Decentralized Multi Agent Protect and Decoy Strategies for Capture the Flag

Akshay Krishnan, Ankitha Manoj, Nithin Shrivatsav Srikanth and Prateek Vishal

Abstract— **This project focuses on deploying a team of mobile robots to play the capture the flag game. We have demonstrated the protect and decoy tasks using decentralized networked control protocols. The attackers try to decoy their opponents and try to fetch the flag while the opposing team tries to protect the flag. The team protecting the flag makes use of local sensing information of the position of the attacker and tries to defend the flag by tagging or neutralizing their opponents. Locally designed protocols such as cyclic pursuit, formation control, connectivity preserving consensus, connectivity maintenance and alignment control protocols effectively demonstrate robot behaviours to achieve the desired aspects of the game.**

I. Introduction

With the emergence of swarm robotics, coordination of a large number of robots with control strategies has shown great characteristics like autonomy, modularity, collaboration and local interaction. Thus, the multi-agent paradigm becomes a natural choice to address this issue. The use of local and distributed protocols in robotics helps build scalable teams of cooperative robots. A distributed multi-robot coordination should be:

- **Local** in the sense that individual robots can only act on information that is available to it through sensing communications.
- **Scalable** control algorithm is independent of the size of the team.
- **Safe** and stable multi-agent systems.
- **Emergent** global properties that emerge from the local interaction rules.

This project in particular addresses the issue of using distributed robot teams for demonstrating scenarios that are similar to that in the game of capture the flag. The game consists of two teams, where each team tries to protect their own flag and steal the opponent's flag. In order to simulate the game, the tasks could be delineated as given below.

- **Protection** involves defending one's flag against possible decoy or attack strategies employed by opponents.
- **Exploration** is carried out by agents trying to agents map the environment and collect information on opponents movements
- **Decoy** behaviour is an important strategy to deceive the opponent agents thereby creating favourable conditions to capture the flag.
- **Fetch** is followed by a successful capture of the flag where agents try to bring the flag back to their camp without getting attacked by the opponents.
- **Communication Network** is an important aspect of the game which deals with the various graph topologies that could be employed by the agents to establish secure communication with other agents.

Practical applications of robots with such capabilities can be very diverse. One application of particular importance is in the military. The military forces of the future will use multi-agent robotic work-forces for a variety of applications like security, defence and tactical considerations like decoy, deception and precision strike. There is a definite need to deploy decentralized and reliable coordination among multi-agent robots to accomplish tasks more quickly and effectively than a single agent working independently on a task. Capture the Flag is a perfect game scenario to simulate a military base situation where some agents could be used to protect a resource, and others to attack resources of opponents. In such real world applications, centralized control of autonomous systems comes with problems such as resource sharing between agents in the network and very high dependency on the central node. Multi-agent systems can be effectively modelled by locally designed control protocols that would solve the problems stated earlier. We can use ideas from algebraic graph theory and control theory, these systems could be realized. [1]

The focus on local protocols can also allow us to limit sensing capabilities of robots by using local range sensors such as LiDAR or RADAR. The resulting graph from these sensors results in a delta disk. Such autonomous systems when employed for practical applications can have local range sensors and act on local information obtained from neighbours to accomplish complicated tasks.

Section II of this report begins by laying out the problem statement related to the theme of capture the flag and the underlying assumptions. Section III illustrates our proposed solution and also discusses the node level dynamics of the agents. Section IV deals with the simulation results obtained from running the experiment in the Robotarium.[4]

II. Problem Statement

The protect and decoy tasks are being approached in this report. The arena is simulated in Figure 1 where the red team is the attacking team and the blue team is the defending team. A key problem to be solved to achieve this is to develop a decentralized strategy for the robots to protect their own flag. This could involve a them arranging themselves in formations around the flag. However, simply implementing a formation control behaviour around the flag will not solve the problem. It is necessary to take into consideration the ability of the opponents to decoy the protectors away from the flag. This calls for dynamically switched behaviours that depend on the position of opponents near the flag.

