International Journal of Innovative Computing, Information and Control
Volume **13**, Number **6**, December **2017**

ICIC International ©2017 ISSN 1349-4198
pp. 1855–1872

MULTI-AGENT LEARNING IN THE GAME OF GUARDING A TERRITORY

Chidozie Vincent Analikwu and Howard Marvin Schwartz

Department of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive, Ottawa, ON, K1S 5B6, Canada
chidozie.analikwu@carleton.ca; schwartz@sce.carleton.ca

Received April 2017; revised August 2017

Abstract. *We consider the problem of having a team of guards to learn a joint cooperative strategy to pursue and capture a high speed invader before the invader can reach a territory. In this scenario, the invader is also simultaneously learning its optimal strategy to avoid capture and get as close as possible to the territory. This conflict of interest between the learning agents makes the problem challenging. We adopt the guarding a territory game framework to model the problem, and consider the use of reinforcement learning, particularly the fuzzy actor-critic learning method, to train the players to find their optimal strategies simultaneously. To our knowledge, this is the first work to investigate the development of multi-agent learning for a high speed super invader in the game of guarding a territory. Simulation results from this study demonstrate that all the players are able to learn their optimal behaviors simultaneously.*

Keywords: Reinforcement learning, Fuzzy logic controller, Guarding a territory game, Multi-agent systems, Apollonius circle, Fuzzy actor-critic learning

1. Introduction.

The guarding a territory differential game was first introduced by Isaacs [1]. The game is played between two players (the invader and the guard). The goal of the guard is to intercept the invader as far as possible from the territory, whereas the goal of the invader is to avoid capture and get as close as possible to the territory. Practical applications of the game can be found in protecting critical infrastructures, such as nuclear facilities, transportation systems and emergency services, from physical attacks and in protecting international borders against illegal entries and smuggling activities.

The guarding a territory game has been investigated previously in [2, 3]. However, in these published literature [2, 3], the authors of the studies assumed that the players have a priori knowledge of their optimal behaviors. As such, these papers applied fixed algorithms with no learning. In this paper, we consider the other possibility of the game where the players do not have a priori knowledge of their optimal behaviors. We do so by investigating the use of a learning algorithm to train the players to find their optimal behaviors simultaneously.

Several papers [4, 5, 6] have investigated ways to use learning algorithms, particularly reinforcement learning algorithms, to solve differential game problems. The majority of these works applied the well-known Q-learning method [7]. The Q-learning method have yielded many useful results for problems with small finite state and action spaces. However, this approach is not suitable for problems with continuous state and action spaces. This is because the Q-learning method requires a priori discretization of the continuous spaces. As such, if a finely grained discretization is done so as to obtain good accuracy, this will result in heavy memory requirements and slow learning procedures [8, 9].

This paper investigates the use of the fuzzy actor-critic learning (FACL) method to solve the guarding a territory problem. The FACL method makes use of the fuzzy logic controller (FLC) to effectively deal with the problem of continuous state and action spaces. Some elements of this learning method were presented in [10, 11] for the case of a single guard against a low speed invader. In [10], the FACL method was used to train only the guard in the game. In this scenario, the invader followed a fixed strategy with no learning. Simulation results in [10] demonstrated that the guard was able to learn its optimal strategy to capture the invader in an effective way. The possibility of both the single guard and the low speed invader learning simultaneously was investigated in [11]. Simulation results presented in [11] demonstrated that both players, the invader and the guard, were able to learn their optimal strategies simultaneously. This work extends the approach presented in [10, 11] to investigate a more complex problem which involves two guards against a high speed invader. In this scenario, all three players have no a priori knowledge of their optimal behaviors and therefore are simultaneously learning. To our knowledge, this is the first work to investigate how a group of guards can work together to effectively capture a high speed super invader.

The main contributions of this paper are: (i) the proposal of a novel multi-agent learning method to train all the robot agents simultaneously; (ii) the investigation of a cooperative control strategy to capture a high speed super invader using multiple guards; (iii) the design of instantaneous reward functions for the learning process of the players; (iv) the choice of inputs that define the state.

The remainder of the paper is organized as follows. Preliminary concepts are presented in Section 2. Section 3 describes the FACL method and provides specifics for the design of the control system. We introduce the Apollonius circle and subsequently present the Apollonius circle approach to determine the optimal strategy of the players in Section 4. Section 5 presents the simulation results that demonstrate the performance of the learning agents. Detailed analyses and discussions of the simulation results will be provided. Finally, conclusions and future work are summarized in Section 6.

2. Preliminaries.

2.1. The guarding a territory differential game.

Isaacs' [1] guarding a territory game between two players is a zero-sum differential game played in the continuous domain. Adopting unicycle robots as our players in the game, the dynamics of any one of the players (the invader or the guard) is defined as [12]

$$\dot{x} = V\cos\theta \tag{1}$$

$$\dot{y} = V\sin\theta \tag{2}$$

$$\dot{\theta} = \omega \tag{3}$$

where V is the linear speed of the robot, ω is the angular speed of the robot, $[x, y]$ is the geometrical location of the robot and θ is the orientation of the robot with respect to the global x-axis.

In order to simplify the problem, the territory T is assumed to be a circle with center C_T and radius R_T. The payoff $P(\bar{u}_G, \bar{u}_I)$ of the game is the Euclidean distance between the position of the invader I and the territory T at game termination defined as

$$P(\bar{u}_G, \bar{u}_I) = \sqrt{(x_I - x_{C_T})^2 + (y_I - y_{C_T})^2} - R_T$$

where $\bar{u}_G$ and $\bar{u}_I$ are the strategies of the guard and the invader respectively. For simplicity, the game terminates when the distance between the invader and the guard is less than the capture radius R_C of the guard. Since the payoff implies how close the invader could get to the territory at game termination, it is rational that the invader chooses its strategy $\bar{u}_I$ to minimize the payoff and the guard $\bar{u}_G$ to maximize it. Our research focuses on investigating the use of a learning algorithm to train the invader and the guard simultaneously in order to find their respective optimal strategies u_I^*, u_G^*.

We made certain assumptions in the game. The game environment is obstacle-free. The game is played in a two-dimensional bounded space, such that if any player attempts to move out of the game area, the player will remain at its current position. The players have perfect information of the geometrical location of the other players. All the players have a constant speed, but not necessary the same speed. At any time instant t, the players have perfect knowledge of the control decisions chosen by their opponents prior to t, but none regarding the future control decisions their opponents will choose.

2.2. **Problem formulation.** In this section, we formulate the three-player game that we make use of to investigate the learning process of the players in the game of guarding a territory. In our formulation, the players have no a priori knowledge of their optimal behaviors. The three-player game consists of two guards against a high speed invader, as shown in Figure 1. Also shown in Figure 1, the game area was chosen as a square area of size 30 × 30 units. We increase the complexity of the guarding a territory problem by setting the invader to be 30 percent faster than the two guards. As such, the invader is superior to the two guards. In this paper, we refer to any player with speed greater than that of the other players as a superior player. The speed of the invader was arbitrarily selected taking consideration of the length of a simulation run and how it will present in the paper. We have done simulations with the invader at different speeds with similar results.

The goal of this problem is to investigate the possibility of the two guards learning a joint cooperative strategy to capture the superior invader before it reaches the territory. We also expect the invader to learn its optimal behavior to avoid capture and get as close as possible to the territory. Kindly note that the dimensions of the robots shown in Figure 1 do not represent the size of the robot nor the type of robot used in our simulation. The robots shown in Figure 1 are used only for illustrative purposes.

