomlin}@eecs.berkeley.edu

Z. Zhou is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA zyzhou@stanford.edu

We thank Haomiao Huang for sharing MatLab code for 4D HJ reachability calculations.

a $4N$-dimensional HJI PDE to solving N^2 4-dimensional HJI PDEs. This is a substantial complexity reduction; however, without solving each 4-dimensional HJI PDE is still very time- and memory-intensive. This is the motivation of this paper.

Our contributions can mainly be stated as follows. First, it provides a computationally efficient approximation to the solution proposed in [5] and [19] on the two-player reach-avoid game. Having computed this approximiation, we the same extension of the two-player game to the multiplayer game is easily implemented at almost no additional computational cost compared to the two-player games. Hence, our approach provides an appealing solution, especially when the number of players becomes large. In addition, some cooperation is taken into account by the maximum matching process. Second, we quantify the degree of conservatism in our solution using the 4-dimensional HJI calculations in [19] as a bench mark.

II. Problem Formulation

We consider a multiplayer reach-avoid game between a team of N attackers, $\{P_{A_i}\}_{i=1}^N = \{P_{A_1}, P_{A_2}, \dots, P_{A_N}\}$ and a team of N defenders, $\{P_{D_i}\}_{i=1}^N = \{P_{D_1}, \dots, P_{D_N}\}$. Each player is confined in a bounded, open domain $\Omega \subset \mathbb{R}^2$, which can be partitioned as follows: $\Omega = \Omega_{free} \cup \Omega_{obs}$. Ω_{free} is a compact set representing the free space in which the players can move, while $\Omega_{obs} = \Omega \setminus \Omega_{free}$ represents the obstacles that obstruct movement in the domain. Let $x_{A_i}, x_{D_j} \in \mathbb{R}^2$ denote the states of the players P_{A_i} and P_{D_j} respectively. Initial conditions of the players are denoted by $x^0_{A_i}, x^0_{D_i} \in \Omega_{free}, i = 1, 2, \dots, N$. We assume that the dynamics of the players are defined by the following decoupled system for $t \geq 0$:

$$\begin{aligned} \dot{x}_{A_i}(t) &= v_{A_i} a_i(t), \qquad & x_{A_i}(0) &= x^0_{A_i}, \\ \dot{x}_{D_i}(t) &= v_{D_i} d_i(t), \qquad & x_{D_i}(0) &= x^0_{D_i}, \\ & i = 1, 2, \dots, N \end{aligned} \tag{1}$$

where $a_i(\cdot), d_i(\cdot)$ represent the control functions of P_{A_i} and $P_{D_i}, i = 1, 2, \dots, N$ respectively. v_{A_i}, v_{D_i} represent the maximum speeds of the i^{th} attacker and i^{th} defender, respectively; each player can have a different maximum speed. We assume that every defender is no slower than all attackers: $\forall i, v_{D_i} \geq v_{A_j} \forall j$. We further assume that the control functions $a_i(\cdot), d_i(\cdot)$ are drawn from the set $\Sigma = \{\sigma \colon [0, \infty) \to \overline{B}_n \mid \sigma \text{ is measurable}\}$, where $\overline{B}_n$ denotes the closed unit ball in $\mathbb{R}^2$.

As a clarification on the notation and terminology, the control functions (with a dot notation, e.g. $a_i(\cdot), d_i(\cdot), u(\cdot)$ etc.) which are the entire control trajectories, are distinguished from the control inputs (such as $a_i, a_i(t), d_i, d_i(t)$ etc.) which are the instantaneous control inputs. Furthermore, given $x^0_{A_i} \in \Omega_{free}$, we define the admissible control function set for P_{A_i} to be the the set of all control functions such that $x_{A_i}(t) \in \Omega_{free}, \forall i, \forall t \geq 0$. The admissible control function set for defenders $P_{D_i}, i = 1, 2, \dots, N$ is defined similarly, given that $x^0_{D_i} \in \Omega_{free}$. The joint state of all the players is denoted by $\mathbf{x} = (x_{A_1}, \dots x_{A_N}, x_{D_1}, \dots, x_{D_N})$. The joint initial condition is denoted by $\mathbf{x^0} = (x^0_{A_1}, \dots, x^0_{A_N}, x^0_{D_1}, \dots, x^0_{D_N})$.

In this reach-avoid game, the attacking team aims to reach the target $\mathcal{T} \subset \Omega_{free}$, a compact subset of the domain, without getting captured by the defenders. The capture conditions are formally described by the capture sets $\mathcal{C}_{ij} \subset \Omega^{2N}$ for the pairs of the players $(P_{A_i}, P_{D_j}), i, j = 1, \dots, N$. In general, $\mathcal{C}_{ij}$ can be an arbitrary compact subset of Ω^{2N}, which represents the set of the joint player states $\mathbf{x}$ at which P_{A_i} is captured by P_{D_j}. Hence, in the general case, the interpretation of capture is given by the set $\mathcal{C}_{ij}$, which in turn depends on the specific situation one wishes to model. In this paper, we define the capture sets to be $\mathcal{C}_{ij} = \{\mathbf{x} \in \Omega^{2N} \mid x_{A_i} = x_{D_j}\}$, the interpretation of which is that P_{A_i} is captured by P_{D_j} if P_{A_i}'s position coincides with P_{D_j}'s position. The lack of capture radius allows for clearer presentation of our analysis, easily extends to the case in which there is a positive capture radius.

In our multiplayer reach-avoid game, the team of the attackers $\{P_{A_i}\}_{i=1}^N$ wins when at least m attackers reach the target $\mathcal{T}$ without being captured. The team of the defenders $\{P_{D_i}\}_{i=1}^N$ wins the game if they can delay at least $N-m+1$ attackers from reaching the target indefinitely. An illustration of the game setup is shown in Figure 1.

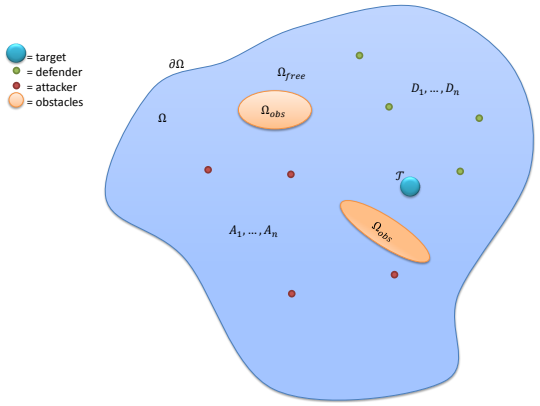


Fig. 1: An illustration of the components of a multiplayer reach-avoid game.

With the above definitions, we now state the main question that this paper answers. Given the minimum number m of the attackers that need to reach the target, the joint initial state $\mathbf{x^0}$ of all the players in domain Ω with obstacles Ω_{obs}, the target set $\mathcal{T}$, and the capture sets $\mathcal{C}_{ij}$, can the defenders be guaranteed to win?

As an example of the multiplayer reach-avoid game, consider the special case in which each team only has one player. The solution to this special case will be a building block for the the solution to the full multiplayer game. In the two-player game, we denote the attacker as P_A, the defender as P_D, their states as x_A, x_D, and their initial conditions as

x_A^0, x_D^0. Their dynamics are

$$\begin{aligned} \dot{x}_A(t) &= v_A a(t), \quad x_A(0) = x_A^0, \\ \dot{x}_D(t) &= v_D d(t), \quad x_D(0) = x_D^0. \end{aligned} \tag{2}$$

We assume $v_D \geq v_A$. The players' joint state and joint initial condition become $\mathbf{x} = (x_A, x_D), \mathbf{x^0} = (x_A^0, x_D^0)$ respectively. The capture set for P_A is then simply

$$\mathcal{C} = \left\{(x_A, x_D) \in \Omega^2 \mid x_A = x_D\right\}. \tag{3}$$

The attacker P_A wins when she reaches the target $\mathcal{T}$ without being captured by the defender P_D. If the defender P_D can delay P_A from reaching $\mathcal{T}$ indefinitely, the defender wins.

This two-player reach-avoid game and its variant were first studied in [5] and [22]. In this paper, we provide a computationally efficient approximation to the HJI framework, and utilize the results in [19] to deal with the multiplayer reach-avoid games.

III. Hamilton-Jacobi Reachability

Taken from ACC 2014 paper, need to shorten The multiplayer reach-avoid game is a differential game in which two teams have competing objectives [23]. By specifying the winning conditions, we can determine the winning regions of the attacker and the defender by numerically computing the HJ reachable sets. The numerical HJ computations allow us to consider games with arbitrary terrain, domain, obstacles, target set, and player velocities. Furthermore, the results of HJ computation assume a closed-loop strategy for both players given previous information of the other players.

The setup for using HJ reachability to solve differential games can be found in [15], [24], [5]. In summary, we are given the continuous dynamics of the system state:

$$\dot{\mathbf{x}} = f(\mathbf{x}, u, d), \mathbf{x}(0) = \mathbf{x^0}, \tag{4}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the system state, $u \in \mathbb{U}$ is the joint control input of the attacking team, and $d \in \mathbb{D}$ is the joint control input of the defending team. The sets $\mathbb{U}$ and $\mathbb{D}$ represent the sets of the joint admissible control inputs of the attacking team and the defending team, respectively. The attacking team selects a control input based on the past and the current joint states of all the players. The defending team then selects a control input based on the past and the present control inputs of the attacking team, in addition to the past and the current joint states. *A priori*, this information pattern is conservative towards the attackers, as defenders have more information available. However, in the case that the system (described by the function f) is decoupled, which is true in our reach-avoid game defined in Equation (1), the Isaacs condition [14] holds and the two information patterns yield the same optimal solutions for both the attackers and the defenders.