While it is necessary for the agents to protect their flag, it is also equally important for them to procure their opponents flag in order to win the game. After some robots in the team have explored the arena and identified the location of the flag, the next task is to procure it by getting through the opponents. This necessitates a decoy strategy where some agents can distract the opponents who are protecting their flag, while the others go on to procure it from them.

This project aims to develop intelligent distributed, local multi-agent protocols for robot systems to protect a resource and to decoy other enemy robots with similar capabilities, in keeping with the assumptions on the abilities and global information available to the robots that are mentioned below:

- **Sensing abilities:** The robots use sensors that produce delta disk graphs. Practically, this can be achieved by fusing information from multiple LiDAR sensors around the robot to generate 360 degrees of perception. The sensors have finite range which decides the value of Δ for these graphs. In using delta disk graphs, we assume that every agent can sense both the distance to every other agent in within its disk and the relative orientation of this neighbour with respect to itself. We also assume that each agent can detect whether a neighbour belongs to its own team or not.

  An interesting observation that became evident in this context during the project is that it makes it all the more difficult for attackers and protectors to carry out their tasks if they did not have sensing capabilities in all directions. Lack of 360-degree perception (as in the case of a wedge graph) could result in the attackers attacking the capturing the flag by avoiding the field of view of the protectors. At the same time, this could also make it more difficult for the attackers since they are oblivious to protectors that are not in their field of view.
- **Communication abilities:** We assume that the robots can communicate with any agent that is in its field of view. That is, the robots can send information to another robot if the distance between the two is equal to ¡delta¿ which, corresponds to the limits of the delta disk graph. In practice, it is possible that the Δ corresponding to the sensing capabilities are different from that of the communication capabilities.
- **Neutralization:** We make use of the idea of tagging opponents when they are in your half to ensure a fair fight between teams. So, the defenders can neutralize the attackers by tagging them. The attackers, once tagged, cannot move and are frozen.
- **Global coordinate frame:** Each robot has access to its pose in a shared global frame. In practice, they could be using sensors like GPS/IMU to get this information. This is essential for behaviors like going to a goal. We also accept the fact that this information could be prone to errors. This is the reason why not every robot in our approach makes use of this information.
- **Global information among the protectors:** The protectors have access to the location of the flag in their shared coordinate frame. They can sense the flag at all times.
- **Global information among attackers:** A leader among the attacking agents has access to the position of the flag of the opposing team.
- The attackers dont know what the protectors are doing and the protectors dont know what the attackers are doing.
- The switching of a team of robots from one behavior to another is handled in a global sense. That is, information is shared among all robots in a team to meet switching conditions.
- The environment does not have any obstacles and we assume the environment to have zero slip so that the agents move properly without any error in odometry measurements.

III. Proposed Solution

A. Protect Strategy

The strategy first involves the initialization of the agents around the circumference of a circle. Given the radius of a circle the segment of a circle is the inter-agent distance. The formula to calculate the inter-agent distance is as follows:

$$InterAgentDistance = 2(radius)sin(\pi/N_P) \quad (1)$$

In the above equation, N_P is the number of protectors. We assume the agents to know the location of the flag as the agents have sensing capabilities. Just having the agents around the flag to protect it is a very big disadvantage as a attacker with obstacle-avoidance behavior can avoid any of the protector and capture the flag. To counter this problem some other local and scalable network control protocol needs to be designed. When the agents start moving around a circle they form a barrier around the flag making it difficult for the attacking agent to capture the flag. This can be achieved by the cyclic pursuit of protectors around the flag. The cyclic pursuit has been inspired by the ”bugs” problem in mathematics where the framework involves the agent i to follow the motion of agent $i+1$ modulo n. Here we assume