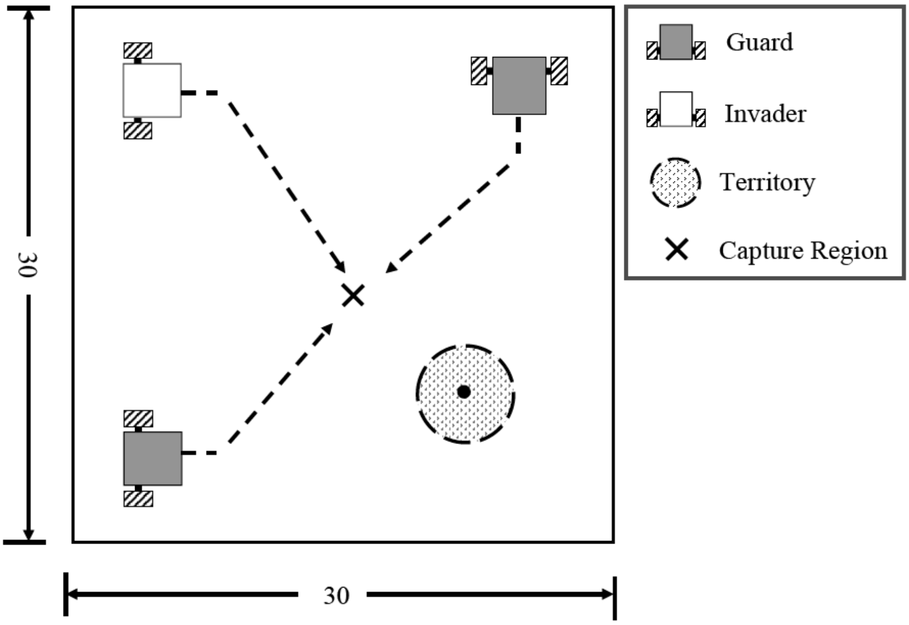

FIGURE 1. Three-player game setup: a superior invader against two inferior guards

3. Reinforcement Learning.

Reinforcement learning is a type of machine learning that can be used to train an agent to optimize some reward function through interaction with the initially unknown environment. Reinforcement learning is different from supervised learning. In supervised learning, the agent learns with the desired input-output pattern provided by an external supervisor [13]. However, in reinforcement learning, there is no external supervisor to guide the learning process of the agent. The agent has no knowledge of the desired output. Therefore, the agent must explore its available actions to discover which actions yield the most reward. The agent must also exploit what it already knows in order to improve its performance in the long run. The overall goal of the agent is to learn a policy (i.e., a mapping from states to actions) that maximizes its long-term cumulative reward [13].

The policy can be in the form of a look-up table, if the state and action spaces are discrete and small in number. However, if the state and action spaces are large or in the continuous domain, it is intractable to maintain a look-up table. In this case, function approximators, such as the FLC, can be used to represent the continuous state and action spaces in an effective way [10, 14]. The FLC has two significant advantages when used as a function-approximation technique: firstly, the FLC permits human heuristic knowledge to be incorporated in the control design; secondly, the learning process can be localized to only the consequent parameters [15, 16].

A class of reinforcement learning methods called actor-critic methods can be combined with the FLC to solve problems in the continuous domain [10, 14]. Actor-critic methods have a separate memory structure to represent the decision mechanism (actor) independent of the value function estimator (critic) [13]. We make use of the FLC to implement both the actor and the critic. When actor-critic learning is combined with the FLC, the approach is sometimes called the FACL method [10, 14].

In this paper, we apply the FACL method in solving the guarding a territory problem. The advantage of the FACL method is that it does not require a priori discretization of the continuous spaces. As such, the FACL has the characteristics of fast action selection and lower memory requirement making it the suitable approach to solve the guarding a territory problem in continuous time. This section describes the FACL method and also provides specifics for the design of the fuzzy controller.

3.1. Fuzzy control system design.

In this section, we present details of the design of the controller for the multi-agent high speed invader guarding a territory problem. The design of the FLC essentially involves:

1) Choosing the controller model;
2) Choosing the method for fuzzification and defuzzification, and the type of fuzzy inference engine;
3) Choosing the inputs to the controller;
4) Designing the fuzzy sets and the fuzzy rules.

In our design, the FLC is implemented using the zero-order Sugeno fuzzy model [17]. We make use of the singleton method for fuzzification, the product inference engine for the fuzzy inference engine and the weighted average method for defuzzification. The choice of the model of the FLC, and the methods for fuzzification and defuzzification are based on a priori knowledge of the problem.

We now define the inputs to the controller. We propose making use of state variables that describe the state of the environment to define the inputs to the controller. Our choice of inputs is similar to those used in [11] for the case of a single guard and a low speed invader. However, in our particular scenario, we defined two additional inputs for the high speed super invader. These additional inputs will enable the superior invader

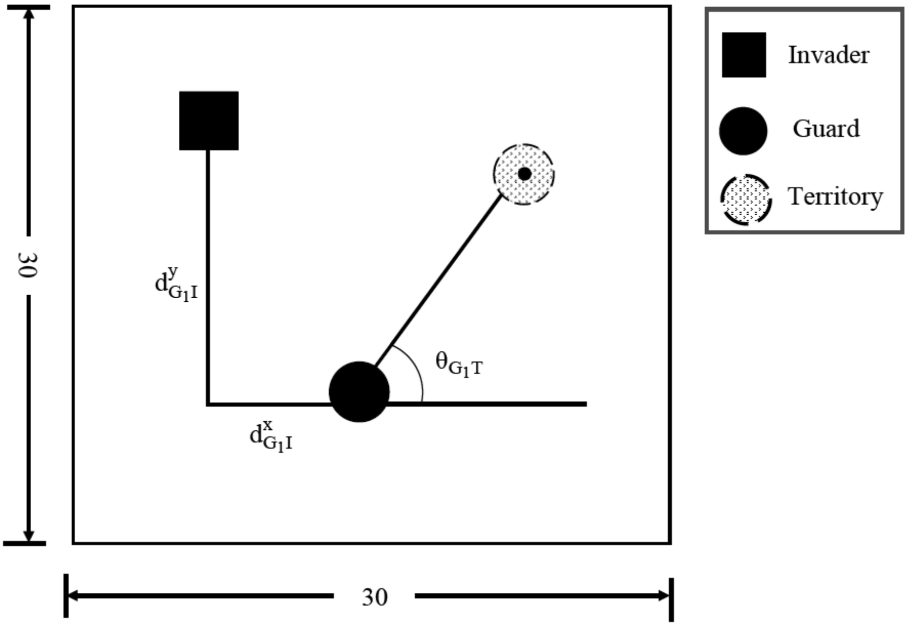

FIGURE 2. The inputs to the controller of guard 1, G_1

to learn with the information of the location of the second guard. Five inputs in total were chosen for the high speed invader. In the following we provide detailed description of these inputs. How one chooses the states or inputs to the actor and the critic is important for learning success.

For the inputs to the controller of any one of the guards, we propose the use of three state variables. The first state variable describes the position of the territory with respect to the guard and the other two state variables describe the position of the invader with respect to the guard. We take for example the inputs to the controller of guard 1, G_1. The first variable θ_{G_1T} is the angle between the global x-axis and guard 1's line of sight towards the territory, as shown in Figure 2. The domain of the variable θ_{G_1T} is defined as $[-\pi, \pi]$. The variable θ_{G_1T} describes the position of the territory with respect to guard 1. The second variable $d^x_{G_1I}$ and the third variable $d^y_{G_1I}$ are the components of the Manhattan distance between the invader and guard 1, as shown in Figure 2. The domain of $d^x_{G_1I}$ and $d^y_{G_1I}$ is defined as $[-30, 30]$. The variables $d^x_{G_1I}$ and $d^y_{G_1I}$ describe the position of the invader with respect to guard 1. Therefore, the vectors of state variables $\bar{x}_{G_1}$ and $\bar{x}_{G_2}$ that define the inputs to the controller of guard 1, G_1, and guard 2, G_2, are given as

$$\bar{x}_{G_1} = \left[\theta_{G_1T},\ d^x_{G_1I},\ d^y_{G_1I}\right] \tag{4}$$

$$\bar{x}_{G_2} = \left[\theta_{G_2T},\ d^x_{G_2I},\ d^y_{G_2I}\right] \tag{5}$$

respectively. Our objective in choosing three state variables to define the inputs to the controller of the guards is to investigate whether the guards are able to learn in a decentralized manner. In other words, it was done to investigate the possibility of both guards jointly working to capture the superior invader with no form of communication between the guards.