To use the HJ reachability framework, we specify the terminal set R (described in detail in the next subsection) as the attackers' winning condition, and propagate backwards this set subject to the constraint imposing that the attackers be outside the capture regions and the obstacles. This constraint is described by the avoid set A in more detail in the next subsection. The result is a reachable set that partitions the state space into two regions. All points inside the reachable set represent the joint initial conditions from which the attacking team is guaranteed to win, and all points outside represent the joint initial conditions from which the defending team is guaranteed to win.

More precisely, the HJ reachability calculation is as follows. First, given a set G, the level set representation of G is a function $\phi_G : \mathbb{R}^n \to \mathbb{R}$ such that $G = \{\mathbf{x} \in \mathbb{R}^n \mid \phi_G \leq 0\}$. In particular, the terminal set R and the avoid set A will be represented by the functions ϕ_R and ϕ_A respectively.

Let $\Phi : \mathbb{R}^n \times [-T, 0] \to \mathbb{R}$ be the viscosity solution [25] to the constrained terminal value HJI PDE:

$$\frac{\partial \Phi}{\partial t} + \min\left[0, H\left(\mathbf{x}, \frac{\partial \Phi}{\partial \mathbf{x}}\right)\right] = 0, \ \Phi(\mathbf{x}, 0) = \phi_R(\mathbf{x}) \tag{5}$$

subject to

$$\Phi(\mathbf{x}, t) \geq -\phi_A(\mathbf{x}),$$

where the optimal Hamiltonian is given by

$$H(\mathbf{x}, p) = \min_{u \in \mathbb{U}} \max_{d \in \mathbb{D}} p^T f(\mathbf{x}, u, d).$$

By the argument presented in [15] and [26], the set of initial conditions from which the attackers are guaranteed to win within time T is given by

$$\mathcal{RA}_T(R, A) := \{\mathbf{x} \in \mathbb{R}^n \mid \Phi(\mathbf{x}, -T) \leq 0\}. \tag{6}$$

Hence, $\Phi(\mathbf{x}, -T)$ is the level set representation of $\mathcal{RA}_T(R, A)$.

The optimal control input for the attacking team is given by [27], [28], [5]:

$$u^*(\mathbf{x}, t) = \arg\min_{u \in \mathbb{U}} \max_{d \in \mathbb{D}} p(\mathbf{x}, -t)^T f(\mathbf{x}, u, d), \, t \in [0, T] \tag{7}$$

where $p = \frac{\partial \Phi}{\partial \mathbf{x}}$.

Similarly, an initial player configuration outside $\mathcal{RA}_T(R, A)$ guarantees that the defenders will win by using the optimal control input

$$d^*(\mathbf{x}, t) = \arg\max_{d \in \mathbb{D}} p(\mathbf{x}, -t)^T f(\mathbf{x}, u^*, d), \ t \in [0, T]. \tag{8}$$

Taking $T \to \infty$, we obtain the set of initial conditions from which the attackers are guaranteed to win. We denote this set $\mathcal{RA}_\infty(R, A)$. The set of initial conditions from which the defenders are guaranteed to win is given by all points not in $\mathcal{RA}_\infty(R, A)$. Note that for an N vs. N game on a two-dimensional domain $\Omega \subset \mathbb{R}^2$, the reachable set $\mathcal{RA}_\infty(R, A)$ is $4N$-dimensional.

A highly accurate numerical solution to Equation (5) can be computed using the Level Set Toolbox for MATLAB [24].

In the next two subsections, we will describe the terminal set and the avoid set for the multiplayer and the two player reach-avoid games.

A. Hamilton-Jacobi Reachability for the Multiplayer Game

In the multiplayer reach-avoid game, the goal of the attacking team is to send at least m attackers to the target set $\mathcal{T}$. For a particular set of m attackers, the goal set is given by each of the attackers being inside the target but outside of the capture radius of all the defenders. To obtain the terminal set R for the game (i.e. at least m attackers reaching the target), we take the union of all $\binom{N}{m}$ goal sets.

The avoid set is defined by the losing condition of the attackers: the attackers lose the game when at least $N - m + 1$ of them have been captured by the defender. For a particular attacker P_{A_i}, the individual keep-out set is given by $\bigcup_{j=1}^{N} \mathcal{C}_{ij}$, corresponding to P_{A_i} being captured by any of the defenders $P_{D_j}, j = 1, 2, \ldots, N$. For a particular set of $N - m + 1$ attackers, the joint keep-out set is characterized by each of these players being within the capture radius of some defender. The avoid set G_A for at least $N - m + 1$ attackers being captured is the union of all $\binom{N}{N-m+1}$ such joint keep-out sets.

The HJ reachability calculation for the multiplayer reach-avoid game is not only cumbersome to set up, but also intractable computationally due to the HJI PDE being solved on a discrete grid, which makes the computation complexity scale exponentially with the number of dimensions. In general, an HJI PDE of dimensions higher than five cannot be solved practically. Thus, we are limited to only being able to solve the HJI PDE corresponding to a two player game in which each player's state space is two-dimensional. Instead of directly solving the HJI PDE corresponding to the $2N$-player game, which is a $4N$-dimensional problem, we will solve the multiplayer game by combining the solution to the two player game and maximum matching from graph theory. This approach is outlined below.

B. Hamilton-Jacobi Reachability for the Two Player Game

In the two-player reach-avoid game, the goal of the attacker is to reach the target set $\mathcal{T}$ while avoiding capture by the defender. This goal is represented by the attacker being inside $\mathcal{T}$ but outside of the defender's capture radius.. En route to $\mathcal{T}$, the attacker must avoid capture by the defender. This is represented by the set $\mathcal{C}$.

In addition, both players need to avoid the obstacles Ω_{obs}, which can be considered as the locations in Ω where the players have zero velocity. In particular, the defender wins if the attacker is in Ω_{obs}, and vice versa. Therefore, we define the terminal set as

$$\begin{aligned} R = &\left\{\mathbf{x} \in \Omega^2 \mid x_A \in \mathcal{T} \wedge \|x_A - x_D\|_2 > R_C\right\} \\ &\cup \left\{\mathbf{x} \in \Omega^2 \mid x_D \in \Omega_{obs}\right\} \end{aligned} \tag{9}$$

Similarly, we define the avoid set as

$$\begin{aligned} A &= \left\{\mathbf{x} \in \Omega^2 \mid \|x_A - x_D\|_2 \leq R_C\right\} \cup \left\{\mathbf{x} \in \Omega^2 \mid x_A \in \Omega_{obs}\right\} \\ &= \mathcal{C} \cup \left\{\mathbf{x} \in \Omega^2 \mid x_A \in \Omega_{obs}\right\} \end{aligned} \tag{10}$$

Given these sets, we can define the corresponding level set representations ϕ_R, ϕ_A, and solve (5). Assuming $\Omega \subset \mathbb{R}^2$, the result is $\mathcal{RA}_\infty(R, A) \in \mathbb{R}^4$, a four dimensional reachable set with the level set representation $\Phi(\mathbf{x}, -\infty)$. The attacker wins if and only if the joint initial condition is such that $(x_A^0, x_D^0) = \mathbf{x^0} \in \mathcal{RA}_\infty(R, A)$.

If $\mathbf{x^0} \in \mathcal{RA}_\infty(R, A)$, then the attacker is guaranteed to win the game by using the optimal control input given in (7). Applying Equation (7) to the two player game, we have that the attacker winning strategy satisfies

$$a^*(x_A, x_D, t) = \arg\min_{a \in \mathbb{U}} \max_{d \in \mathbb{D}} p(x_A, x_D, -t)^T f(x_A, x_D, a, d) \tag{11}$$

for $t \in [0, T]$. The explicit winning strategy satisfying (11) is given in [5] as

$$a^*(x_A, x_D, t) = -v_A \frac{p_u(x_A, x_D, -t)}{\|p_u(x_A, x_D, -t)\|_2}. \tag{12}$$

where $p = (p_u, p_d) = \frac{\partial \Phi}{\partial (x_A, x_D)}$.

Similarly, if $\mathbf{x^0} \notin \mathcal{RA}_\infty(R, A)$, then the defender is guaranteed to win the game by using the optimal control input given in (8). Applying Equation (8) to the two player game, we have that the defender winning strategy satisfies

$$d^*(x_A, x_D, t) = \arg\max_{d \in \mathbb{D}} p(x_A, x_D, -t)^T f(x_A, x_D, a^*, d) \tag{13}$$

for $t \in [0, T]$. The explicit winning strategy satisfying (13) is given in [5] as

$$d^*(x_A, x_D, t) = v_D \frac{p_d(x_A, x_D, -t)}{\|p_d(x_A, x_D, -t)\|_2}. \tag{14}$$

IV. The Path Defense Solution to the Two Player Game

The HJI solution to the two-player reach-avoid game involves gridding up a 4-dimensional space, and is thus time- and memory-intensive to compute. Instead of computing the entire 4-dimensional reach-avoid set given by the HJI solution, we will approximate 2-dimensional slices of the reach-avoid set.