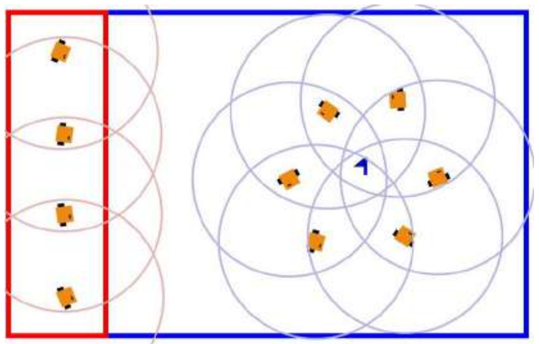


Fig. 1. Structure of arena and initial conditions being assumed. The team that is attacking had red delta disks and those that are protecting their flags have blue. Flag is initially at the centroid of the protectors.

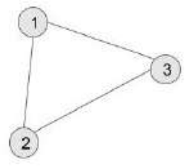


Fig. 2. Undirected Connected Graph for Attackers

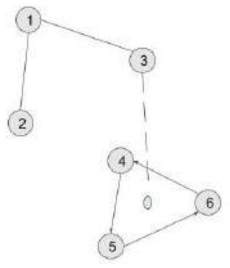


Fig. 3. Directed Cyclic Graph and Line Graph for Protectors

each robot to have the same constant speed when moving in the cyclic pursuit. The protocol is local and distributed in the sense that the an agent has information of its neighbors and no agent knows the state or control of all other agents. The Figure 4 depicts a directed graph between agents undergoing cyclic pursuit. Each agent knows the relative configuration of only its neighbors and it is scalable as we can increase the radius to accommodate more agents without changing the control protocol. Also, since we are writing the dynamics of cyclic pursuit in single-integrator systems and the actual robots are modeled using unicycle dynamics we need to carry out orientation control initially. Unicycle model of robot:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\phi}\end{bmatrix} = \begin{bmatrix}v_o cos(\phi)\\ v_o sin(\phi)\\ \omega\end{bmatrix} \tag{2}$$

Since, angles cause a lot of problems we need to make sure that our angle error is maintained within $[-\pi, \pi]$ Equations

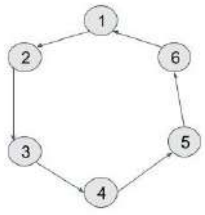


Fig. 4. Directed Cyclic Graph for Protectors

for heading control:

$$e = \phi_d - \phi \tag{3}$$

$$e' = arctan(\frac{sin(e)}{cos(e)}) \tag{4}$$

$$\omega = K_{P1} e' \tag{5}$$

The heading controller is run only once at initialization. After that it is switched off. The dynamical equation describing cyclic pursuit is as follows:

$$\dot{x_i} = R(\theta)(x_{i+1} - x_i) \tag{6}$$

$$\dot{x_N} = R(\theta)(x_1 - x_N) \tag{7}$$

Rotation Matrix $R(\theta)$ is as formulated as follows(the rotation takes place in counter-clockwise direction):

$$R(\theta) = \begin{pmatrix} cos(\theta) & sin(\theta) \\ -sin(\theta) & cos(\theta) \end{pmatrix} \tag{8}$$

When implementing the equations in MATLAB, we need to implement these as discrete-time systems. Hence, we need to use Euler Integration to numerically solve the differential equation. Here we face a problem if the size of each time-step is too small then the system becomes slow or if we increase the size of each time-step then the error accumulates and the agents spiral out of cyclic pursuit and in the process lose connectivity. There is a trade-off between the size of time step and speed of system. To counter this problem we implemented two behaviors on the robot: Bang-Bang Controller on the heading and Separation from a virtual agent at the centre of circle. The final dynamics of the system switches between $\theta + \pi/18$ and $\theta - \pi/18$ and is as follows:

$$\begin{gathered}\dot{x_i} = R(\theta)(x_{i+1} - x_i)\\ +(\|x_i - center\| - radius)(x_i - center)\end{gathered} \tag{9}$$