For the inputs to the controller of the superior invader, we propose the use of five state variables. The first variable θ_{IT} describes the position of the territory with respect to the invader. The second variable $d^x_{IG_1}$ and the third variable $d^y_{IG_1}$ describe the position of guard 1, with respect to the invader. Similarly, the fourth variable $d^x_{IG_2}$ and the fifth variable $d^y_{IG_2}$ describe the position of guard 2, with respect to the invader. Therefore, the vector of state variables $\bar{x}_I$ that define the inputs to the controller of the invader is given as

$$\bar{x}_I = \left[\theta_{IT},\ d^x_{IG_1},\ d^y_{IG_1},\ d^x_{IG_2},\ d^y_{IG_2}\right] \tag{6}$$

We make use of triangular membership functions to define all the fuzzy sets in the various input domains. Ten symmetrical and uniformly spread triangular membership functions were defined for the angle variable θ. The variable θ describes the inputs: θ_{IT}, θ_{G_1T} and θ_{G_2T}. Similarly, seven triangular membership functions were defined for the Manhattan distance variable d. The variable d describes the inputs: $d^x_{IG_1}$, $d^y_{IG_1}$, $d^x_{IG_2}$, $d^y_{IG_2}$, $d^x_{G_1I}$, $d^y_{G_1I}$, $d^x_{G_2I}$, and $d^y_{G_2I}$.

Since the FLC is implemented using a zero-order Sugeno fuzzy model, any rule l in the fuzzy rule base defined for the controller of the guards will take the form

$$\mathrm{Ru}^{(l)}:\ \text{IF } \theta_{GT} \text{ is } \mathrm{A}^l_1,\ d^x_{GI} \text{ is } \mathrm{A}^l_2,\ \text{and } d^y_{GI} \text{ is } \mathrm{A}^l_3,\ \text{THEN } \phi^l_t(\bar{x}_t) = c^l \tag{7}$$

where A^l_1 in θ_{GT}, A^l_2 in d^x_{GI} and A^l_3 in d^y_{GI} are fuzzy sets in the antecedent, $\phi^l_t(\bar{x}_t)$ is the output of rule l at time step t, and c^l is the constant consequent parameter for rule l.

Similarly, for the controller of the high speed invader any rule l in the fuzzy rule base will take the form

$$\begin{aligned}\mathrm{Ru}^{(l)}:\ \text{IF } \theta_{IT} \text{ is } \mathrm{B}^l_1,\ d^x_{IG_1} \text{ is } \mathrm{B}^l_2,\ d^y_{IG_1} \text{ is } \mathrm{B}^l_3,\ d^x_{IG_2} \text{ is } \mathrm{B}^l_4,\ \text{and } d^y_{IG_2} \text{ is } \mathrm{B}^l_5,\\ \text{THEN } \phi^l_t(\bar{x}_t) = c^l\end{aligned} \tag{8}$$

where B^l_1 in θ_{IT}, B^l_2 in $d^x_{IG_1}$, B^l_3 in $d^y_{IG_1}$, B^l_4 in $d^x_{IG_2}$ and B^l_5 in $d^y_{IG_2}$ are fuzzy sets in the antecedent.

For the invader, we apply a total of $10 \times 7 \times 7 \times 7 \times 7 = 24010$ fuzzy rules. However, for each of the guards, we apply a total of $10 \times 7 \times 7 = 490$ fuzzy rules. We observe that there is an exponential growth in the number of fuzzy rules as the dimension of the input space increases. This is due to the fact that the the product inference engine requires an exhaustive list of all possible permutation of the fuzzy rules. In fuzzy logic, this problem is well-known as the *curse of dimensionality* [18]. To increase the computational efficiency of the system we made use of triangular membership functions to define all the fuzzy sets. As a result, only the rules that are active (or have non-zero firing strength) will contribute to the overall output of the system. The remaining rules will have zero membership value.

The degree of fulfillment (or firing strength) $\psi^l_t(\bar{x}_t)$ of any rule l will take the form

$$\psi^l_t(\bar{x}_t) = T\left[\mu_{A^l_1}(\theta_{GT}),\ \mu_{A^l_2}(d^x_{GI}),\ \mu_{A^l_3}(d^y_{GI})\right] \tag{9}$$

where $T[.]$ implies the T-norm operator and $\mu_{A^l_1}(\theta_{GT})$ provides the membership value of the input variable θ_{GT}. We make use of the singleton method for fuzzification, the product inference engine for the fuzzy inference engine, and the weighted average method for defuzzification. Therefore, the overall crisp output $\phi_t(\bar{x}_t)$ of the controller is given as

$$\phi_t(\bar{x}_t) = \frac{\sum\limits_{l=1}^{L} \psi^l_t c^l}{\sum\limits_{l=1}^{L} \psi^l_t} = \sum_{l=1}^{L} \varphi^l_t c^l \tag{10}$$

where L is the total number of fuzzy rules and φ^l_t is the normalized degree of fulfillment for rule l defined as

$$\varphi^l_t = \frac{\psi^l_t}{\sum\limits_{l=1}^{L} \psi^l_t} \tag{11}$$

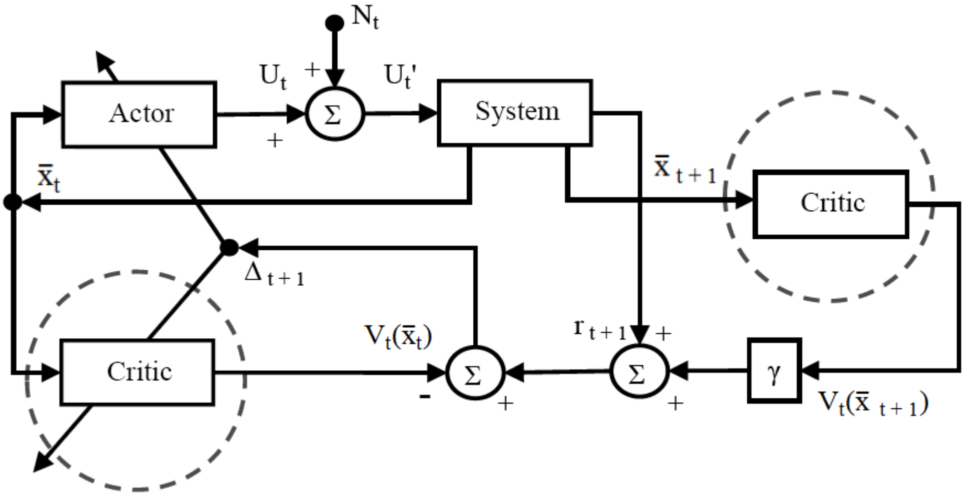

FIGURE 3. FACL scheme for the guarding a territory problem

3.2. **The FACL method.** The FACL scheme for the guarding a territory problem is shown in Figure 3. The dotted circle enclosing the critic blocks in Figure 3, signifies that we are referring to the same critic. We abstracted the robot and its environment to the system block, as shown in Figure 3.

The role of the actor is to generate the continuous control action $U_t(\bar{x}_t)$. In our implementation, $U_t(\bar{x}_t)$ translates to the desired heading angle of the robot. To generate the correct control actions, the actor maintains a vector W which has as many components as the number of fuzzy rules. The components of the vector W are used as the consequent parameters in the defuzzification procedure for the actor in Equation (10). Therefore, the output of the actor is given as

$$U_t(\bar{x}_t) = \sum_{l=1}^{L} \varphi_t^l W_t^l \tag{12}$$

A key issue of reinforcement learning is balancing exploration and exploitation. To cause sufficient exploration of the action space in the FACL algorithm, a Gaussian white noise parameter N_t with mean zero and variance σ_t^2 is added to the continuous control action $U_t(\bar{x}_t)$ to produce the final control action $U_t'(\bar{x}_t)$, as shown in Figure 3. The adjusted control action $U_t'(\bar{x}_t)$ is used to run the game.