In this section, we first introduce the path defense game, a specific type of reach-avoid games, describe an efficient way to approximately solve the path defense game, and quantify the conservatism of the approximation. The solution to the path defense game will be a building block for approximating 2-dimensional reach-avoid slices of the general two player reach-avoid game, which will be presented in the next section.

A. The Path Defense Game

The Path Defense Game is a two player reach-avoid game in which the boundary of the target set is the shortest path between two particular points on $\partial\Omega$, and the target set is on one side of the shortest path. We denote the target set as $\mathcal{T} = \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ for some given points on the boundary e_a, e_b. $\mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right), \bar{\mathcal{P}}(e_a, e_b), e_a, e_b$ are defined below.

Definition 1: **Path of defense**. Denote the shortest path between two points x, y to be $\mathcal{P}(x, y)$. For convenience, the

length of $\mathcal{P}(x, y)$ will be denoted $\text{dist}(x, y)$, and the time it takes for the attacker and defender to traverse $\mathcal{P}(x, y)$ will be denoted $t_A(x, y), t_D(x, y)$, respectively. Note the distinction between $\text{dist}(\cdot, \cdot)$, which denotes distance, and $d(\cdot)$, which denotes control function of the defender.

A path of defense, $\bar{\mathcal{P}}(e_a, e_b)$, is the shortest path between points e_a and e_b located on $\partial\Omega$, the boundary of the domain. e_a and e_b are referred to as the anchor points of path $\bar{\mathcal{P}}(e_a, e_b)$.

Definition 2: **Attacker's side of the path**. A path of defense $\bar{\mathcal{P}}(e_a, e_b)$ partitions the domain Ω into two regions, one of which contains the attacker. Define $\mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ to be the region that contains the attacker, not including points on the path $\bar{\mathcal{P}}(e_a, e_b)$. The attacker seeks to reach the target set $\mathcal{T} = \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$.

The basic setup of the path defense game is illustrated in figure 2 below.

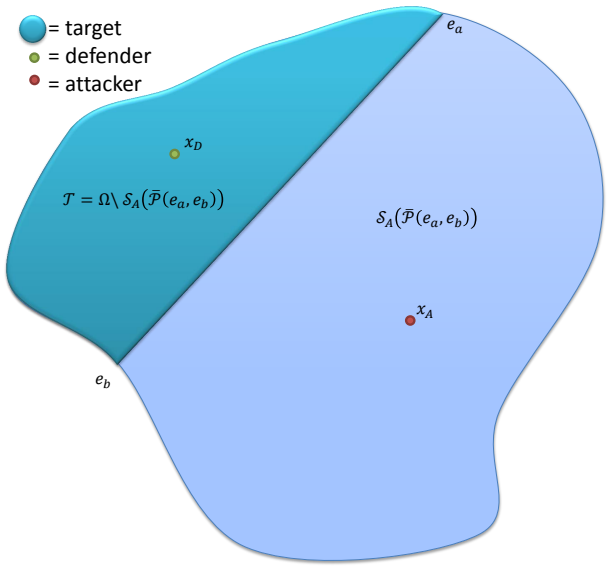


Fig. 2: An illustration of the components of a path defense game between two players.

B. Solving the path defense game

A path defense game can be solved by solving a 4-dimensional HJI corresponding to the problem as described in section III-A with $\mathcal{T} = \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$. However, the following definitions will allow us to solve the game conservatively for the defender without solving an HJI.

Definition 3: **Defendable path**. Given initial conditions $\mathbf{x^0} = (x_A^0, x_D^0)$, a path $\bar{\mathcal{P}}(e_a, e_b)$ is defendable if regardless of the attacker's actions, the defender has a control function $d(\cdot)$ to prevent the attacker from reaching $\bar{\mathcal{P}}(e_a, e_b)$ without being captured.

Definition 4: **Strongly defendable path**. Given initial conditions $\mathbf{x^0} = (x_A^0, x_D^0)$, a path $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable if regardless of the attacker's actions, the defender has a control function $d(\cdot)$ to reach $\bar{\mathcal{P}}(e_a, e_b)$ after finite time and prevent the attacker from reaching $\bar{\mathcal{P}}(e_a, e_b)$ without being captured. Note that the defender is not explicitly required to stay on $\bar{\mathcal{P}}(e_a, e_b)$ after reaching it.

Remark 1: A path $\bar{\mathcal{P}}(e_a, e_b)$ is defendable if it is strongly defendable.

Observe that checking whether a path $\bar{\mathcal{P}}(e_a, e_b)$ is defendable is exactly the path defense problem. Since solving the path defense problem involves a 4-dimensional HJI calculation, we instead consider the problem of checking whether a path $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable. The following definitions lead to our first Lemma which describes how to determine strong defendability; the definitions and Lemma are illustrated in Figure 3.

Definition 5: **Level set image of attacker**. Given attacker position $x_A(t)$, define the level set image of the attacker with respect to anchor point e_a to be $x_A\prime(t; e_a) = \{x \in \bar{\mathcal{P}}(e_a, e_b) : t_A(x, e_a) = t_A(x_A(t), e_a)\}$. $x_A\prime$ is unique. Define $x_A\prime(t; e_b)$ similarly by replacing e_a with e_b. For convenience, we will sometimes omit the time argument and write $x_A\prime(e_a)$.

Remark 2: $x_A\prime(e_a)$ is closer to e_b than $x_A\prime(e_b)$.

Remark 3: Given some path of defense $\bar{\mathcal{P}}(e_a, e_b)$, if the defender's position coincides with the level set image of the attacker, i.e. $x_D(s) = x_A\prime(s; e_a)$ (or $x_A\prime(s; e_b)$) at some time s, then there exists a control for the defender to always remain on the level set image of the attacker thereafter, i.e. $x_D(t) = x_A\prime(t; e_a)$ (or $x_A\prime(t; e_b)$) $\forall t \geq s$.

Definition 6: **Regions induced by point p on path**. Given a point p on a path of defense $\bar{\mathcal{P}}(e_a, e_b)$, define a region $\mathcal{R}_a(p)$ associated the point p and anchor point e_a as follows:

$$\mathcal{R}_a(p) = \{x : \text{dist}(x, e_a) \leq \text{dist}(p, e_a)\} \tag{15}$$

Define $\mathcal{R}_b(p)$ similarly by replacing e_a with e_b.

Lemma 1: Suppose that the defender is on some point p on the path $\bar{\mathcal{P}}(e_a, e_b)$, i.e. $x_D^0 = p$. Furthermore, assume that $v_A = v_D$. Then, $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable if and only if the attacker's initial position x_A^0 is outside the region induced by p: $x_A^0 \in \Omega \backslash (\mathcal{R}_a \cup \mathcal{R}_b)$.

proof 1: We assume $x_A^0 \notin \mathcal{T} = \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$, otherwise the attacker would start inside the target set.

First, we show that if $x_A^0 \in \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right) \cap (\mathcal{R}_a \cup \mathcal{R}_b)$, then the attacker can reach e_a or e_b and hence $\Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ without being captured.

Without loss of generality, suppose $x_A^0 \in \mathcal{R}_a$. To capture the attacker, the defender must necessarily be on $x_A\prime(e_a)$ or $x_A\prime(e_b)$ at some time t. By definition 6, we have $\text{dist}(x_A^0, e_a) < \text{dist}(p, e_a)$, so $t_A(x_A\prime(e_a), e_a) < t_D(p, e_a)$. By remark 2, we also have $t_D(p, x_A\prime(e_a)) \leq t_D(p, x_A\prime(e_b))$, so it suffices to show that the defender never reaches $x_A\prime(e_a)$ before the attacker reaches e_a.

If the attacker moves towards e_a along $\mathcal{P}(x_A^0, e_a)$ with maximum speed, then $x_A\prime(e_a)$ will move towards e_a along $\mathcal{P}(x_A\prime(e_a), e_a)$ at the same speed. Since $t_A(x_A, e_a) = t_A(x_A\prime(e_a), e_a) < t_D(p, e_a)$, $x_A\prime(e_a)$ will reach e_a before x_D does. When $x_A\prime(e_a) = e_a$, we also have $x_A = e_a \in \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$. Thus, the defender never captures the attacker's level set image. Therefore, no matter what the

defender does, the attacker can reach $\Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ by moving towards e_a at maximum speed along $\mathcal{P}(x_A, e_a)$.

Next, we show, by contradiction, that if $x_A \in \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right) \setminus (\mathcal{R}_a \cup \mathcal{R}_b)$, then the attacker cannot reach $\Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ without being captured.