The above strategy can successfully prevent most of the attackers from capturing the flag. But, if the attacker has increased speed than the protectors they may capture the flag by obstacle avoidance. Hence, the protectors should be provided with a defensive strategy where they try to neutralize the attackers by tagging. This is achieved through the use of a hybrid system where some of the protectors try to neutralize the attackers and the others carry out cyclic pursuit in a reduced radius. The Figure 3 shows the graph depicting connectivity in this case. Agents 4, 5 and 6 have a directed cyclic graph while 1, 2, 3 have an undirected line graph where agents 2, 3 maintain connectivity with agent 1 using a connectivity maintenance consensus protocol. [**?**] Agent 1 tries to do a consensus with the attacker so that it can tag the attacker. Equations 10-13 can be used to implement this behaviour.

$$\dot{x_l} = (x_{decoyer} - x_l) + 0.5W_{12}(x2 - xl) \\ +0.5W_{13}(x_3 - xl) \quad (10)$$

$$\dot{x_2} = W_{12}(x_l - x_2) \quad (11)$$

$$\dot{x_3} = W_{13}(x_l - x_2) \quad (12)$$

$$W_{ij} = \frac{1 - \frac{\delta}{|x_i - x_j|^2}}{(\Delta - |x_i - x_j|)^3} \quad (13)$$

If the protector robot senses more than one attacker agent, it calculates the consensus of their position and heads in that direction. A general robotic system has three parameters: Sense, Act and Plan. But, a reactive robotic system tightly couples the perception to the actions and there is no concept of a plan. The protector agents are carrying out a series of such reactive behaviors. The agents react the moment they sense an attacker in their Δ disk.

B. Decoy and Attack Strategy

In order to successfully complete the game, a team of robots must be able to procure the opposing teams flag. In order to do so, it must employ a decoying mechanism so that some of its agents distract the opposing team. To be more particular, the robot that is decoying must make the opponents protectors believe that it is trying to attack the flag, while actually trying to draw the protectors away from their flag. This is a scenario of one robot interacting with another. The decoying agents can use a blend of multiple deliberative behaviors. The blending must be such that although the robot appears to be exhibiting a particular behavior in the local sense, it must be exhibiting a less evident deliberative strategy in the global sense.

Our approach proposes a strategy that assumes that the team is divided into attacking agents and decoying agents. Figure 2 depicts the graph that shows the interaction between attackers. We also assume that the attacking agent has the information regarding the position of the flag. This was the agent that located the opponents flag during the explore task and therefore behaves as a leader. Starting from this, the protocol adopted by the agents is summarized below.

Initially, both the decoying and attacking agents do consensus so that the information of the flag can be transferred to the decoying agent.

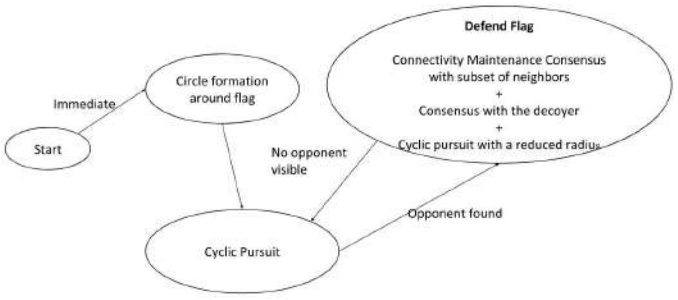
Once the decoying agent has the information for the position of the flag, it approaches the flag with a blend of Go to Goal and Avoid Obstacle behaviors, the equations for which are as follows:

$$\dot{x_{iao}} = \sum_{j \epsilon obs(i)} k_{ji}(x_i - x_j) \quad (14)$$

$$\dot{k_{ji}} = \frac{c_1}{||x_i - x_j||(||x_i - x_j||^2 + c_2)} \quad (15)$$

$$\dot{x_i} = w_1(x_g - x_i) + w_2(x_{iao}) \quad (16)$$

Here, x_g is the position of the goal.