The role of the critic is to predict the sum of future discounted rewards (or the state-value function) V_t in reinforcement learning given as

$$V_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{13}$$

where $\gamma \in [0, 1]$ is the discounting factor and r is the external reinforcement signal. Notice Equation (13) can also be written recursively as

$$V_t = r_{t+1} + \gamma V_{t+1} \tag{14}$$

The critic is able to make a good estimate of the state-value function by maintaining a vector ζ which has as many components as the number of fuzzy rules. The components of the vector ζ are used as the consequent parameters in the defuzzification procedure for the critic in Equation (10). Therefore, the output of the critic is given as

$$V_t(\bar{x}_t) = \sum_{l=1}^{L} \varphi_t^l \zeta_t^l \tag{15}$$

The external reinforcement signal r_{t+1} received from the environment after executing action $U_t(\bar{x}_t)$ in state $\bar{x}_t$ and transiting to state $\bar{x}_{t+1}$ is used to compute a temporal difference (TD) error Δ_{t+1} given by

$$\Delta_{t+1} = r_{t+1} + \gamma V_{t+1}(\bar{x}_{t+1}) - V_t(\bar{x}_t) \tag{16}$$

We provide details of the external reinforcement signal r_{t+1} in Section 3.3.

The TD error is used to adapt the actor and the critic. To prevent large adaption steps of the actor in the wrong direction, a more stable way to adapt the actor is to use only the sign of the TD error Δ_{t+1} and the exploration N_t [16]. The adaptation law for the actor is therefore given as

$$W^l_{t+1} = W^l_t + \left\{ \beta \ \mathrm{sign}[N_t \Delta_{t+1}] \frac{\partial U_t(\bar{x}_t)}{\partial W^l_t} \right\} \tag{17}$$

where $\beta \in [0, 1]$ is the learning rate of the actor. The adaptation law for the critic is given as

$$\zeta^l_{t+1} = \zeta^l_t + \left\{ \alpha \Delta_{t+1} \frac{\partial V_t(\bar{x}_t)}{\partial \zeta^l_t} \right\} \tag{18}$$

where $\alpha \in [0, 1]$ is the learning rate of the critic. The partial derivate in Equations (17) and (18) is easily determined from Equations (12) and (15) respectively to be

$$\frac{\partial U_t(\bar{x}_t)}{\partial W^l_t} = \frac{\partial V_t(\bar{x}_t)}{\partial \zeta^l_t} = \varphi^l_t \tag{19}$$

The FACL algorithm is summarized in Algorithm 1 [10]. This FACL algorithm was implemented in [10, 11] for the case of a single guard and a low speed invader. In our high speed super invader and multi-guard problem, the states/inputs, the fuzzy rules and the rewards had to be modified for learning success. In comparison to the method proposed in [10, 11], we increased the number of inputs to the invader from three to five. These additional inputs resulted in an increase in the number of fuzzy rules from 490 to 24010. The two additional inputs will enable the invader to learn with the information of the location of the second guard. For each of the guards, we propose the use of three inputs as described in Section 3.1. Our goal is to investigate if the guards can learn to capture the high speed super invader with no direct communication between the guards. It will be interesting to investigate the possibility of the guards learning a decentralized control strategy to capture the high speed invader. In Section 3.3, we provide specifics for the design of the reward functions.

Algorithm 1 FACL algorithm

```
Initialize $V(.) = 0$, $W^l = 0$, $\zeta^l = 0$ for $l = 1, \dots, L$.
for each time step $t$ do
    Obtain the inputs $\bar{x}_t$.
    Compute the control action $U_t(\bar{x}_t)$ in Equation (12).
    Estimate the state-value function $V_t(\bar{x}_t)$ in Equation (15).
    Use $U_t(\bar{x}_t)$ to run the game for the current time step $t$.
    Obtain the new inputs $\bar{x}_{t+1}$ and the reward $r_{t+1}$ at time $t + 1$.
    Estimate the state-value function $V_t(\bar{x}_{t+1})$ in Equation (15).
    Compute the TD error $\Delta_{t+1}$ in Equation (16).
    Adapt $W^l_{t+1}$ in Equation (17) and $\zeta^l_{t+1}$ in Equation (18) for $l = 1, \dots, L$.
end for
```

3.3. **Reward function.** In the guarding a territory game, one may use the payoff obtained at game termination to reward the performance of the learning agents. However, delaying the reward so that it is only received at game termination will result in a slow learning procedure and will also make the learning process difficult. This is because by game termination the agents would have taken many actions. As such, it would be difficult to know which actions were good and should be rewarded, and which were bad and should be penalized. In reinforcement learning, this problem is well-known as the *temporary credit assignment problem* [19].

To improve the learning process of the agents, one may introduce instantaneous rewards to the environment [20, 21]. Instantaneous rewards provide immediate reinforcement to compensate for the delayed reward. As a result, the agents would learn instantaneously which actions they take are good and which are bad. The reward function must be designed such that it reflects the goal of the agents.

3.3.1. *Reward function for the invader.* The reward function for the invader R^I_{t+1} is designed using a method similar to the one proposed in [6] for the case of a single guard against a low speed invader. In our particular scenario, we modified the reward function for the invader such that the invader learns with the information of the location of the multiple guards. This is essential to ensure that the invader is able to learn its optimal behavior to get as close as possible to the territory before capture occurs. In the following we provide detailed description of the reward function for the invader.

The reward function R^I_{t+1} for the high speed invader has three components. The first component δ_{IT} gives a positive reward if the invader moves closer to the territory T and a negative reward otherwise, given as

$$\delta_{IT} = dist_{IC_T}(t) - dist_{IC_T}(t+1) \tag{20}$$

where $dist_{IC_T}(t)$ denotes the Euclidean distance between the invader I and the center of the territory C_T at time step t. The second component δ_{IG_1} gives a positive reward if the invader moves farther from guard 1 and a negative reward otherwise, given as

$$\delta_{IG_1} = dist_{IG_1}(t+1) - dist_{IG_1}(t) \tag{21}$$

Similarly, the third component δ_{IG_2} gives a positive reward if the invader moves farther from guard 2 and a negative reward otherwise, given as

$$\delta_{IG_2} = dist_{IG_2}(t+1) - dist_{IG_2}(t) \tag{22}$$

Putting the three components together results in the reward function given in Equation (23)

$$R^I_{t+1} = D\delta_{IT} + J_1\delta_{IG_1} + J_2\delta_{IG_2} \tag{23}$$

where D, J_1, J_2 are constants that determine the importance attached to the various components of the reward function.

A trade-off between moving farther from the guards and moving closer to the territory has to be reached. For example, if we choose D as a large number, this will make the invader move towards the territory while ignoring the guard. On the other hand, if we choose J_1, and J_2 as a large number, this will make the invader overly afraid of the guards. As a result, the invader will prefer a longer path to reach the territory. In Section 5, we provide details of how we set the values of these constants D, J_1, and J_2.