Suppose x_A will reach some point $p\prime$ before x_D does. Without loss of generality, assume $p\prime \in \mathcal{P}(p, e_a)$, and note that $\text{dist}(p, e_a) < \text{dist}(x_A, e_a)$ since the attacker is not in $\mathcal{R}_a$. Since $t_A(x_A, e_a)$ is the minimum time for the attacker to reach e_a, we have

$$\begin{aligned} \text{dist}(x_A, e_a) &\leq \text{dist}(x_A, p\prime) + \text{dist}(p\prime, e_a) \\ &\quad \text{(Definition of shortest path)} \\ &< \text{dist}(p, p\prime) + \text{dist}(p\prime, e_a) \\ &\quad \text{(by assumption, } \text{dist}(x_A, p\prime) < \text{dist}(p, p\prime)) \\ &= \text{dist}(p, e_a) \\ &< \text{dist}(x_A, e_a) \\ &\quad \text{since } x_A \notin \mathcal{R}_a \end{aligned} \tag{16}$$

This is a contradiction. Therefore, the attacker cannot cross any point $p\prime$ on $\bar{\mathcal{P}}(e_a, e_b)$ without being captured. This proves lemma 1. The proof is illustrated in figure 3

Given that the defender starts at p on $\bar{\mathcal{P}}(e_a, e_b)$, lemma 1 partitions Ω into two regions, assuming $v_A = v_D$: if the attacker is initially in $\mathcal{R}_a \cup \mathcal{R}_b$, then $\bar{\mathcal{P}}(e_a, e_b)$ is not strongly defendable; otherwise, the path is strongly defendable. In the case that $v_A < v_D$, the attacker being in $\mathcal{R}_a \cup \mathcal{R}_b$ becomes a sufficient condition (not necessary) for the strong defendability of p.

In general, the initial position of the defender, x_D^0, may not be on $\bar{\mathcal{P}}(e_a, e_b)$. In this case, to strongly defend $\bar{\mathcal{P}}(e_a, e_b)$, the defender needs to first arrive at some point $p \in \bar{\mathcal{P}}(e_a, e_b)$. If the defender can arrive at p, before the attacker moves into the region $\mathcal{R}_a \cup \mathcal{R}_b$, then the path p is strongly defendable.

Thus, given initial conditions $\mathbf{x^0} = (x_A^0, x_D^0)$, we can check whether a path $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable by the following procedure:

For all points on the path $p \in \bar{\mathcal{P}}(e_a, e_b)$,

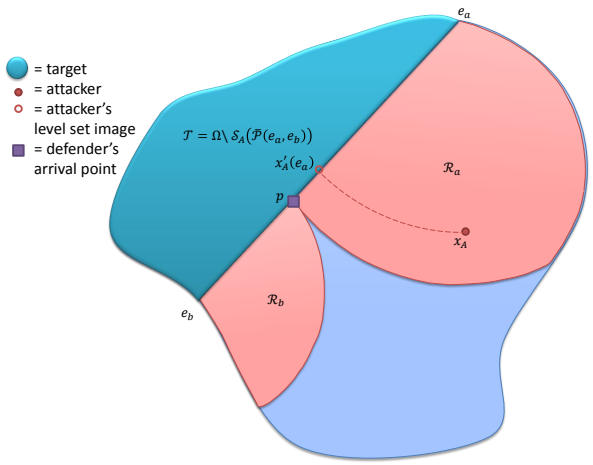
1) Compute the time it takes the defender to move from x_D^0 to p.
2) Compute the time it takes the attacker to move from x_A^0 to $\mathcal{R}_a \cup \mathcal{R}_b$.

If there exists a point p on $\bar{\mathcal{P}}(e_a, e_b)$ such that the defender can get to p before the attacker can get to the corresponding $\mathcal{R}_a \cup \mathcal{R}_b$, then the path is strongly defendable. Otherwise, the path is not strongly defendable.

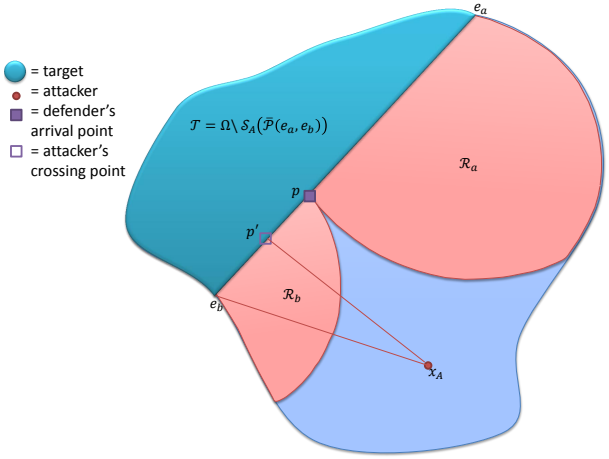
The above procedure requires two computations for every point on the path $\bar{\mathcal{P}}(e_a, e_b)$,. The following lemma shows that it is necessary and sufficient to perform the computations for only one point. This one special point will be denoted p^*.

Remark 4: It will be convenient to note that given $p \in \bar{\mathcal{P}}(e_a, e_b)$, the distance from the attacker to $\mathcal{R}_a(p)$ is $\text{dist}\left(x_A^0, \mathcal{R}_a(p, e_a)\right) = \text{dist}(x_A^0, e_a) - \text{dist}(p, e_a)$. Similarly, $\text{dist}\left(x_A^0, \mathcal{R}_b(p)\right) = \text{dist}(x_A^0, e_b) - \text{dist}(p, e_b)$

Lemma 2: Let a point p^* on the path $\bar{\mathcal{P}}(e_a, e_b)$ be such that the time it takes the attacker to get to $\mathcal{R}_a$ and the time



(a) Suppose the attacker is in $\mathcal{R}_a \cup \mathcal{R}_b$. If the attacker moves towards e_a, the defender cannot capture the attacker's level set image before the attacker reaches e_a.



(b) Suppose the attacker is not $\mathcal{R}_a \cup \mathcal{R}_b$, there is no point $p\prime$ on $\bar{\mathcal{P}}(e_a, e_b)$ that the attacker can reach without being captured.

Fig. 3: An illustration of the proof of lemma 1.

it takes the attacker to get to $\mathcal{R}_b$ are equal. Then, $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable if and only if the defender can defend $\bar{\mathcal{P}}(e_a, e_b)$ by first going to p^*.

proof 2: One direction is clear: if the defender can defend $\bar{\mathcal{P}}(e_a, e_b)$ by first going to p^*, then $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable by definition.

We will show the other direction by showing its contrapositive: if the defender cannot defend $\bar{\mathcal{P}}(e_a, e_b)$ choosing p^* as the first point of entry, then $\bar{\mathcal{P}}(e_a, e_b)$ is not strongly defendable. Equivalently, we will show that if choosing p^* as the first point of entry does not allow the defender to defend $\bar{\mathcal{P}}(e_a, e_b)$, then no other choice of p as the first point of entry does.

Suppose that the defender cannot defend $\bar{\mathcal{P}}(e_a, e_b)$ by choosing p^* as the first point of entry, but can defend $\bar{\mathcal{P}}(e_a, e_b)$ by choosing another point $p\prime$ as the first point of entry. Without loss of generality, assume $\text{dist}(p^*, e_a) - \text{dist}(p\prime, e_a) > 0$. This assumption moves $p\prime$ further away from e_a than p^*, causing $\mathcal{R}_a$ to move closer to x_A^0. Starting with

remark 4, we have

$$\begin{aligned} \text{dist}\left(x_A^0, \mathcal{R}_a(p^*)\right) &= \text{dist}(x_A^0, e_a) - \text{dist}(p^*, e_a) \\ t_A\left(x_A^0, \mathcal{R}_a(p^*)\right) &= t_A(x_A^0, e_a) - t_A(p^*, e_a) \end{aligned} \tag{17}$$

Similarly, for the point $p\prime$, we have

$$\begin{aligned} \text{dist}\left(x_A^0, \mathcal{R}_a(p\prime)\right) &= \text{dist}(x_A^0, e_a) - \text{dist}(p\prime, e_a) \\ t_A\left(x_A^0, \mathcal{R}_a(p\prime)\right) &= t_A(x_A^0, e_a) - t_A(p\prime, e_a) \end{aligned} \tag{18}$$

Then, subtracting the above two equations, we derive that the attacker will be able to get to $\mathcal{R}_a$ sooner by the following amount of time:

$$\begin{aligned} t_A\left(x_A^0, \mathcal{R}_a(p^*)\right) - t_A\left(x_A^0, \mathcal{R}_a(p\prime)\right) &= t_A(p\prime, e_a) - t_A(p^*, e_a) \\ &= t_A(p\prime, p^*) \\ &\geq t_D(p\prime, p^*) \end{aligned} \tag{19}$$

We now show that the defender can get to $p\prime$ sooner than to p^* by less than the amount $t_D(p\prime, p^*)$, and therefore the defender in effect "gains less time" than the attacker does by going to $p\prime$. We assume that $p\prime$ is closer to the defender than p^* is. Then, by the triangle inequality, we have

$$\begin{aligned} \text{dist}(x_D^0, p^*) &\leq \text{dist}(x_D^0, p\prime) + \text{dist}(p\prime, p^*) \\ \text{dist}(x_D^0, p^*) - \text{dist}(x_D^0, p\prime) &\leq \text{dist}(p\prime, p^*) \\ t_D(x_D^0, p^*) - t_D(x_D^0, p\prime) &\leq t_D(p\prime, p^*) \end{aligned} \tag{20}$$

This completes the proof.

Lemmas 1 and 2 give a simple algorithm to compute, given x_A^0, the region that the defender must be in for a path of defense $\bar{\mathcal{P}}(e_a, e_b)$ to be strongly defendable:

1) Given anchor points e_a, e_b and attacker position x_A^0, compute the point p^* and the induced regions $\mathcal{R}_a, \mathcal{R}_b$.
2) $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable if and only if the defender can get to p^* before the attacker gets to the induced regions $\mathcal{R}_a, \mathcal{R}_b$. Therefore, if the defender's initial position is in the region $t_D(x_D^0, p^*) \leq t_A(x_A^0, \mathcal{R}_a \cup \mathcal{R}_b)$, then $\bar{\mathcal{P}}(e_a, e_b)$ is strongly defendable.

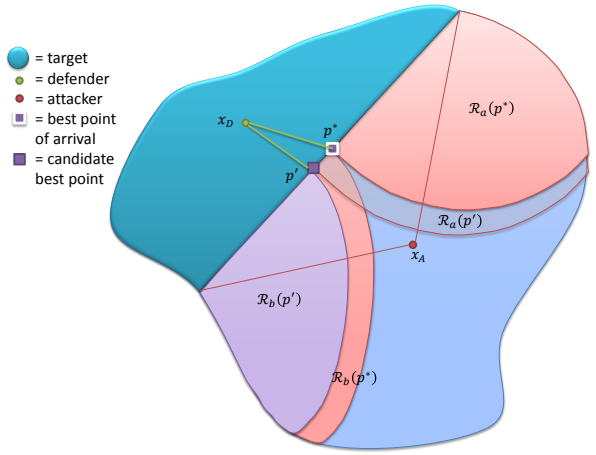
The computations in this algorithm can be efficiently computed by applying the fast marching method [] on a two-dimensional grid. Thus, we have conservatively solved the path defense problem, orginally a four-dimensional HJI problem, using a series of two-dimensional fast marching calculations.

Figure 4 illustrates the proof of lemma 2 and $\mathcal{G}(e_a, e_b)$.

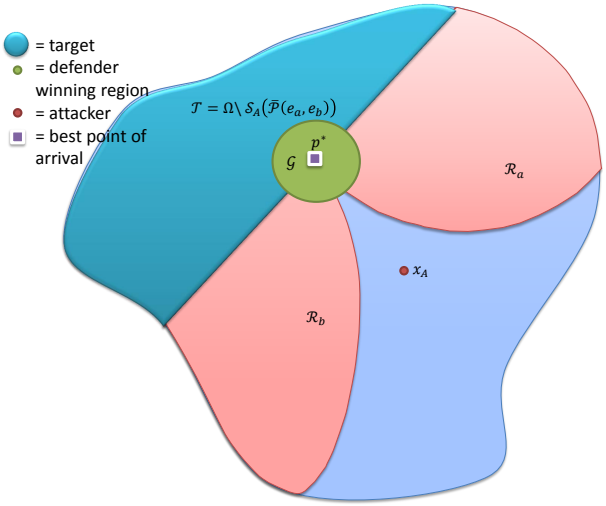
C. Conservatism of strong path defense

To quantify how conservative strong path defense is, we consider the following question: what paths of defense are defendable, but not strongly defendable? One example of such a path is the Voronoi line between an attacker and a defender with equal speeds in a convex domain. This is shown in figure 5.

To quantify the conservatism of strong path defense compared to path defense, we compute the defendable region and the strongly defendable region, defined below.



(a) If the defender cannot defend $\bar{\mathcal{P}}(e_a, e_b)$ by first going to p^*, then the defender cannot defend $\bar{\mathcal{P}}(e_a, e_b)$ by going to any other point p.

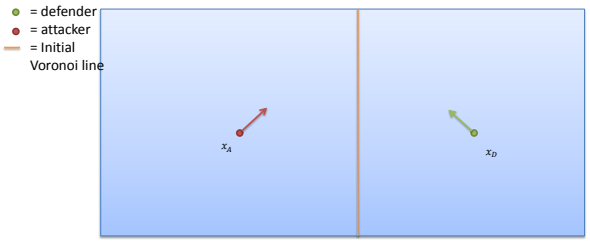


(b) Lemma 2 allows an easy computation of $\mathcal{G}(e_a, e_b)$.

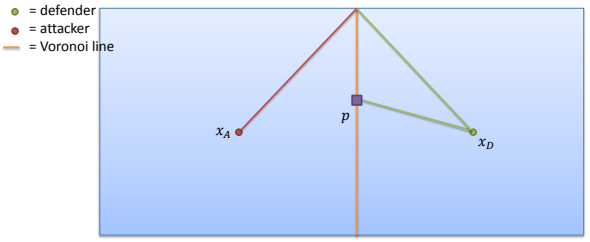
Fig. 4: An illustration of the proof of lemma 2 and $\mathcal{G}(e_a, e_b)$.

First, observe that if a path is strongly defendable, any path "behind" it on the defender's side is also strongly defendable. This is illustrated in figure 6. Suppose $\bar{\mathcal{P}}_i$ is strongly defendable, then it is defendable. This necessarily implies that $\bar{\mathcal{P}}_j$ is defendable, because if the defender defends $\bar{\mathcal{P}}_i$, then the attacker can never cross $\bar{\mathcal{P}}_i$ and thus never cross $\bar{\mathcal{P}}_j$. Since $\bar{\mathcal{P}}_i$ is strongly defendable, the defender can move to some point on $\bar{\mathcal{P}}_i$, and then defend $\bar{\mathcal{P}}_i$. However, on the way to $\bar{\mathcal{P}}_i$, the defender would have crossed $\bar{\mathcal{P}}_j$. Now, we have that the defender arrives at some point on $\bar{\mathcal{P}}_j$ and can defend it afterwards; therefore $\bar{\mathcal{P}}_j$ is strongly defendable.

Definition 7: Given joint initial condition $\mathbf{x}^0$ and a (strongly) defendable path $\bar{\mathcal{P}}(e_a, e_b)$, call the region $\Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$ the **(strongly) defendable region given** $\bar{\mathcal{P}}(e_a, e_b)$. Define the union of all (strongly) defendable regions given paths to be the **(strongly) defendable region**,



(a) The defender can defend the initial Voronoi line by mirroring the attacker's movement.



(b) If the defender tries to arrive at some point p, and then strongly defend it, then the attacker will arrive at one of the anchor points before the defender.

Fig. 5: The Voronoi line is a path of defense that is defendable but not strongly defendable.

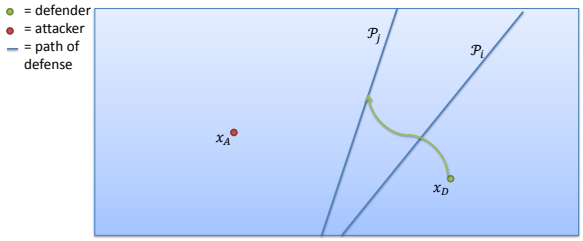


Fig. 6: $\bar{\mathcal{P}}_i$ is strongly defendable if $\bar{\mathcal{P}}_j$ is strongly defendable.

denoted ($\mathcal{D}_S$) $\mathcal{D}$:

$$\begin{aligned} \mathcal{D} &= \bigcup_{\bar{\mathcal{P}}(e_a,e_b) \text{ defendable}} \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a,e_b)\right) \\ \mathcal{D}_S &= \bigcup_{\bar{\mathcal{P}}(e_a,e_b) \text{ strongly defendable}} \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a,e_b)\right) \end{aligned} \tag{21}$$

For a given joint initial condition $\mathbf{x}^0$, we express the conservatism of strong path defense as the ratio area($\mathcal{D}$)/area($\mathcal{D}_S$).

V. The Path Defense Solution to the Two Player Reach-Avoid Game

In section II, we formulated the two-player reach-avoid game, and in section III-A, we described how to solve the game by solving a 4D HJI PDE. In this section, we will present an alternative, more efficient way of solving the two-player reach-avoid game conservatively. The new method we present is based on the path defense game from section IV and only involves solving the 2D Eikonal equation, which can be solved efficiently using the fast marching method.

In section IV, we described the path defense game as a reach-avoid game with the target set $\mathcal{T} = \Omega \backslash \mathcal{S}_A\left(\bar{\mathcal{P}}(e_a, e_b)\right)$. We now consider the case where the target set can be any arbitrary set. We will divide our analysis into three different cases depending on the domain: convex, simply connected non-convex, and general non-convex.

A. Convex Domain

Likely won't include this in the conference paper.

In a convex domain, the two player reach-avoid game can be solved exactly without solving an HJI PDE.

Lemma 3: In a two player reach-avoid game in a convex domain Ω, the defender wins if and only if the target set is contained in the Voronoi cell of the defender.

proof 3: Suppose part of the target set $\mathcal{T}$ lies in the attacker's Voronoi cell. Then, by definition of the Voronoi cell, the attacker can reach that part of $\mathcal{T}$ before the defender can exit the defender's Voronoi cell. Therefore, the attacker wins by simply taking the shortest path $\mathcal{P}(x_A^0, \mathcal{T})$ to reach the target without being captured. The payoff of the game is $t_A(x_A^0, \mathcal{T})$.

For the other direction, first denote the orthogonal projection of the attacker's position and of the defender's position onto the Voronoi line x_A^{op} and x_D^{op}, respectively. Note that $x_A^{op} = x_D^{op}$ initially. Consider the attacker's (defender's) velocity components in the direction $\mathcal{P}(x_A, x_A^{op})$, ($\mathcal{P}(x_D, x_D^{op})$, respectively) and in the direction perpendicular to it. Denote these components $\dot{x}_A^{\parallel}$ and $\dot{x}_A^{\perp}$ ($\dot{x}_D^{\parallel}$ and $\dot{x}_D^{\perp}$ for the defender).

Suppose the entire target set lies in the defender's Voronoi cell. The following defender strategy ensures that the attacker never crosses the Voronoi line (and hence the Voronoi line is defendable), and thus never reach the target. Given any $\dot{x}_A^{\perp}$, choose $\dot{x}_D^{\perp} = \dot{x}_A^{\perp}$, and use the remaining speed in the direction $\dot{x}_D^{\parallel}$. This ensures that $x_A^{op} = x_D^{op}$ for all time, and that the Voronoi line does not change if $\dot{x}_A^{\parallel} \geq 0$ and moves towards the attacker otherwise.

Figure 7 illustrates lemma 3.

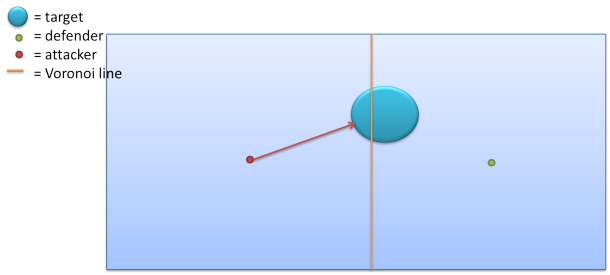
B. Simply Connected Non-Convex Domain

First paragraph not needed in conference paper.

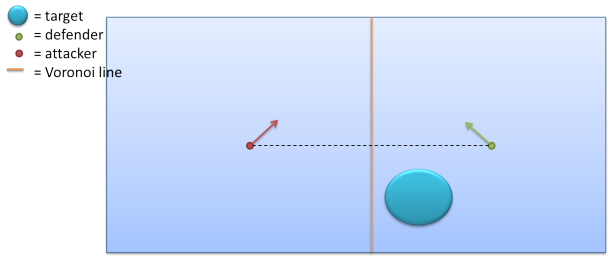
(In a non-convex domain, lemma 3 no longer holds, because the orthogonal projection of the attacker may no longer be unique. Figure 8 shows an example where x_A is equidistant to all points on the Voronoi line along e_a, e_c, while the defender's orthogonal projection is on e_c.)

To solve the reach-avoid problem in a non-convex domain efficiently and conservatively for the defender, we introduce the path defense solution, which leverages the idea of path defense described in section IV: if the target set is enclosed by some defendable (in particular, strongly defendable) path $\bar{\mathcal{P}}(e_a, e_b)$ for some e_a, e_b, then the defender can win the game.

Naively, one could fix e_a, then search all other points on $e_b \in \partial\Omega$ to find a defendable path. If a defendable path is found, then the defender wins the game; if not, try another e_a. However, by the argument in section IV-C, only paths of defense $\bar{\mathcal{P}}(e_a, e_b)$ that touch the target set need to be checked. Therefore, we can pick e_a, which will determine e_b. Then, check whether $\bar{\mathcal{P}}(e_a, e_b)$ is defendable.



(a) If part of the target set is in the attacker's Voronoi cell, the attack can win the game in minimum time by taking the shortest path to the target.



(b) If the target set is entirely within the Voronoi cell of the defender, the defender can "mirror" the attacker's control to defend the Voronoi line, and thus prevent the attacker from reaching the target.

Fig. 7: An illustration of the proof of lemma 3.

As noted in section IV, checking whether a path of defense is defendable is computationally expensive. Instead, we check whether each path is strongly defendable. This adds more conservatism towards the defender, but makes computation much more tractable.

If some strongly defendable path $\bar{\mathcal{P}}(e_a, e_b)$ encloses the target set, then the defender's strategy would be to first go to $p^* \in \bar{\mathcal{P}}(e_a, e_b)$, then move towards x_A/e_a or x_A/e_b until the level set image is captured. Finally, the defender can simply track the captured level set image. Note that this is a "semi-open-loop" strategy. The first part of the defender's control strategy is to move to p^* regardless of what the attacker does; this is an open-loop strategy. The second part of the defender's control strategy is to track the attacker's level set image, which depends on how the attacker moves; this is a closed-loop strategy.

C. General Non-Convex Domain

Currently, same as simply connected, except possibly more conservative, since some important paths of defense may not be checked.

D. Conservatism of Path-Defense

The path defense solution of the reach-avoid game is conservative for the defender in the sense that if a strongly defendable path of defense that encloses the target set is not found, we cannot conclude that the defender has no strategy to win the game.

To quantify the conservatism, we first specify the initial position of the attacker x_A^0, and then compute the winning region for the defender given x_A^0, denoted $\mathcal{D}$. Using the path

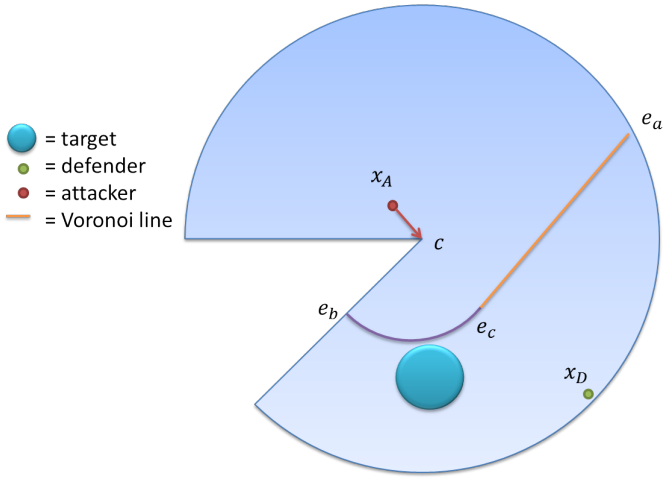


Fig. 8: In a non-convex domain, the orthogonal projection of the attacker may not be unique. In this case, x_A is equidistant to all points along the segment of the Voronoi line e_a, e_c.

defense solution, this region is given by the union over all paths of defender's winning regions given a path:

$$\mathcal{D} = \bigcup_{e_a, e_b} \left(\mathcal{G}\left(e_a, e_b\right)\right) \tag{22}$$

In a simply connected domain, the paths of defense are parameterized by only one anchor point, as the other anchor point is determined by the first. In this case,

$$\mathcal{D} = \bigcup_{e_a(e_b)} \left(\mathcal{G}\left(e_a, e_b\right)\right) \tag{23}$$

Ideally, we would like to compare the path defense winning region $\mathcal{D}$ to the winning region we obtain from the HJI solution by taking the ratio of the areas of the two regions. However, to reduce computation and avoiding solving an HJI PDE, we take the ratio of the area of $\mathcal{D}$ to the area of the entire domain as a measure of conservatism. This is a conservative estimate of how conservative the path defense solution is.

VI. Maximum Matching

Taken from ACC 2014 paper, shorten We can determine whether the attacker can win the multiplayer reach-avoid game by combining the solution to the two player game and maximum matching [20], [21] from graph theory as follows:

1) Construct a bipartite graph with two sets of nodes $\{P_{A_i}\}_{i=1}^N, \{P_{D_i}\}_{i=1}^N$, where each node represents a player.
2) For each P_{D_i}, determine whether P_{D_i} can win against P_{A_j}, for all j using strong path defense.
3) Form a bipartite graph: Draw an edge between P_{D_i} and P_{A_j} if P_{D_i} wins against P_{A_j}
4) Run any matching algorithm to find a maximum matching in the graph. This can be done using, for example, a linear program [20], or the Hopcroft-Karp algorithm [21].

After finding a maximum matching, we can determine whether the defending team can win as follows. After constructing the bipartite graph, if the maximum matching is of size k, then the defending team would be able to prevent k attackers from reaching the target. In particular, if the maximum matching is of size at least $N - m + 1$, then the attacking team would only be able to send at most $m - 1$ attackers to the target and thus the defending team would win.

May even cut out control strategy?

The optimal strategy for the defenders can be obtained from (14). If the i^{th} defender P_{D_i} is assigned to defend against the j^{th} attacker P_{A_j} by the maximum matching, then the P_{D_i} would use the semi-open-loop strategy outlined in section V-B to guarantees that P_{A_j} never reaches the target

The entire procedure of applying maximum matching to the strong path defense calculations is illustrated in Figure 9.

Our solution to the multiplayer reach-avoid game is an approximation to the optimal solution that would be obtained by directly solving the $4N$ dimensional HJI PDE obtained in Section III-A; it is conservative for the defending team for a couple of reasons. By creating defender-attacker pairs, each defender restricts its attention to only one opposing player. Also, because the defender uses the suboptimal semi-open-loop strategy within each defender-attacker pair.

If no suitable matching is found, the defending team is not guaranteed to lose, as the defending team could potentially win without using a strategy that creates defender-attacker pairs and without using strong path defense. Nevertheless, our solution is able to overcome the curse of dimensionality to approximate an intractable reachability calculation, and is useful in many game configurations.

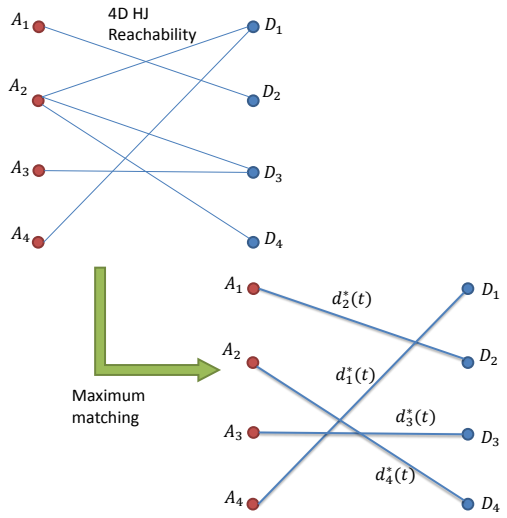


Fig. 9: An illustration of using maximum matching to solve the multiplayer reach-avoid game. A bipartite graph is created based on results of the 4D HJ reachability calculation. Then, a maximum matching of the bipartite graph is found to optimally assign defender-attacker pairs.

A. Time-Varying Defender-Attacker Pairings

A bit cumbersome with the path defense idea.

The procedure outlined in Section VI assigns an attacker to each defender that is part of a maximum pairing in an open-loop manner: the assignment is done in the beginning of the game, and does not change during the course of the game. However, the bipartite graph and its corresponding maximum matching can be updated as the players change positions during the game.

1) Given the position of each defender x_{D_i} and each attacker x_{A_j}, determine whether x_{A_j} can win for all j.
2) Construct the bipartite graph and find its maximum matching to assign an attacker to each defender that is part of the maximum matching.
3) For a short, chosen duration Δ, compute the optimal control input and trajectory for each defender that is part of the maximum matching via Equation (14). For the rest of the defenders and for all attackers, compute the trajectories assuming some control function.
4) Repeat the procedure with the new player positions.

As $\Delta \to 0$, the above procedure computes a bipartite graph and its maximum matching as a function of time. Whenever the maximum matching is not unique, the defenders can choose a different maximum matching and still be guaranteed to prevent the same number of attackers from reaching the target. As long as each defender uses the optimal control input given in Equation (14), the size of the maximum matching can never decrease as a function of time.

On the other hand, it is possible for the size of the maximum matching to increase as a function of time. This occurs if the joint configuration of the players becomes such that the resulting bipartite graph has a bigger maximum matching than before, which may happen since the size of the maximum matching only gives an upper bound on the number of attackers that are able to reach the target. Furthermore, because of the curse of dimensionality, there is no numerically tractable way to compute the joint optimal control input for the attacking team, so a suboptimal strategy from the attacking team can be expected, making an increase of maximum matching size likely. Determining defender control strategies that optimally promote an increase in the size of the maximum matching would be an important step towards the investigation of cooperation, and will be part of our future work.

VII. Computation Results

Need new results for path defense

We illustrate our reachability and maximum matching approach in the two examples below. The HJ reachable sets are calculated using the Level Set Toolbox [24] developed at the University of British Columbia. We calculated $\mathcal{RA}_\infty(R, A)$ by incrementing T until $\mathcal{RA}_T(R, A)$ converges. Each computation of $\mathcal{RA}_\infty(R, A)$, done on a $45 \times 45 \times 45 \times 45$ grid, took approximately 30 minutes on a Lenovo T420s laptop with a Core i7-2640M processor.

The two examples are shown in Figures 9 and 9. In both examples, there are four attackers and four defenders playing on a square domain with obstacles; the defenders have a capture radius of 0.1 units. In example 1, all players have equal speeds ($v_D = v_A$). In example 2, the defenders are twice as fast as the attackers, who have equal speeds ($v_D = 2v_A$).

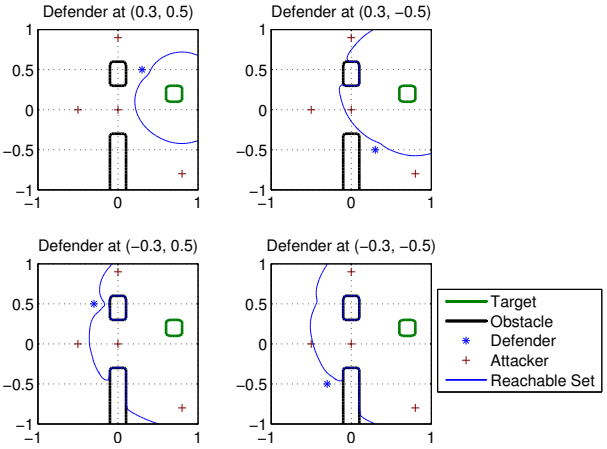
$\mathcal{RA}_\infty(R, A)$ is a 4D set that represents the joint configurations in which the attacker wins the game. To visualize the 4D set in 2D, we view the reachable set at the slices representing the positions of particular players. Figure 10a and 10a show boundaries of $\mathcal{RA}_\infty(R, A)$ with fixed defender positions. In each subplot, attackers which are closer to the target set than the reachable set boundary is win against the particular defender. For example, in the right top subplot of Figure $10a$, the defender at $(0.3, -0.5)$ loses to the attacker at $(0, 0)$, but wins against the other three attackers.

Similarly, Figures 10b and 10b show boundaries of $\mathcal{RA}_\infty(R, A)$ with fixed attacker positions. Defenders which are closer to the target set than the reachable set boundary win against the particular attacker. For example, in the top left subplot of Figure 10b, the attacker at $(0.8, 0.6)$ wins against the defender at $(-0.7, 0.2)$ but loses against the other three defenders.

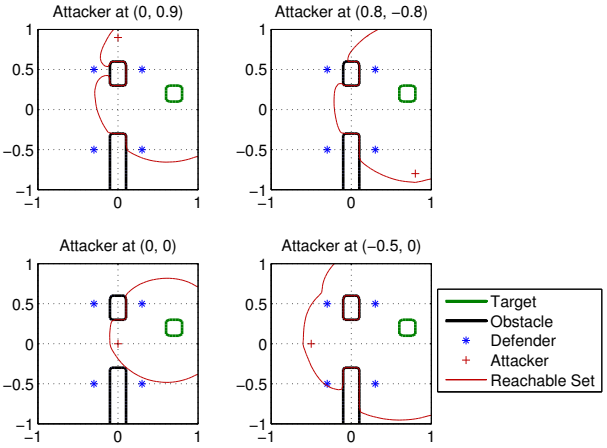
Figures 9a and 9a show the resulting bipartite graphs (edges shown as thin solid blue lines) and the maximum matching (edges shown as thick dashed blue lines) after applying the algorithm described in Section VI. In example 1, the maximum matching is of size 3, which means that at most 1 attacker is able to reach the target. So if the game requires $m > 1$ attackers to reach the target for the attacking team to win the game, then the defending team would win. In example 2, we have a perfect matching, so no attacker is able to reach the target.

Figure 10 shows the result of a 4 vs. 4 reach-avoid game simulation being played out in a course of 0.6 time units to illustrate the potential usefulness of time-dependent defender-attacker pairings. Every $\Delta = 0.005$ time units, a bipartite graph and its maximum matching are computed according to the algorithm in Section VI-A. The defender that is not part of the maximum matching plays optimally against the closest attacker. The attackers use the suboptimal strategy of taking the shortest path to the target while steering 0.125 units clear of the obstacles and disregarding the control inputs of other players in the game.

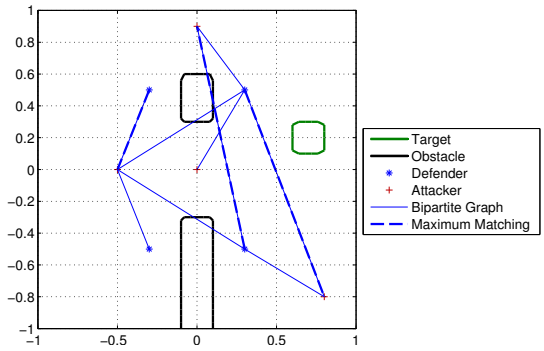
The initial size of the maximum matching is 3. At $t = 0.4$, because the attackers have not been playing optimally, the attacker at $(-0.50, 0.16)$ becomes in a losing position against the defender at $(-0.26, -0.34)$. Thus, the maximum matching now assigns the attacker at $(-0.50, 0.16)$ to the defender at $(-0.26, -0.34)$. At the same time, the defender at $(-0.09, -0.22)$ switches from defending the attacker at $(-0.50, 0.16)$ to defending the attacker at $(-0.27, 0.14)$, creating a perfect matching and preventing any attacker from reaching the target.



(a) Slices of reachable set at the positions of the four defenders. The defender is guaranteed to win against any attacker who is farther away from the target than the reachable set boundary is.

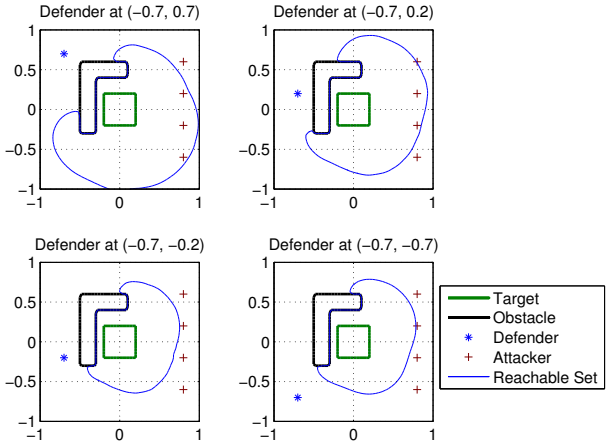


(b) Slices of reachable set at the positions of the four attackers. The attacker is guaranteed to win against any defender who is farther away from the target than the reachable set boundary is.

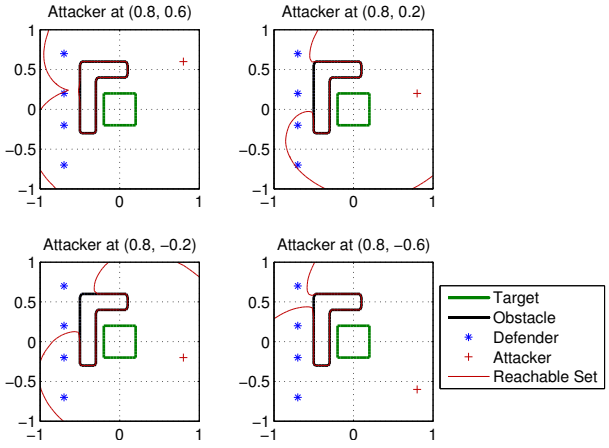


(a) Bipartite graph and maximum matching. Each edge, shown as a thin solid blue line, connects a defender to an attacker against whom the defender is guaranteed to win, creating a bipartite graph. The dashed thick blue lines show a maximum matching of the bipartite graph. Here, a maximum matching of size 3 indicates that at most one attacker can reach the target.

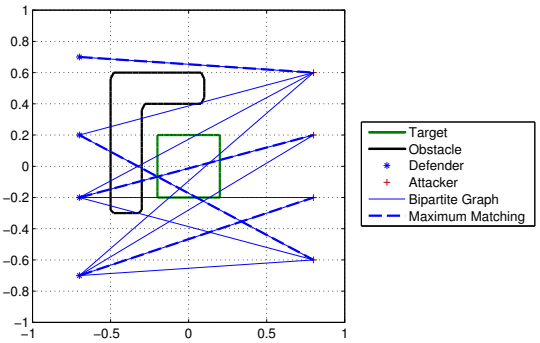
Fig. 9: Example 1: A 4 vs. 4 reach-avoid game in which all players have equal maximum speeds and the defenders have a capture radius of 0.1 units.



(a) Slices of reachable set at the positions of the four defenders. The defender is guaranteed to win against any attacker who is farther away from the target than the reachable set boundary is.



(b) Slices of reachable set at the positions of the four attackers. The attacker is guaranteed to win against any defender who is farther away from the target than the reachable set boundary is.



(a) Bipartite graph and maximum matching. Each edge, shown as a thin solid blue line, connects a defender to an attacker against whom the defender is guaranteed to win, creating a bipartite graph. The dashed thick blue lines show a maximum matching of the bipartite graph. Here, a maximum matching of size 3 indicates that at most one attacker can reach the target.

Fig. 9: Example 2: A 4 vs. 4 reach-avoid game in which players of the same team have the same maximum speed, and the maximum speed of the defenders is twice that of the attackers. The defenders have a capture radius of 0.1 units.

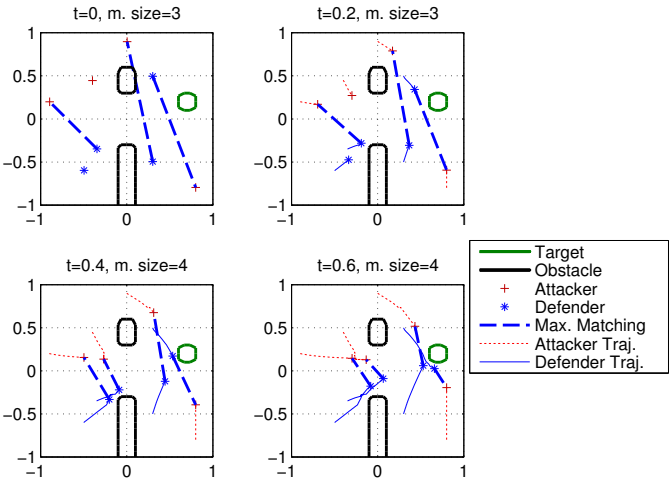


Fig. 10: An illustration of how the size of maximum matching can increase over time. Throughout the game, the defenders are updating the bipartite graph and maximum matching via the procedure described in Section VI-A. Because the attacking team is not playing optimally, the defending team is able to find a perfect matching after $t = 0.4$ (bottom plots) and prevent all attackers from reaching the target.

VIII. Conclusions and Future Work

By applying maximum matching to the solution to the two player reach-avoid game, we were able to approximate the solution to the multiplayer reach-avoid game without significant additional computation overhead over the two player computation. By solving a single 4D HJI PDE, we obtained all possible pairings between the defenders and attackers. Then, a maximum matching algorithm determines the pairing that prevents the maximum number of attackers from reaching the target. This way, we were able to analyze

the multiplayer reach-avoid game without directly solving the corresponding high dimensional, numerically intractable HJI PDE. Calculating time-varying defender-attacker pairings allows the defending team to potentially increase the size of the maximum matching over time.

An immediate extension of this work is to investigate defender strategies that optimally promote an increase in the size of maximum matching. This would be a step towards real-time defending team cooperation, in which each defender is not only responsible for preventing a particular assigned attacker from reaching the target, but also enabling other defenders to defend previously seemingly undefendable attackers. Many other extensions of this work naturally arise, including analyzing reach-avoid games with unfair teams with different numbers of players on each team, more complex player dynamics, or partial observability.

REFERENCES

[1] Department of the Air Force, "United states air force unmanned aircraft systems flight plan 2009-2047," *oai.dtic.mil*, Jan 2009.

[2] H. Erzberger, "Automated conflict resolution for air traffic control," *25 th International Congress of the Aeronautical Sciences*, Jan 2006.

[3] A. Madrigal, "Autonomous robots invade retail warehouses," http://www.wired.com/wiredscience/2009/01/retailrobots/.

[4] H. Huang, "Reachability-based control for human and autonomous agents in adversarial games," Ph.D. dissertation, Stanford University, 2012.

[5] H. Huang, J. Ding, W. Zhang, and C. Tomlin, "A differential game approach to planning in adversarial scenarios: A case study on capture-the-flag," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 1451–1456.

[6] R. D'Andrea and M. Babish, "The roboflag testbed," *Proceedings of the American Control Conference*, vol. 1, pp. 656–660, 2003.

[7] M. Earl and R. D'Andrea, "A decomposition approach to multi-vehicle cooperative control," *Robotics and Autonomous Systems*, vol. 55, no. 4, pp. 276–291, 2007.

[8] M. Campbell, R. D'Andrea, D. Schneider, A. Chaudhry, S. Waydo, J. Sullivan, J. Veverka, and A. Klochko, "Roboflag games using systems based, hierarchical control," *Proceedings of the American Control Conference*, vol. 1, pp. 661–666, 2003.

[9] S. Waydo and R. Murray, "Vehicle motion planning using stream functions," *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA'03*, pp. 2484–2491, 2003.

[10] R. Parasuraman, S. Galster, P. Squire, H. Furukawa, and C. Miller, "A flexible delegation-type interface enhances system performance in human supervision of multiple robots: Empirical studies with roboflag," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 35, no. 4, p. 481, 2005.

[11] G. Chasparis and J. Shamma, "Linear-programming-based multi-vehicle path planning with adversaries," *American Control Conference, 2005. Proceedings of the 2005*, pp. 1072–1077, 2005.

[12] J. Sprinkle, J. Eklund, H. Kim, and S. Sastry, "Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller," *Decision and Control, 43rd IEEE Conference on*, 2004.

[13] J. McGrew, J. How, L. Bush, B. Williams, and N. Roy, "Air combat strategy using approximate dynamic programming," *AIAA Guidance, Navigation, and Control Conference*, Aug 2008.

[14] R. Isaacs, *Differential Games*. New York: Wiley, 1967.

[15] I. Mitchell, A. Bayen, and C. Tomlin, "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, vol. 50, no. 7, pp. 947–957, July 2005.

[16] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996. [Online]. Available: http://www.pnas.org/content/93/4/1591.abstract

[17] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, 2002, ISBN: 978-0-387-95482-0.

[18] J. Ding, J. Sprinkle, S. S. Sastry, and C. J. Tomlin, "Reachability calculations for automated aerial refueling," in *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008.

[19] M. Chen, Z. Zhou, and C. Tomlin, "Multiplayer reach-avoid games," in *American Control Conference 2014*, 2014.

[20] A. Schrijver, *Combinatorial Optimization : Polyhedra and Efficiency (Algorithms and Combinatorics)*. Springer, Jul. 2004. [Online]. Available: http://www.worldcat.org/isbn/3540204563

[21] M. Karpinski and W. Rytter, *Fast parallel algorithms for graph matching problems*. New York, NY, USA: Oxford University Press, Inc., 1998.

[22] Z. Zhou, R. Takei, H. Huang, and C. Tomlin, "A general, open-loop formulation for reach-avoid games," in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, 2012, pp. 6501–6506.

[23] T. Basar and G. Olsder, *Dynamic Noncooperative Game Theory*, 2nd ed. Philadelphia, PA: SIAM, 1999.

[24] I. Mitchell, *A Toolbox of Level Set Methods*, 2009, http://people.cs.ubc.ca/ mitchell/ToolboxLS/index.html.

[25] M. G. Crandall and P.-L. Lions, "Viscosity solutions of Hamilton-Jacobi equations," *Transactions of the American Mathematical Society*, vol. 277, no. 1, pp. 1–42, 1983.

[26] I. Mitchell, "Application of level set methods to control and reachability problems in continuous and hybrid systems," Ph.D. dissertation, Stanford University, 2002.

[27] J. Lygeros, C. Tomlin, and S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, no. 3, pp. 349 – 370, 1999.

[28] C. Tomlin, J. Lygeros, and S. Shankar Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 949 –970, Jul 2000.