Defend Flag
Connectivity Maintenance Consensus with subset of neighbors + Consensus with the decoyer + Cyclic pursuit with a reduced radius
Circle formation around flag
Immediate
Start
No opponent visible
Opponent found
Cyclic Pursuit

Fig. 5. Finite State Machine of Protectors

While going towards the flag, the decoyer encounters the protectors of the opponent teams. In this case, it switches to the decoy behavior. This is illustrated in the FSA of the decoyer as in Fig. 6. The decoying behavior is a blend of two primitive behaviors. One of them maintains a particular distance with the protecting opponent. The second behavior slowly draws this robot toward its starting position. These behaviors are blended in such a way that the maintaining-distance behavior is weighted much higher than the returning to start behavior. This makes it difficult for the protectors to perceive that they are being decoyed. The mathematics for maintaining distances is the same as those used in equations 11 - 13.

In order to test the combination of protect and decoy behaviors explained above, we employ the following case-based approach.

- The first scenario with respect to the protectors is that they create a static formation around the flag. The attacking agent has two behaviors: Go to goal and Obstacle avoidance. Here, there is no concept of attacker and a single agent does both the tasks. The deliberative behavior of the attacker carries out Go to goal and switches to obstacle avoidance behavior when it senses a protector. The attacking agent successfully captures the flag. Equations of obstacle avoidance:
- The second scenario is when the protectors are doing cyclic pursuit around the flag. The attacking agent will not be able to get to the flag if it follows hard constraints on the obstacle avoidance. But, if it has a higher velocity than the protecting agents then it maybe possible to break the cyclic pursuit.
- The third scenario is when the protectors have both the cyclic pursuit and tag attacker behavior. In this case a single attacking agent will never be able to capture the flag.
- The last scenario is what brings out the true nature of the decoy strategy. Here, the protectors carry out cyclic pursuit and tag attacker reactive behaviors. When a decoying agent is in the vicinity of the protectors, some of them switch to tag attacker behavior. This reduces the number of agents doing cyclic pursuit leaving it vulnerable for attack by a formation of attackers.

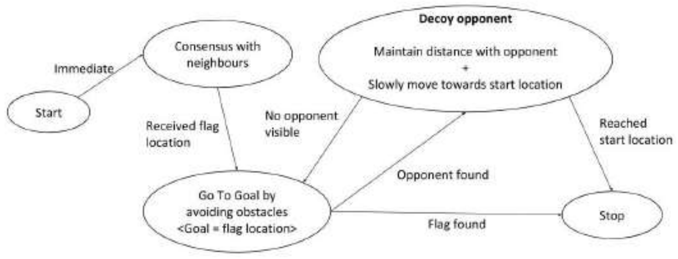

Decoy opponent
Maintain distance with opponent
+
Slowly move towards start location
Consensus with neighbours
Immediate
Start
Received flag location
No opponent visible
Reached start location
Opponent found
Go To Goal by avoiding obstacles <Goal = flag location>
Stop
Flag found

Fig. 6. Finite State Machine of Decoyers

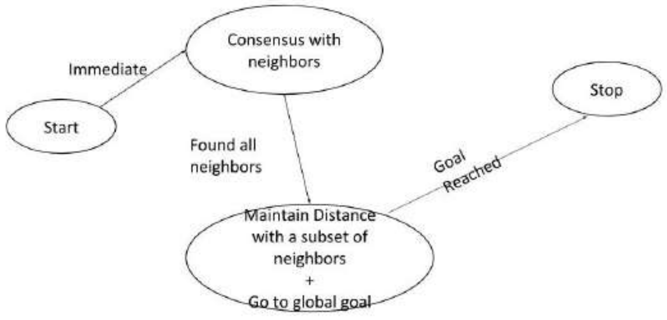


Fig. 7. Finite State Machine of Attackers

IV. Results

This section describes the experiments that were conducted to test the validity of the decoy and protect strategies proposed in the previous sections, and discusses the inferences made from the results.