3.3.2. *Reward function for the guard.* For the reward function of any one of the guards, we propose the use of two components. The first component gives a positive reward if the guard moves closer to the invader and a negative reward otherwise. While the second component gives a positive reward if the guard moves closer to the territory and

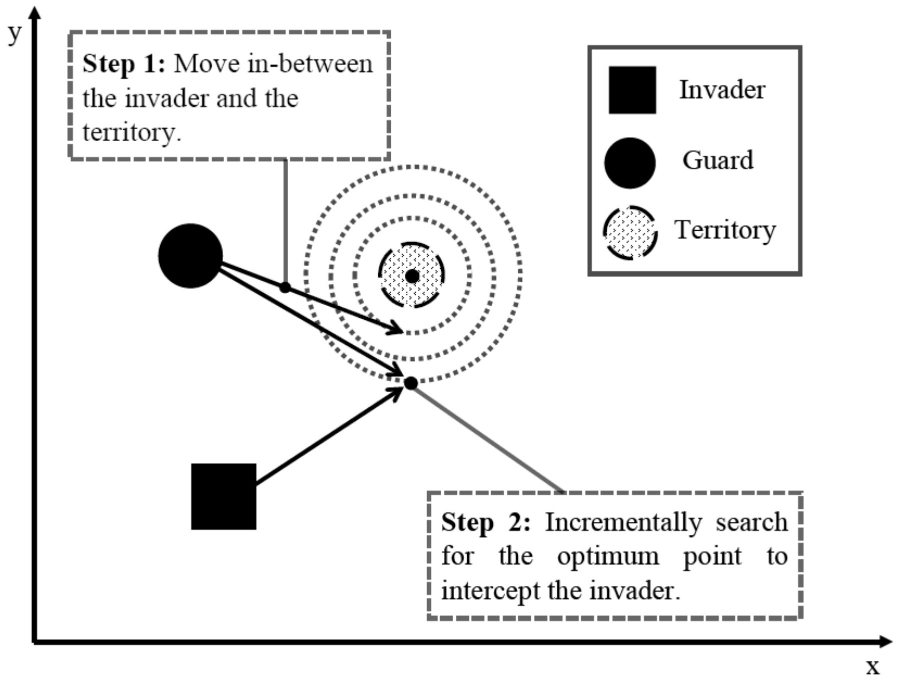
y
x
Step 1: Move in-between the invader and the territory.
Step 2: Incrementally search for the optimum point to intercept the invader.
Invader
Guard
Territory

FIGURE 4. The guard's reward function description

a negative reward otherwise. We take for example the reward function $R_{t+1}^{G_1}$ for guard 1. The first component δ_{G_1I} gives a positive reward if guard 1 moves closer to the invader and a negative reward otherwise, given as

$$\delta_{G_1I} = dist_{G_1I}(t) - dist_{G_1I}(t+1) \tag{24}$$

The second component δ_{G_1T} gives a positive reward if guard 1 moves closer to the territory and a negative reward otherwise, given as

$$\delta_{G_1T} = dist_{G_1T}(t) - dist_{G_1T}(t+1) \tag{25}$$

Therefore, the reward functions $R_{t+1}^{G_1}$ and $R_{t+1}^{G_2}$ for guard 1 and guard 2 are given as

$$R_{t+1}^{G_1} = P\delta_{G_1I} + M\delta_{G_1T} \tag{26}$$

$$R_{t+1}^{G_2} = P\delta_{G_2I} + M\delta_{G_2T} \tag{27}$$

respectively, where P and M are constants that determine the importance attached to the various components of the reward functions.

The reward function for the guard is illustrated in Figure 4. A trade-off between moving closer to the territory and moving closer to the invader must be reached. For example, if we choose P as a large number, the guard will directly pursue after the invader, which is an irrational strategy well studied in [6]. On the other hand, if we choose M as a large number, this will make the guard ignore the invader and move towards the territory. In Section 5, we provide details of how we set the values of the constants P and M.

4. **The Optimal Strategies of the Players.** When the players have the same speed, the optimal strategies of the players can be determined using the perpendicular bisector approach [1]. However, when the players do not have the same speed, it would be difficult, if not impossible, to use the same approach to determine the optimal strategies of the players. In this section, we briefly introduce the Apollonius circle and subsequently present the Apollonius circle approach to determine the optimal strategies of the players when

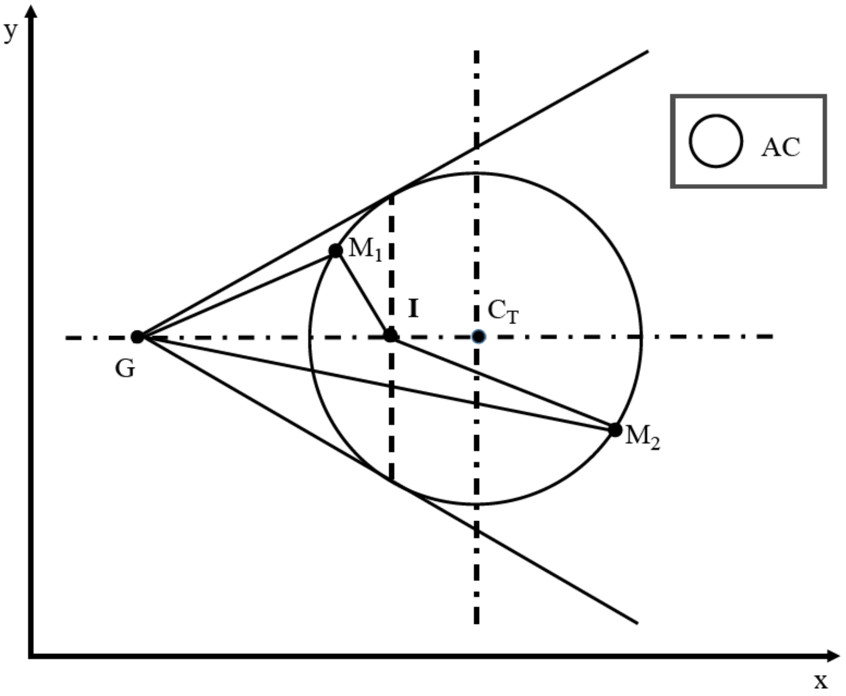

Figure 5. The Apollonius circle AC constructed for an inferior invader I against a superior guard G

the invader is superior to two guards. We make use of the Apollonius circle solution only for the purpose of evaluating the players' learning outcome.

4.1. **The Apollonius circle.** Consider the Apollonius circle AC formed by the invader I and the guard G as shown in Figure 5. In this scenario, the guard is superior to the invader (i.e., $V_G > V_I$). The Apollonius circle will always encircle the inferior player. If we pick any point M_1, M_2, on the Apollonius circle as shown in Figure 5, both players can reach that point at the same time, given their individual speeds. Let λ be the ratio of the speeds of the players, such that

$$\lambda = \frac{V_G}{V_I} = \frac{|GM_1|}{|IM_1|} = \frac{|GM_2|}{|IM_2|} \tag{28}$$

where $|GM_1|$ denotes the Euclidean distance between the position of the guard G and the point M_1. The center of the Apollonius circle C_{AC} and its radius R_{AC} can be computed using

$$C_{AC} = \left(\frac{x_G - \lambda^2 x_I}{1-\lambda^2}, \frac{y_G - \lambda^2 y_I}{1-\lambda^2} \right) \tag{29}$$

$$R_{AC} = \frac{\lambda\sqrt{(x_G - x_I)^2 + (y_G - y_I)^2}}{1-\lambda^2} \tag{30}$$

respectively [22, 23].

4.2. **The optimal strategy of a superior invader against two inferior guards.** Consider the scenario presented in Figure 6. In this scenario, there are two guards in the game against a high speed super invader. The invader is located at position I, the first guard (i.e., guard 1) is located at position G_1 and the second guard (i.e., guard 2) is located at position G_2. The Apollonius circle formed by the invader and each guard is also shown in Figure 6. The Apollonius circle will encircle the guards, since the guards

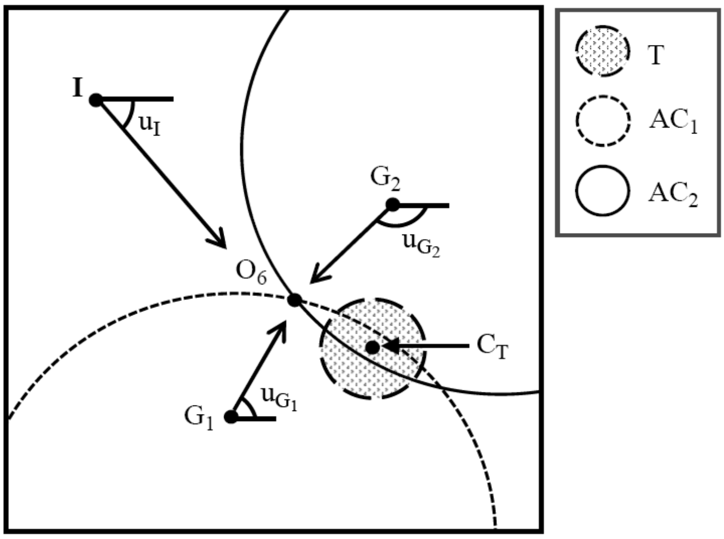

FIGURE 6. The optimal strategy of a superior invader I against two inferior guards G_1 and G_2

are the inferior players. As shown in Figure 6, AC_1 denotes the Apollonius circle formed between the invader and guard 1, while AC_2 denotes the Apollonius circle formed between the invader and guard 2.

