Modeling and control of a multi-agent system using mixed integer linear programming

Matthew G. Earl[1] Raffaello D'Andrea[2]

Abstract

The RoboFlag competition was proposed by the second author as a means to study cooperative control of multi-vehicle systems. Here we study a simplified version of the competition that we model as a hybrid system. We use an optimization approach to synthesize control policies by solving mixed integer linear programs.

1 Introduction

We are interested in using collections of agents in a cooperative manner to autonomously perform complex tasks. This is not a new idea given that there exist autonomous multi-agent systems today that work fairly well—Cornell's RoboCup team [4] is a good example. But, these systems lack prevalence which is in part due to two important issues. First, for many applications, the control strategy for such systems must meet performance and safety criteria which currently is hard to guarantee. And second, developing strategies for large complex systems is extremely difficult. Therefore, there is a need for systematic analysis and synthesis tools. Numerous researchers from computer science, controls, and the operations research communities have contributed valuable work toward this end—see the papers [1, 3, 6, 8] and the references therein; in fact, an interdisciplinary approach seems ideal. In this paper we present first steps toward strategy synthesis, via an optimization approach inspired by [2], within the context of a new test-bed for studies in cooperative control. In [7] similar ideas, developed independently, are used for aircraft trajectory planning.

The RoboCup competition is a valuable means to study multi-agent systems but, due to the small field and fast pace of the game, complex strategies are usually not optimal. Due to this the second author has proposed RoboFlag as a companion competition in hopes to provide a fertile test-bed for advances in cooperative control technology involving complex strategies. The RoboFlag competition is described in detail on the second authors web page; here we give a brief description

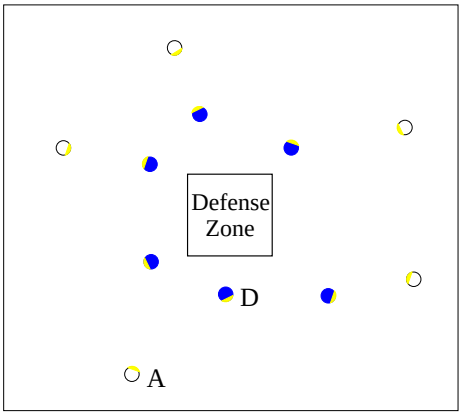

Figure 1: The defenders D guard the defense zone from the attacking robots A .

to give a flavor for the game. The competition is between two teams of eight robots appease and is similar to the well known games "capture the flag" and "paintball." The objective is to infiltrate your opponents territory, get their flag, and bring it to your home base while, at the same time, trying to thwart your opponent from capturing your own flag. While pursuing the objective there are a number of other factors to worry about. Some parts of the field are off limits, moving obstacles and golf balls shot by the opponents must be avoided, and each robot has a limited amount of fuel. This game is complicated and it would be a daunting task to develop winning strategies at this point in time. As a first step we consider a vastly simplified version of the competition which we call the RoboFlag Drill. The drill involves two teams of robots, the attackers and the defenders, on a playing field with a region at its center called the defense zone. The attackers are drones directed toward the defense zone. The objective for the defenders is to thwart the attackers from entering the defense zone by intercepting each attacker before it enters the zone. Once an attacker enters the defense zone or is intercepted by a defender it remains stationary for the remainder of the drill. While pursuing its objective defenders must avoid collisions with other defenders and obstacles as well as avoid entering the defense zone which is off limits to defending robots.

To intelligently perform the drill the defenders need a strategy that is consistent with the objective and ide-

[1]Cornell University, e-mail: mge1@cornell.edu
[2]Cornell University, e-mail: rd29@cornell.edu

ally optimal with respect to the objective. We assume the strategy is implemented with a centralized controller with perfect knowledge of the system, perfect access to all states, and with the ability to transmit control signals to the defenders instantaneously. The controller needs to figure out the inputs to provide each defending robot so that the objective is achieved. To achieve this we pose the problem as an optimization. We seek a set of control inputs that minimize the number of attackers that enter the defense zone over the duration of the drill and, in addition, is consistent with the system dynamics (robot dynamics) and the constraints (no collisions, etc.).

We model the RoboFlag Drill as a system composed of continuous and discrete states (a hybrid system) with linear dynamics subject to inequality constraints and logical rules. We convert this model to a mixed integer linear program (MILP) using techniques that can be found in [2, 10, 9]. We solve the resulting optimization problem using ILOG CPLEX [5].

2 Modeling

2.1 The defense zone

The region which the attackers are trying to infiltrate and the defenders are trying to guard, but must not enter, is called the defense zone as shown in Figure 1. The defense zone G is a square centered at the origin of the playing field and is defined by

$$G := \{(x, y) : (-b \leq x \leq b) \wedge (-b \leq y \leq b)\} \qquad (1)$$

where b is a positive constant.