The first test involved the positioning the protectors along a static formation and then checking if an attacker that uses obstacles avoidance can capture the flag. Figure 8 shows a simulation on MATLAB showing the trajectory of the robot breaking into the static formation and capturing the flag. The same can be verified from the experiment conducted in the robotarium as shown in Figure 13, 14. This called for a better protect strategy and hence, cyclic pursuit behaviour was implemented for the protector while attacker did the same. In this case, the attacker does not go through easily and requires a more complicated protocol to go through the defence as shown in Figure 9.

Second, we tested decoying behaviour of the attacking team. In this case, the protectors were assumed to be reactive. So, when the attackers come close to them, they try to follow the attackers and tag them. Figure 10 shows the decoyer taking a few defenders out of the cyclic pursuit. Now, the attacking team can break through the defence by employing a decoy strategy and thereby having a better probability to break the defence and capture the flag. Figure 11 shows a few protectors getting decoyed by one attacking agent and the other team ready to attack and capture the flag. Without an attacking team, the decoyer might get neutralized as seen in Figure 15. Hence, once the protectors get decoyed, an attack should happen from a different location as can be seen in Figure 14 where the decoyer is distracting the defenders while an attacking team is ready for an attack.

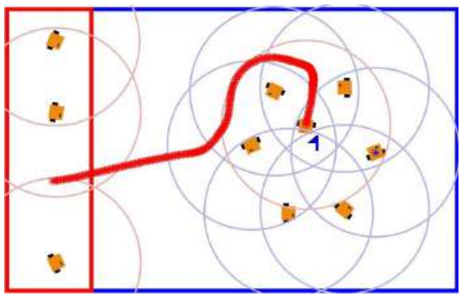


Fig. 8. An illustration of the disadvantage of using formation control as a protect strategy. An attacker can easily avoid the protectors and capture the flag. (The attackers trajectory is shown in red).

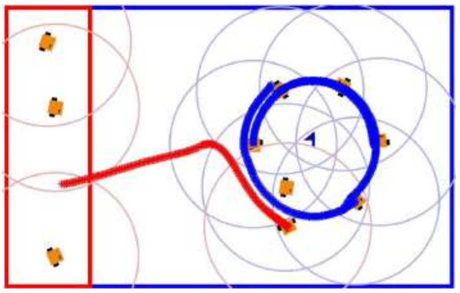


Fig. 9. An illustration of cyclic pursuit as a protection strategy. The flag is not captured in this case, since the attacker is strict about not colliding with the protectors. (The trajectory of the protectors are shown in blue and that of the attacker is shown in red).

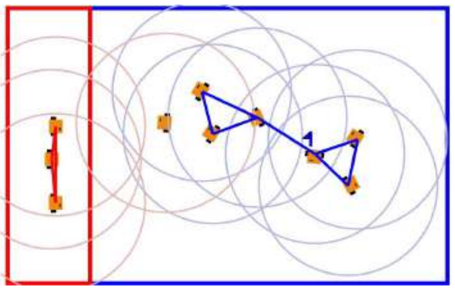


Fig. 10. Protectors exhibiting a reactive behavior can be decoyed by an opponent. The the figure shows a simulated case where the protectors that are initially preforming cyclic pursuit are forced away from the flag by a decoying agent.

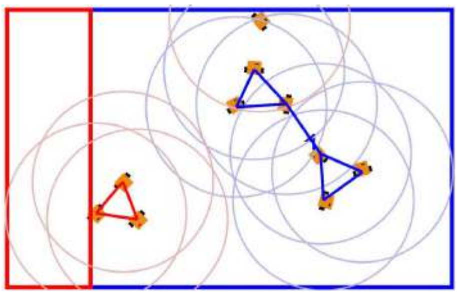


Fig. 11. The other attackers approach the flag together in another direction from where it can be captured easily after the decoy has been successful.