In order to simplify the problem, we assume that the environment is bounded such that it is impossible for the invader to go around any of the Apollonius circles. Therefore, for this scenario, the optimal strategy of all three players is to move to the point O_6. The point O_6 is the closest the invader can get to the territory as well as the farthest from the territory the guards can intercept the invader. The point O_6 must be recomputed at each time instant.

The invader, guard 1 and guard 2 must find their respective strategies $\bar{u}_I = \bar{u}_I^*$, $\bar{u}_{G_1} = \bar{u}_{G_1}^*$ and $\bar{u}_{G_2} = \bar{u}_{G_2}^*$, in order to optimize their payoffs. Should any one of the guards fail to find its optimal strategy, this may lead to adjacent Apollonius circles no longer intersecting each other (i.e., an opening between the adjacent Apollonius circles). A rational invader should be able to exploit this opening between adjacent Apollonius circles to achieve a more favorable payoff. Likewise, should the invader fail to find its optimal strategy, the guards should be able to achieve a more favorable payoff assuming they play rationally.

5. **Simulation Results.** In this section, we apply the reinforcement learning method described in Section 3 to the guarding a territory game. We make use of the problem formulated in Section 2.2 to simulate and evaluate the performance of the players. In our simulations, all the players in the game (the invader and the guards) are simultaneously learning. We set the learning rates of the actor and the critic in the FACL algorithm as follows: $\beta_0 = 0.05$ in Equation (17), $\alpha_0 = 0.1$ in Equation (18). The variance of the exploration noise σ and the discount factor γ are set as follows: $\sigma_0^2 = 1$ and $\gamma = 0.5$. The decay rates of the learning factors after each epoch k are set as follows: $\beta_{k+1} = 0.994^k\beta_0$, $\alpha_{k+1} = 0.999^k\alpha_0$, and $\sigma_{k+1}^2 = 0.994^k\sigma_0^2$. The values for the learning rates, the exploration noise and the discount factor were chosen based on a priori knowledge of the problem.

The kinematic equations of the players given in Equations (1)-(3) were solved in simulation using the second-order Runge-Kutta method [12]. The sampling time is set to 10 milliseconds. The players have a constant linear speed throughout the game. However, the angular speed ω_t of any one of the players is given as

$$\omega_t = \mathrm{K}(U_t - \theta_t) \tag{31}$$

where K is the proportional gain, U_t is the desired heading angle obtained from the controller and θ_t is the current heading angle of the player relative to the global x-axis. We set $\mathrm{K} = 1$, whereas U_t is determined using Equation (12).

The total simulation time for an epoch is set to 60 seconds. An epoch describes a single run of the game, which starts when the players are in their initial positions and ends when the game meets a terminating condition. The game terminates if the simulation timer for an epoch expires, and the invader reaches the territory or the invader is within the capture radius R_C of the guards. We set $R_C = 2$ units for all the guards.

Each epoch is carried out in two phases: a training phase where the learning occurs and a test phase where the players apply what they have learned. In the training phase, we set the players' initial positions at random in specified training regions, whereas in the test phase, we set the players' initial positions at fixed locations. We record only the payoff obtained in the test phase. We average the payoff over 10 learning trials. In this scenario, a single learning trial describes one complete cycle of the game which contains 500 training epochs and 500 test epochs. The procedure is summarized in Algorithm 2 [6]. In Algorithm 2, k and p denote the current index of the epoch and the learning trial respectively. The objective of the Test-Train algorithm is to investigate whether the training performed in one area can be applied to other regions.

Algorithm 2 Train-Test algorithm

```
Initialize $p = 1$, $k = 1$.
while $p \leq 10$ do
    Initialize $W^l = 0$, $\zeta^l = 0$ for $l = 1, \ldots, L$.
    while $k \leq 500$ do
        Run training epoch with the players' initial positions assigned at random in
          training area.
        Adapt $W^l_{t+1}$ in Equation (17) and $\zeta^l_{t+1}$ in Equation (18) for $l = 1, \ldots, L$.
        Run test epoch with the players' initial positions fixed at assigned test
          locations.
        Record the payoff achieved.
        $k = k + 1$.
    end while
    Record the average payoff achieved.
    $p = p + 1$.
end while
```

5.1. **Simulation of the multi-agent superior invader guarding a territory problem.** The game setup is shown in Figure 7. In this scenario, there are three players in the game: two guards and a high speed invader. We set the speed of the guards to $V_{G_1} = V_{G_2} = 1$ unit/second, while the speed of the invader is set to $V_I = 1.3$ units/second. Therefore, the invader is 30 percent faster than the guards.

The training regions for the players are illustrated in Figure 7. We set the test location for guard 1 at $[5, 5]$ and its initial orientation to 0 radians. The test location for guard 2 is set at $[25, 25]$ and its initial orientation to $-\pi/2$ radians. Similarly, we set the test location for the invader at $[5, 25]$ and its initial orientation to 0 radians. The center of the territory C_T is located at $[22, 8]$ and the radius $R_T = 3$ units. The Apollonius circles constructed

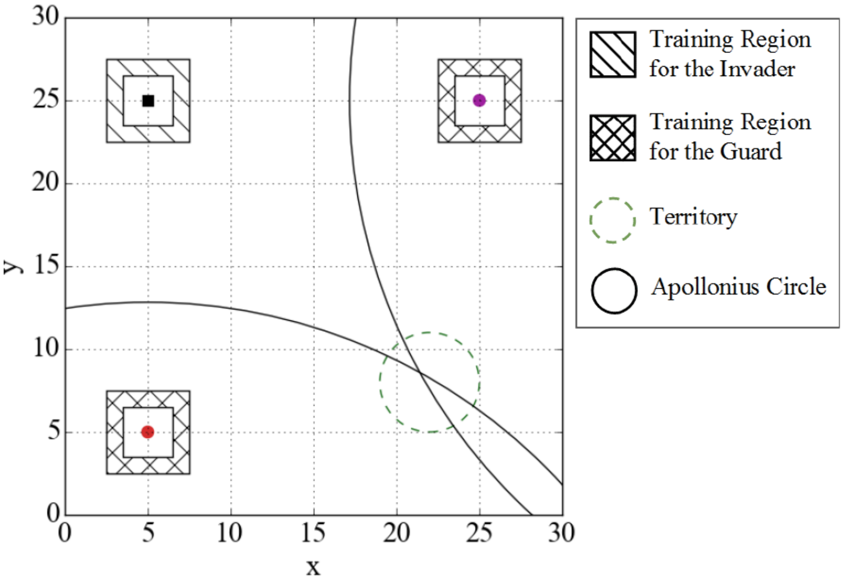
30
25
20
15
10
5
0
y
0
5
10
15
20
25
30
x
Training Region for the Invader
Training Region for the Guard
Territory
Apollonius Circle

Figure 7. The game setup for a superior invader against two inferior guards

in Figure 7 describe the regions the guards can reach before the invader. Although the Apollonius circles do not completely encircle the territory, the guards should be able to intercept the invader before the invader reaches the territory. This is possible if the guards exploit their 2 units capture radius.

For the reward function of the invader given in Equation (23), we set the parameters $D = 3.0$, $J_1 = 1.0$ and $J_2 = 1.0$. The parameter D describes the importance the invader attaches to moving towards the territory, while J_1 and J_2 describe the importance the invader attaches to moving farther away from guard 1 and guard 2 respectively. The value of the parameter D was chosen to be three times the value of the parameters J_1 and J_2 so that the invader places greater importance on moving towards the territory.

Similarly, for the reward function of guard 1 and guard 2 given in Equation (26) and Equation (27) respectively, we set the parameters $P = 1.1$ and $M = 1.0$. The parameter P describes the importance the guard attaches to moving towards the invader, while M describes the importance the guard attaches to moving towards the territory. Both components of the reward functions described in Equation (26) and Equation (27) will work jointly to guide the guards to determine the optimal point in-between the invader and the territory that maximizes the capture distance of the invader from the territory, as illustrated in Figure 4.

We run the simulation for 500 epochs with guard 2 located at $[25, 25]$ absent. This is done to investigate whether a single guard can defend the territory in this scenario. The trajectories of the learning agents at the final epoch are shown in Figure 8. We observe in Figure 8 that guard 1 alone cannot stop the superior invader from reaching the territory. As shown in Figure 8, the guard made an attempt to move as close as possible to the invader to effect capture. However, the invader learned its optimal strategy to aim for the region of the territory that it can reach before the guard. In this paper, we make use of the payoff obtained at game termination to evaluate the performance of the learning agents. We run the present simulation for 10 learning trials using the Test-Train algorithm. The payoff averaged over 10 learning trials is shown in Figure 9. We observe in Figure 9 that, on average, the learning process is complete after 100 epochs. From the 100th epoch onwards, the invader is able to reach the territory regardless of what the single guard did.

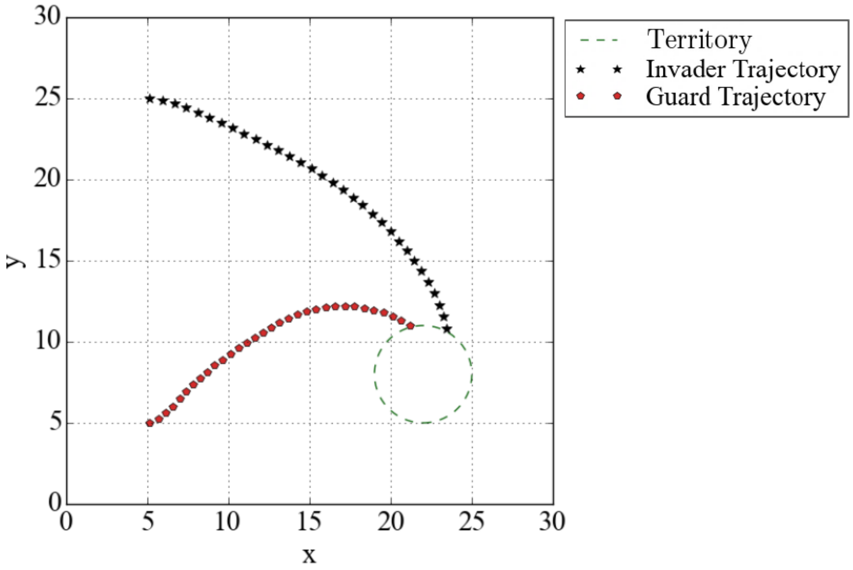

Figure 8. FACL simulation: the trajectory plot of the final epoch for a superior invader against two inferior guards with guard 2 located at coordinate $[25, 25]$ absent

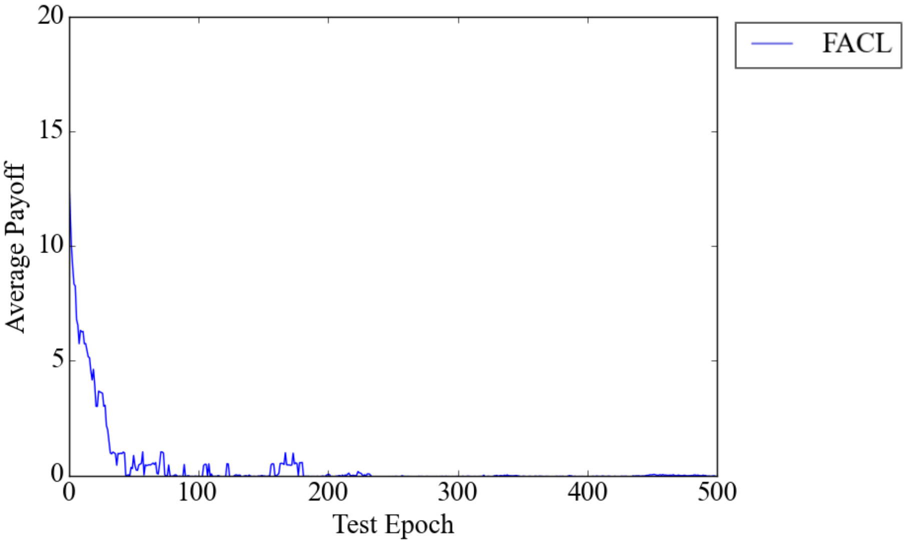

Figure 9. The average payoff plot for a superior invader against two inferior guards with guard 2 located at coordinate $[25, 25]$ absent

We introduce guard 2 into the game and run the simulation for 500 epochs. The trajectories of the learning agents at the final epoch are shown in Figure 10. We observe at the final epoch shown in Figure 10 that the two guards are able to work jointly to capture the superior invader. The guards are able to implicitly cooperate to capture the invader although there is no direct communication between the guards. This is possible because the flow of information of the presence of a second guard in the game is transmitted to the other guard through the motion of the invader. In other words, guard 1 has implicit information about the presence of guard 2 in the game through the motion of the invader.

The payoff averaged over 10 learning trials is shown in Figure 11. The average payoff converges to 5.34 units. In order to compare the player's performance to the optimal strategy given by the Apollonius circle approach, we run one test epoch where all three

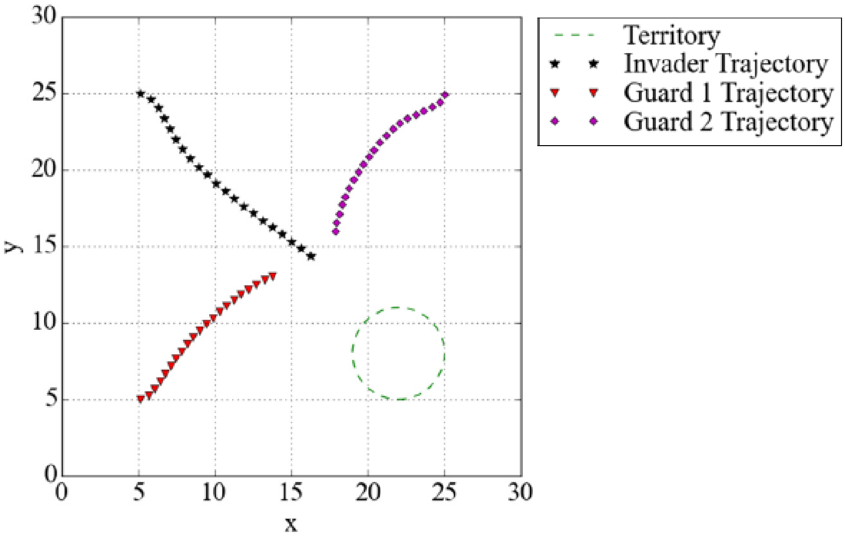

FIGURE 10. FACL simulation: the trajectory plot of the final epoch for a superior invader against two inferior guards with guard 2 located at coordinate [25, 25] present

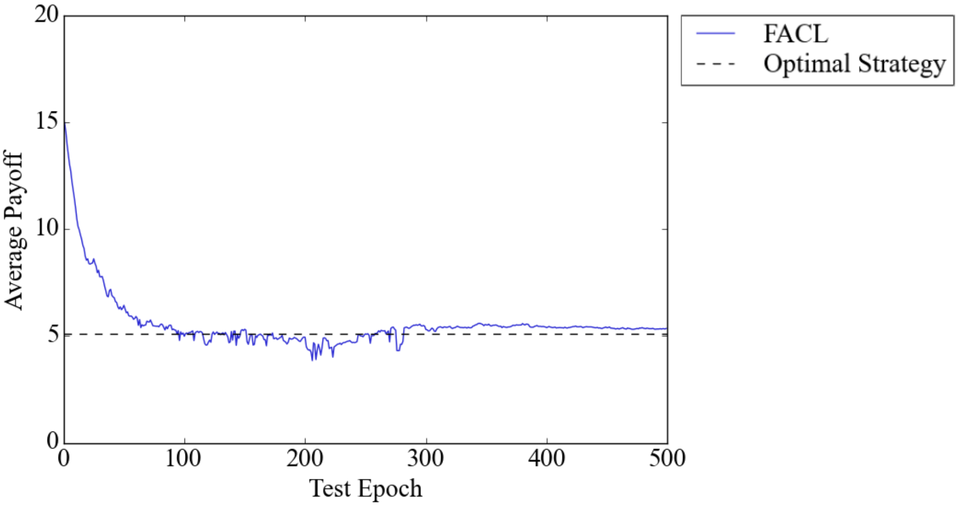

FIGURE 11. The average payoff plot for a superior invader against two inferior guards with guard 2 located at coordinate [25, 25] present

players apply their optimal strategy (with no learning). The optimal strategy gives the payoff value of 5.13 units, as shown in Figure 11. We observe in Figure 11 that there is only a minor difference, approximately 0.21 units, between the average payoff obtained using the learning algorithm and the payoff given by the optimal strategy. This demonstrates that all three players are able to learn their optimal strategies using the FACL algorithm.

5.2. **Discussion.** When designing the controller inputs for the three-player game, we initially thought that the guards would require explicit information about the position of the other guard. However, as the simulation results demonstrate, the information about the existence of the second guard is implicitly transmitted through the motion of the invader. As such, the information about the location of the second guard is not required and that drastically simplified the computational complexity of the system.

The simulation results presented in this section also demonstrated that the learning algorithm was able to determine the optimal strategies of all three players in an effective way. We observe in Section 5.1 that a single guard was not able to stop the superior invader from reaching the territory. However, when we introduced an additional guard to the game, both guards were able to work jointly to intercept the invader before it can reach the territory. The guards were able to implicitly cooperate to capture the superior invader although there is no direct communication between the guards. This demonstrates that the players were able to learn in a decentralized manner.

The simulation results further demonstrated that the training in one area can be applied to other regions. As shown in Figure 7, the test locations for the players are fixed locations within the specified area. Specifically, the test locations for the invader and the guards are 1.5 units distance away from their respective training regions. As demonstrated by the simulation results in Section 5.1, the agents were able to perform optimally in the test locations although they had received no previous training in these locations.

6. Conclusions and Future Work.

This paper investigated the use of reinforcement learning to train the players (the invader and the guards) in the game of guarding a territory. We made use of multi-robot systems as our players in the game. We demonstrated that multi-robot systems with conflicting interests can learn their optimal strategies against their opponents using reinforcement learning technique. We designed a control system to guide the players in the game. Our control system was designed using the FLC. With the aid of reinforcement learning we tuned the control system to improve the performance of the players. The combination of the FLC with reinforcement learning resulted in the FACL method.

In many practical applications of the guarding a territory game in the real-world, the invader (intruder) may have speed greater than the speeds of the guards. Therefore, we considered the possibility of a scenario where the invader is faster than two guards. Simulation results from this study demonstrated that the two guards were able to learn their optimal strategies to defend the territory in an effective way although there was no direct communication between the two guards. Our designed control system was able to guide all three players in this scenario to achieve their optimal performance.

In conclusion, the simulation results demonstrate that our designed control system with reinforcement learning is able to effectively guide multi-robot systems to determine their optimal strategies against their opponent in the game of guarding a territory. Future work will extend this study to investigate the more general scenario of multiple invaders against multiple guards. It would be interesting to investigate how behaviors emerge in such a scenario, such as the different possible collaborations of a team of guards defending the territory against the invaders. This study will be highly beneficial especially as the future looks towards autonomous multi-robot systems for security and surveillance.

REFERENCES

[1] R. Isaacs, *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*, John Wiley and Sons, 1965.

[2] K. H. Hsia and J. G. Hsieh, A first approach to fuzzy differential game problem: Guarding a territory, *Fuzzy Sets and Systems*, vol.55, no.2, pp.157-167, 1993.

[3] Y. S. Lee, K. H. Hsia and J. G. Hsieh, A strategy for a payoff-switching differential game based on fuzzy reasoning, *Fuzzy Sets and Systems*, vol.130, no.2, pp.237-251, 2002.

[4] H. Wang, Q. Yue and J. Liu, Research on pursuit-evasion games with multiple heterogeneous pursuers and a high speed evader, *The 27th Chinese Control and Decision Conference (CCDC)*, pp.4366-4370, 2015.

[5] N. Ono and K. Fukumoto, Multi-agent reinforcement learning: A modular approach, *Proc. of the Second International Conference on Multi-Agent Systems*, pp.252-258, 1996.

[6] H. Raslan, H. Schwartz and S. Givigi, A learning invader for the guarding a territory game, *Proc. of the 2016 Annual IEEE Systems Conference*, pp.1-8, 2016.

[7] C. J. Watkins, *Learning from Delayed Rewards*, Ph.D. Thesis, University of Cambridge, England, 1989.

[8] S. S. Keerthi and B. Ravindran, A tutorial survey of reinforcement learning, *Sadhana*, vol.19, no.6, pp.851-889, 1994.

[9] X. Xu, L. Zuo and Z. Huang, Reinforcement learning algorithms with function approximation: Recent advances and applications, *Information Sciences*, vol.261, pp.1-31, 2014.

[10] H. M. Schwartz, *Multi-Agent Machine Learning: A Reinforcement Approach*, John Wiley and Sons, 2014.

[11] C. V. Analikwu and H. M. Schwartz, Reinforcement learning in the guarding a territory game, *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp.1007-1014, 2016.

[12] B. Siciliano, L. Sciavicco, L. Villani and G. Oriolo, *Robotics: Modelling, Planning and Control*, Springer, 2009.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Massachusetts, 1998.

[14] L. Jouffe, Fuzzy inference system learning by reinforcement methods, *IEEE Trans. Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol.28, no.3, pp.338-355, 1998.

[15] L. Jouffe, Actor-critic learning based on fuzzy inference system, *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, vol.1, pp.339-344, 1996.

[16] W. M. Buijtenen, G. Schram, R. Babuska and H. B. Verbruggen, Adaptive fuzzy control of satellite attitude by reinforcement learning, *IEEE Trans. Fuzzy Systems*, vol.6, no.2, pp.185-194, 1998.

[17] T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, *IEEE Trans. Systems, Man and Cybernetics*, vol.15, no.1, pp.116-132, 1985.

[18] L.-X. Wang, *A Course in Fuzzy Systems and Control*, Prentice Hall, Upper Saddle River, New Jersey, 1997.

[19] M. van Otterlo and M. Wiering, Reinforcement learning and Markov decision processes, in *Reinforcement Learning: State-of-the-Art*, M. Wiering and M. van Otterlo (eds.), 2012.

[20] V. Gullapalli and A. G. Barto, Shaping as a method for accelerating reinforcement learning, *Proc. of the 1992 IEEE International Symposium on Intelligent Control*, pp.554-559, 1992.

[21] J. Randløv and P. Alstrøm, Learning to drive a bicycle using reinforcement learning and shaping, *Proc. of the 15th International Conference on Machine Learning*, pp.463-471, 1998.

[22] B. F. Fang, Q. S. Pan, B. R. Hong and L. Ding, Research on high speed evader vs. multi lower speed pursuers in multi pursuit-evasion games, *Information Technology Journal*, vol.11, no.8, pp.989-997, 2012.

[23] S. Jin and Z. Qu, Pursuit-evasion games with multi-pursuer vs. one fast evader, *Proc. of the 8th World Congress on Intelligent Control and Automation (WCICA)*, pp.3184-3189, 2010.