2.2 Defenders

There are N_D defenders that participate in the RoboFlag Drill. Each defender, denoted D_i where $i \in \{1, \ldots, N_D\}$, has state defined by

$$\mathbf{x}_i(t) := (x_i(t), y_i(t), \dot{x}_i(t), \dot{y}_i(t)) \qquad (2)$$

where $(x_i(t), y_i(t))$ are the coordinates at time t of D_i with respect to the origin of the playing field and $\dot{x}_i(t)$ and $\dot{y}_i(t)$ are the respective velocity components. We model the dynamics of D_i with a set of linear ordinary differential equations of the form

$$\begin{aligned} \ddot{x}_i(t) + \alpha \dot{x}_i(t) &= \beta u_{xi}(t) \\ \ddot{y}_i(t) + \alpha \dot{y}_i(t) &= \beta u_{yi}(t) \end{aligned} \qquad (3)$$

where $u_{xi}(t)$ and $u_{yi}(t)$ are the control inputs and α and β are constant parameters. The initial condition is given by

$$\mathbf{x}_i(0) = (x_{0i}, y_{0i}, \dot{x}_{0i}, \dot{y}_{0i}) \qquad (4)$$

This is reasonable since, as described in [4], a local controller can be implemented so that the dynamics of the vehicle obey equation (3). Since these dynamics are linear we can solve for $\mathbf{x}_i(t)$ explicitly.

The state and the input of D_i are both subject to constraints. Since the defending robots have limited control authority we constrain the inputs to the box

$$\begin{aligned} -u_{max} \leq u_{xi} \leq u_{max} \\ -u_{max} \leq u_{yi} \leq u_{max}. \end{aligned} \qquad (5)$$

More complicated regions can be modeled by a set of linear inequalities defining a polygon that approximates the desired region.

Defender D_i is forbidden to enter the defense zone G, therefore, we impose the state constraint $(x_i(t), y_i(t)) \notin G$, i.e.

$$(x_i(t) < -b) \vee (x_i(t) > b) \vee (y_i(t) < -b) \vee (y_i(t) > b). \qquad (6)$$

We associate with each defender D_i a region rigidly attached to D_i and centered at its coordinates $(x_i(t), y_i(t))$. This region, denoted $I_i(t)$, is a box defined by

$$I_i(t) := \{(x, y) : (-c \leq x - x_i(t) \leq c) \wedge (-c \leq y - y_i(t) \leq c)\} \qquad (7)$$

where c is a positive constant. The region serves two purposes:

1. It is used to avoid collisions between defending robots, i.e. the coordinates of all other defending robots must not lie within $I_i(t)$. This condition imposes the additional state constraints: for all $k \in \{1, \ldots, N_D\}$ such that $k \neq i$, $(x_k(t), y_k(t)) \notin I_i(t)$, or equivalently:

$$\begin{aligned} &\forall\ k \in \{1, \ldots, N_D\},\ k \neq i \\ &(x_k(t) - x_i(t) < -c) \vee (x_k(t) - x_i(t) > c) \\ &\qquad \vee (y_k(t) - y_i(t) < -c) \vee (y_k(t) - y_i(t) > c) \end{aligned} \qquad (8)$$

2. It is used to determine if an attacker is intercepted. If an attacking robot is within region $I_i(t)$ and not within the defense zone G then it is considered intercepted which we define precisely in Section 2.3.

2.3 Attackers

There are N_A attacking robots that participate in the RoboFlag Drill. Each attacker, denoted A_j where $j \in \{1, \ldots, N_A\}$, has state composed of continuous and discrete components. The continuous component is given by

$$\mathbf{p}_j(t) := (p_j(t), q_j(t), \dot{p}_j(t), \dot{q}_j(t)) \qquad (9)$$

where $(p_j(t), q_j(t))$ are the coordinates at time t of A_j with respect to the origin of the playing field and $\dot{p}_j$

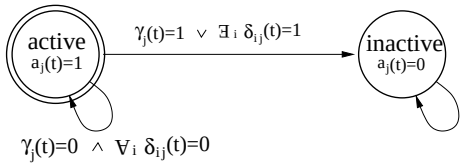

Figure 2: State machine for attacker A_j.

and $\dot{q}_j$ are the respective velocity components which we assume to be constant in this paper. The discrete component $a_j(t) \in \{0,1\}$ denotes the robot's status where $a_j(t) = 1$ signifies A_j is active and $a_j(t) = 0$ signifies A_j is inactive.

We introduce the auxiliary binary variable $\delta_{ij}(t) \in \{0,1\}$ to represent whether A_j is inside or outside defender D_i's region $I_i(t)$. More precisely, we define

$$\delta_{ij}(t) = 1 \iff (p_j(t), q_j(t)) \in I_i(t) \tag{10}$$

for all $i \in \{1, \ldots, N_D\}$ and $j \in \{1, \ldots, N_A\}$. Similarly, we introduce the auxiliary binary variable $\gamma_j(t) \in \{0,1\}$ to represent whether A_j is inside or outside the defense zone G. More precisely we define

$$\gamma_j(t) = 1 \iff (p_j(t), q_j(t)) \in G \tag{11}$$

for all $j \in \{1, \ldots, N_A\}$.

Attacker A_j moves along a straight line with constant velocity when active ($a_j(t) = 1$) and with zero velocity when inactive ($a_j(t) = 0$). In discrete time these dynamics are captured by the equations

$$\begin{aligned} p_j(t+1) &= p_j(t) + \dot{p}_j a_j(t) \\ q_j(t+1) &= q_j(t) + \dot{q}_j a_j(t) \end{aligned} \tag{12}$$

with initial condition

$$\begin{aligned} p_j(0) &= p_{0j} \\ q_j(0) &= q_{0j}. \end{aligned} \tag{13}$$

Initially the attacker A_j is active, i.e.

$$a_j(0) = 1. \tag{14}$$

and it remains active until it enters the defense zone G or it is intercepted by a defender D_i. When one of these two events occur attacker A_j transitions to an inactive state and remains inactive for the remainder of the drill. This is precisely captured by the state machine in Figure 2 or the state equation:

$$a_j(t+1) = \begin{cases} 1 & \text{if } a_j(t) = 1 \wedge \gamma_j(t) = 0 \wedge (\forall i\ \delta_{ij}(t) = 0) \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

2.4 Objective

The defender's objective is to minimize the number of attackers that enter the defense zone. The number of attackers within the defense zone at any time t is given by

$$\sum_{j=1}^{N_A} \gamma_j(t) \tag{16}$$

therefore the objective can be captured precisely by minimizing the number of attackers within the defense zone at the final time of the drill, namely

$$\sum_{j=1}^{N_A} \gamma_j(T-1) \tag{17}$$

where T is the duration of the drill. We take T to allow each attacker, if not impeded, enough time to enter the defense zone G. To allow for this we simply take T to be

$$T = \max_j \left(\sqrt{\frac{r_j^2 + b^2}{\dot{p}_j^2 + \dot{q}_j^2}} \right) \tag{18}$$

where $r_j = \sqrt{p_{0j}^2 + q_{0j}^2}$.

We can also add a penalty on the control effort to the cost for more efficient control strategies in which case the cost to be minimized becomes:

$$\sum_{j=1}^{N_A} \gamma_j(T-1) + \sum_{t=0}^{T-1} \sum_{i=1}^{N_D} (|u_{xi}(t)| + |u_{yi}(t)|) \tag{19}$$

3 Optimization

Using techniques for converting logic into inequalities such as those outlined in [2, 10, 9] we convert this optimization problem into a MILP of the form

$$\begin{aligned} &\min_w f^T w \\ &\text{such that } A_1 w \le b_1, A_2 w = b_2 \end{aligned} \tag{20}$$

where w is a vector of the control inputs and auxiliary binary and continuous variables, introduced by the conversion process, for each time step $t \in \{0, 1, \ldots, T-1\}$.

We solve the MILP problem using ILOG CPLEX [5]. Figure 3 shows an example problem in which three defenders successfully defending the zone from eight attaching robots. Figure 4 shows the control inputs to the defender that corresponds to the lower plot in Figure 3. Note that in this example we did not penalize the control action. This particular problem consists of 4040 integer variables, 400 continuous variables, and 13580 constraints. It was solved in 244 seconds on a PIII 866MHz computer running Linux.

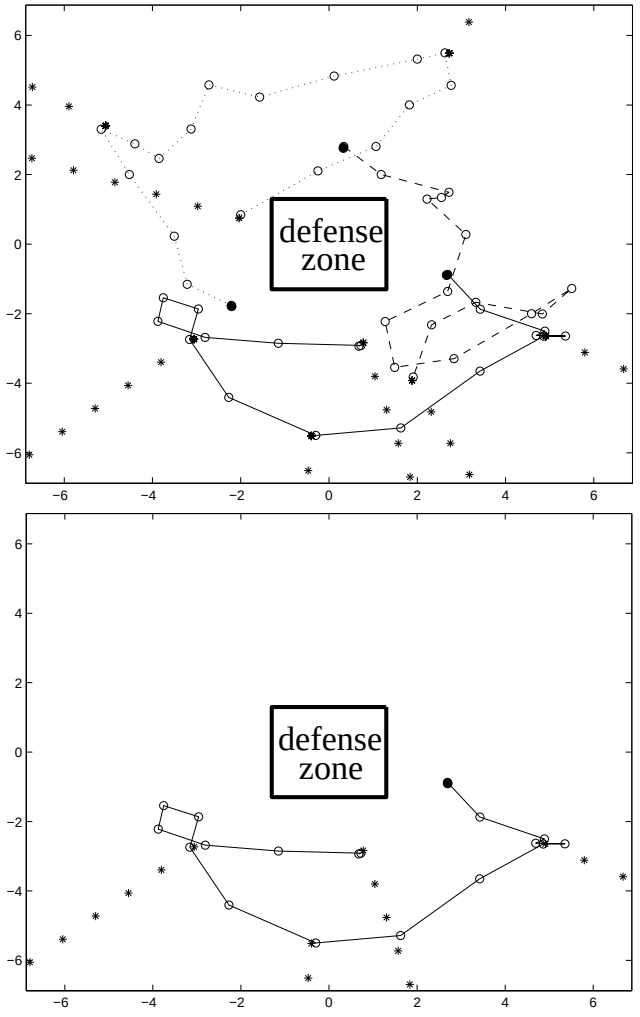

Figure 3: Three defending robots (circles) successfully defend the zone from eight attacking robots (stars). The filled circles denote the initial positions of the defending robots. The lower figure shows the trajectories of a single defending robot and the attackers it intercepts.

4 Conclusions and future work

For the simplified problem at hand we have successfully generated a strategy to minimize the objective, but the procedure is computationally intensive and assumes a static environment. Recently we have augmented this procedure for dynamically changing environments by implementing a replanning strategy in which we perform the above optimization online at every time step. This procedure can handle more general attacker motion since the constant velocity vector assumption is reasonable for short time steps. However, the procedure is limited computationally since the problem does not scale well with increased numbers of agents.

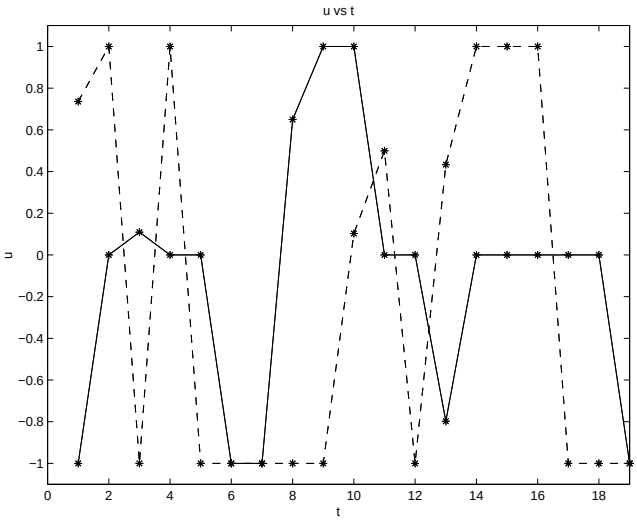

Figure 4: Here we plot u_x (dashed line) and u_y (solid line) for the defender shown in the lower plot of Figure 3.

Acknowledgments

We thank Richard Murray and Jason Hickey from Caltech and Carla Gomes, Bart Selman, and Pritam Ganguly from Cornell for helpful discussions.

References

[1] R. Alur, A. Das, J. Esposito, R. Fierro, Y. Hur, G. Grudic, V. Kumar, I. Lee, J. P. Ostrowski, G. Pappas, J. Southall, J. Spletzer, and C. Taylor. A framework and architecture for multirobot coordination. *Experimental Robotics: LNCS Series*. Springer-Verlag, 2001.

[2] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, vol. 35, pp. 407-428, Mar. 1999.

[3] M. Bowling and M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*. 2002, to appear.

[4] D'Andrea, R., Kalmár-Nagy, T., Ganguly, P., Babish, M. The Cornell Robocup Team. In Kraetzschmar, G., Stone P., Balch T. editors: *Robot Soccer WorldCup IV, Lecture Notes in Artificial Intelligence*. Springer, 2001.

[5] ILOG, Inc. CPLEX 7.1
`http://www.ilog.com/products/cplex/`

[6] M. J. Mataric. Learning in Behavior-Based Multi-Robot Systems: Policies, Models, and Other Agents. In *Cognitive Systems Research, special issue on Multi-disciplinary studies of multi-agent learning*. Ron Sun, ed., 2(1), Apr 2001, 81-93.

[7] A. Richards and J. P. How. Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming. In *Proc. American Control Conf.*, 2002.

[8] P. Stone. *Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer*. MIT Press, 2000.

[9] F.D. Torrisi, A. Bemporad, and D. Mignone. HYSDEL—A language for describing hybrid systems. Technical Report AUT00-03, ETH Zurich, 2000. `http://control.ethz.ch/~hybrid/hysdel`

[10] H. P. Williams. *Model Building in Mathematical Programming*. John Wiley and Sons, Third Edition, 1993.