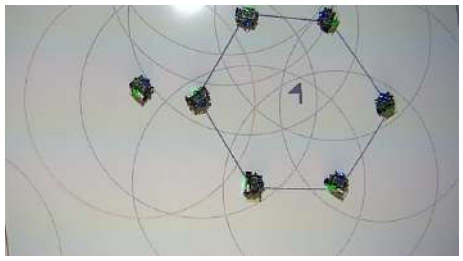


Fig. 12. Protector: Circle Formation, Attacker: Flag Capturing Process

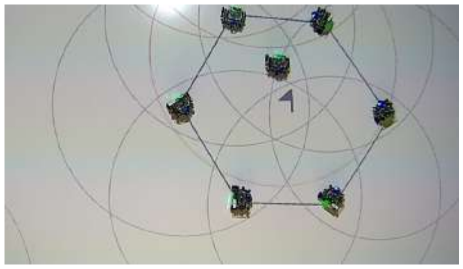


Fig. 13. Protector: Circle Formation, Attacker: Flag Captured

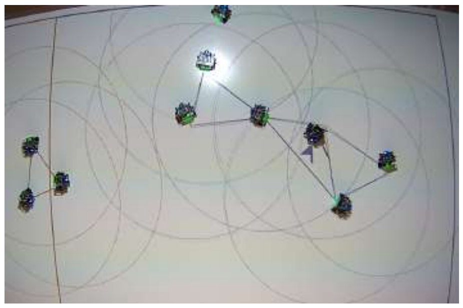


Fig. 14. Cyclic pursuit is broken by the decoy and the attacking team is approaching in another direction

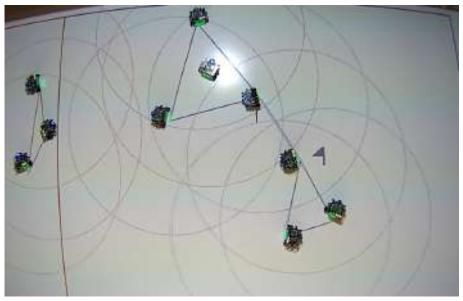


Fig. 15. Decoy neutralized by the Protectors.

V. Conclusion

In this project our motivation has been to study the application of network control algorithms in capture the flag game. We have depicted the strengths and weakness of the our protect and decoy strategies with respect to this game. The usage of distributed control algorithms depicts the ”strength in numbers” and is hence widely used in military applications. We have discussed the use of cyclic pursuit and connectivity maintenance in the context of protectors which depict a highly switched system. On the side of the attackers/decoyers connectivity maintenance, go to goal and obstacle avoidance is being carried out. We have tested the algorithms with wedge graph which has more constrained sensing capabilities than an delta-disk. The future scope of this project exists in extending this framework with more real-life scenarios where more constrained sensing capabilities are used (wedge graphs). ”When an unstoppable force meets an immovable object”

References

[1] Mesbahi, Mehran, and Magnus Egerstedt. Introduction. Graph Theoretic Methods in Multiagent Networks, STU - Student edition ed., Princeton University Press, 2010, pp. 313. JSTOR, www.jstor.org/stable/j.ctt1287k9b.5.

[2] Mesbahi, Mehran, and Magnus Egerstedt. Formation Control. Graph Theoretic Methods in Multiagent Networks, STU - Student edition ed., Princeton University Press, 2010, pp. 117158. JSTOR, www.jstor.org/stable/j.ctt1287k9b.10.

[3] Jin Dai, Shanying Zhu, Cailian Chen, Xinping Guan, Connectivity-Preserving Consensus Algorithms for Multi-agent Systems, IFAC Proceedings Volumes, Volume 44, Issue 1, 2011

[4] Pickem, D.; Glotfelter, P.; Wang, L.; Mote, M.; Ames, A.; Feron, E. Egerstedt, M., ”The Robotarium: A remotely accessible swarm robotics research testbed” 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore