Overcoming Sparse Reward and Local Optima – Multi-agents Leader-Followers

Luke Liem
Department of Computer Science, UCSD
San Diego, CA 92108
A53231779
lliem@eng.ucsd.edu

Abstract

Local optima and sparse reward are two especially challenging problems for reinforcement learning (RL).

In single-agent RL, the standard approach to overcome these problems is to increase agent exploration. This is typically accomplished by increasing the temperature parameter of the softmax function or the epsilon parameter of the ε-greedy function during action selection.

In a multi-agent setting, these approaches may not be as effective. In our experiments, we discovered that increasing the temperature parameter of softmax can lead to worse results in our game. Also, even when a number of the agents chanced upon the global optimum, they have no way of communicating their findings to the other agents.

We propose a cross-functional team-based MARL framework for solving problems involving local optima and sparse reward. Under this framework, a cross-functional team consisting of a *strategist, scouts, leaders and followers* work together so that this team of agents can achieve the global optimum more effectively and efficiently.

Our current paper focuses on the leader-follower aspect of this framework. A team of agents are given a *follower* team culture during training. Under this culture, the *follower* agents are given imaginary rewards for assembling within a "target zone" (which represents the *leader*). The *leader* (target zone) is assumed to have a map of the game space and will guide its *followers* to the area of the game space where they can achieve the global optimum.

1. Introduction

Strategic Intelligence

In "Team-based multi-agent reinforcement learning", we demonstrated that MARL can be vastly simplified by (1) separately encoding agent and organization intelligences, (2) organizing agents under teams and (3) providing them with US versus THEM context. This new approach enables teams of agents to easily exceed the performance of multiple independent agents trained under "state-of-the art" MARL algorithms. For example, team reward enables agents to specialize, which enables a team of specialized agents to execute a dominating strategy over the other agents.

We will use the example of a Roman legion to illustrate this separate encoding of organization versus agent intelligence. The agent intelligence represents the capabilities of the individual legionnaire to march, fight, dig trench and camp. The organization intelligence represents the legion's organizational structure, its maneuvers and special formations, the way it sets up camp, and so on. Through a system of rewards and punishments defined by the legion's culture, simple unsophisticated Italian peasants can be transformed into a highly sophisticated and effective organization which dominated the Mediterranean world for centuries.

We propose that there is yet another level of intelligence over that of the organizations and the agents, and it is called strategic intelligence. Using the analogy of the Roman legion again, its strategic intelligence is embodied within the legate (leader of the legion) and his staff, who decide who will be its enemy, where to maneuver the legion, whether to fight or retreat, what formations to use, and so on.

Table 1: Strategic, Organization and Agent Intelligences

Intelligence	**Roman Legion Analogy**
Strategic	The ability of the Legate and his staff to define the mission for the legion, to decide where, when and how to achieve its mission
Organization	Capabilities of the legion to execute large scale maneuvers, to build defensive camps, to lay siege to cities and so on
Agent	Capabilities of an individual solder to march, fight, dig trench and camp

Cross-functional Team-based MARL

Solving more difficult problems with local optima and sparse reward requires strategic intelligence. In team-based MARL, a team of agents can become very effective at solving one specific problem. But in many real-life situations, the problem is not well-defined, and may evolve and change over time. Even for the simple Gathering game, we can debate whether the problem is about maximizing the common goods or developing a dominating strategy such that one team can dominate over everyone else.

We propose that solving these types of problems requires a *strategist* with strategic intelligence. It will need to decide what the problem should be, gather information about the problem and the environment, conduct the trainings necessary to give the team the ability to solve that problem. Since the *strategist* has the ability to redefine the problem based on context and insight, this new type of team can solve an even larger variety of problems than team-based agents.

Under our proposed Cross-functional Team-based Multi-agent (CTMA) framework, a cross-functional team will be composed of a *strategist*, as well as *scouts, leaders and followers*. The framework is a 4-step process which we illustrate in Figure 1-4.

In Step 1, the *scouts* explore the actual game space and use their experiences to build multiple simulation environments. Simulation environments are simplified versions of the actual environments and are configured according to the problem or problems defined by the *strategist*.

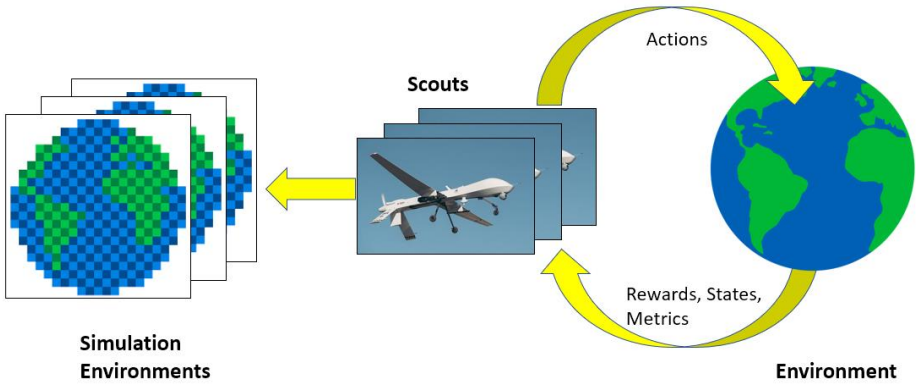

Figure 1: Step 1 – *Scouts* explore the game environment and generate multiple simulation environments

In Step 2, the *strategist* uses the simulation environments to train the *followers* to follow their leader in a variety of maneuvers. This results in a set of *follower* policies that is somewhat decoupled from the *leader* policies.

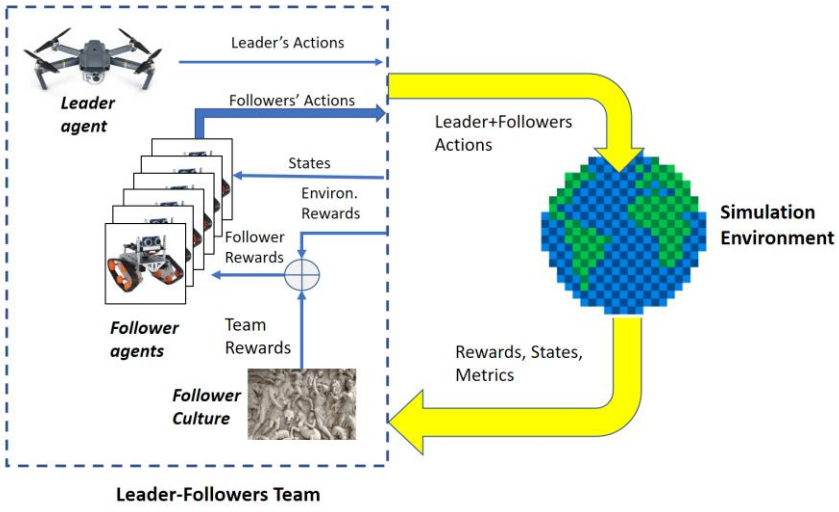

Figure 2: Step 2 – The *followers* learn a set of *follower* policies to follow their *leader* in simulation environments

Then in Step 3, the *strategist* uses the simulation environments to train the *leader* to lead its *followers* in a variety of maneuvers. This results in a set of *leader* policies that is independent of the *follower* policies.

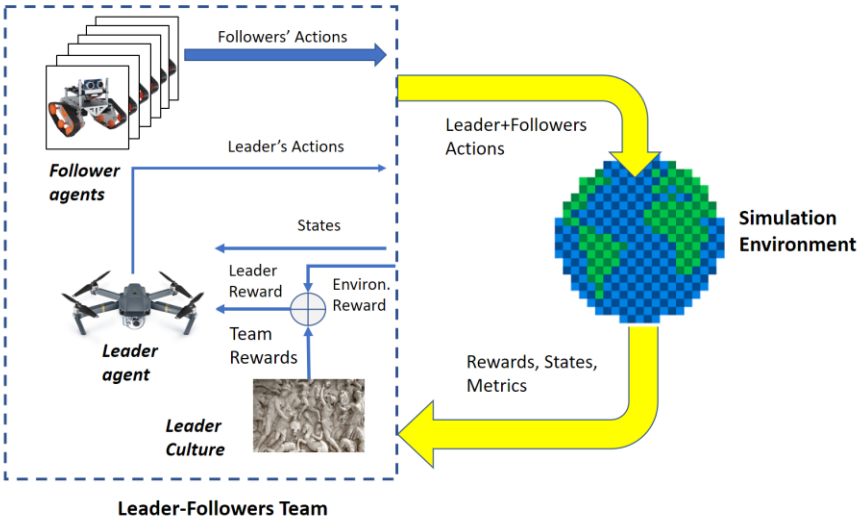

Figure 3: Step 3 – The *leader* learns a set of *leader* policies to lead its *followers* in simulation environments

Finally, in Step 4, the *strategist* learns a master policy which selects the appropriate *leader* and *follower* policies in the actual game environment. The selection of these policies will be based on the *strategist's* perception of what problem the team is facing in the game environment. Thus, this type of team is capable of solving a multitude of problems instead of a single specific problem.

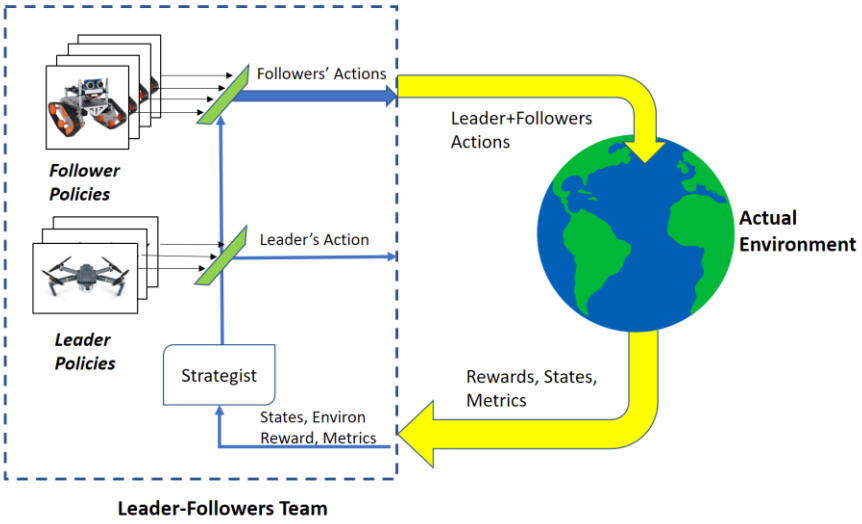

Figure 4: Step 4 – The *strategist* learns a master policy that selects *leader* and *follower* policies based on problems encountered in the actual game environment.

3. Environment

The Crossing

To experiment on the CTMA framework, we have created the Crossing game environment. The Crossing is a more difficult version of the Gathering, which is a partially observable Markov game defined in Deep mind's 2017 paper on Sequential Social Dilemmas [1].

In our paper on team-based MARL, we used the Gathering environment to train teams of agents to gather from one pile of apples located in the center of the game space [2]. For the Crossing, there are two piles of apples in the game space. The first pile of only 3 apples is located close to the starting points of the agents. The 2nd pile with 13 apples is located 37 pixels away from the first pile, and cannot be immediately observed by these agents.

We thus build into the Crossing the two challenging problems for RL:

1. Local optima – the 1st smaller pile of apples immediately observable by the agents
2. Sparse reward – the long distance (with no reward) which an agent needs to travel to get to the 2nd larger pile of apples

The Crossing environment also supports the roles of *leader* and *follower* agents within a team. It allows the *leader* agent to define a "target zone" which the *follower* agents are trained to assemble in. The *leader* agent is envisioned to be a drone that is hovering atop the "target zone" and is thus not physically represented in the game space.

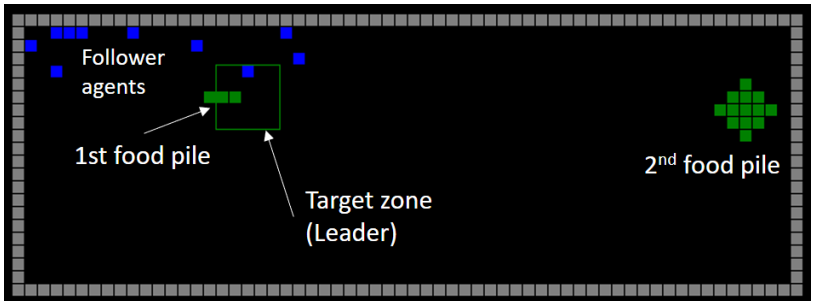

Figure 5: The Crossing Game Environment

Other than what have been mentioned above, the Crossing is identical to the original Gathering:

- 10 *follower* agents are organized into one team with a *leader* (target zone):
  - Team "Vikings" (*follower* agent 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) – blue dots
  - target zone (the *leader*) – green frame
- A *follower* agent has 8 actions:
  - FORWARD = 0
  - RIGHT = 1
  - BACKWARD = 2
  - LEFT = 3
  - ROTATE_RIGHT = 4
  - ROTATE_LEFT = 5
  - LASER = 6
  - NOOP = 7
- It receives a reward of +1 when it eats a green apple (green pixel)
- The eaten apple regenerates itself 15 game steps after it has been consumed
- By firing its laser, the *follower* agent can tag out all the other agents that are in its observation space. The agent does not receive any reward for firing its laser or tagging

out other agents.

- An agent does not receive any reward or penalty for being tagged, but is kept out of the game for 25 game steps, after which it is respawned.

***Follower* Agents**

The interaction between Crossing and the teams of *follower* agents is structured as a partially observable Markov game. The true state of the game from a *follower* agent's viewpoint is represented by 7 frames which identify:

1. Location of the apples
2. Location of the US agents (agents of the same team)
3. Location of the THEM agents (agents of different teams)
4. Location of the walls
5. Not used (reserved for future feature)
6. Not used (reserved for future feature)
7. Location of the target zone of its *leader*

The game's true state is only partially observable by each *follower* agent through an observation space of 10x20 pixels as shown in Figure 6 and 7.

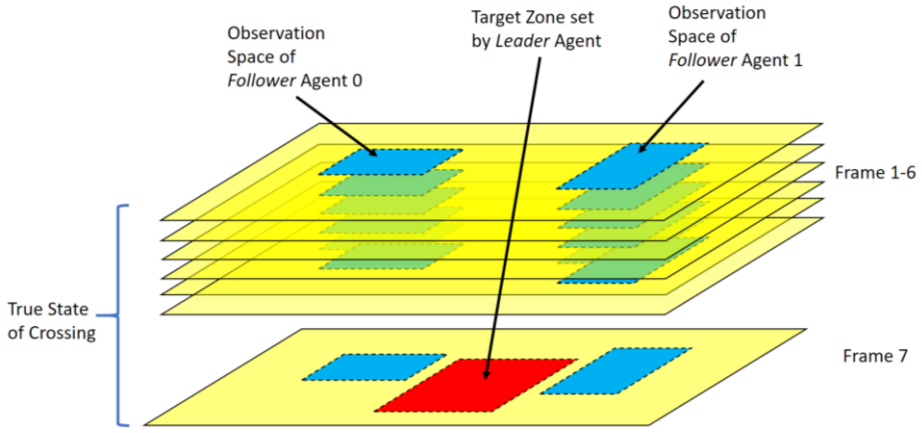

Figure 6: Each *follower* agent has an observation space which is a partial view of the true state of the Crossing environment.

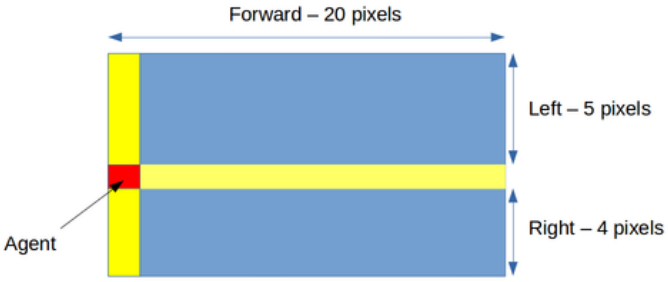

Figure 7: A *follower* agent's observation space.

The agent intelligence is encoded within the agent's policy which is parametrized with a convolutional neural network (CNN) as shown in Figure 8. The 7x10x20 observation space of the agent returned by Crossing is inputted as a 7-channel input into the CNN. The 1st

convolutional layer take this and convolute it into an output of 16x10x20. The 2nd convolutional layer then downsizes this output from the 1st layer into an output of 16x5x10. The 3rd and last convolutional layer further downsize the output from the 2nd layer into an output of 16x3x8. This last output is stretched into a single vector of 384 and entered as input into a fully connected linear neural network with 8 outputs. These outputs are softmax-ed to arrive at the probability distribution for the 8 possible actions for the agent.

The *follower* agent's RL is thus based on the generic REINFORCE policy gradient algorithm which maximizes its individual reward. It does not require any policy parameters from other agents.

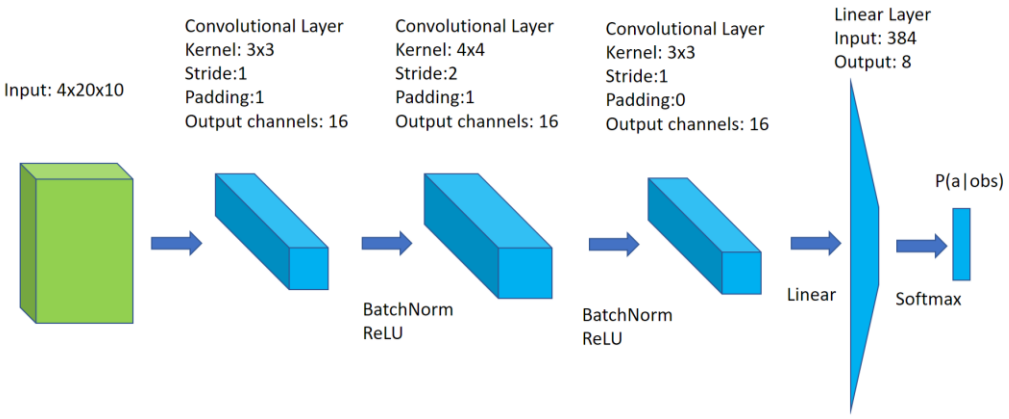

Figure 8: The *follower* agent's CNN policy network

Leader Agents

In the Crossing, the *leader* agent is envisioned as a drone hovering over the target zone and is represented in the game space as a green frame.

In full CTMA implementation, the *leader* will learn a trajectory in a separate simulation environment for guiding its *followers* to the area of the game space where they can achieve the global optimum (the 2nd food pile). For this current paper, we have simply hard-coded eight trajectories for the *leader* (see Figure 10).

Pacifist Follower Culture

We encode the team's organization intelligence into its *pacifist follower* culture (Table 2). During training, this culture doles out "imaginary" team rewards to its members on top of the rewards gathered by these agents from the environment (apples). Penalty for firing the laser is -1.0 while reward per step for being in the target zone is +5.0.

Table 2: *Pacifist Follower* Culture

Culture	**Agent Reward Calculation**	**Parameter**
Pacifist Follower	$agent\ reward = apples - penalty \times laser\ fire + reward \times in\_target\_zone$	penalty per firing of laser; reward per step for being inside the target zone

4. Methods

We conduct experiments to measure the effectiveness of CTMA's leader-follower team structure in achieving the global optimum. We define its effectiveness by two metrics:

1. The number of agents that have crossed over to the 2nd food pile
2. The average apples gathered per agent

The first metric, defined as the number of agents who cross over to the larger 2nd food pile, is the primary indication that the team has effectively overcome the local optima and sparse reward problems.

We compare these two metrics of a leader-follower team against those of a team of agents trained using optimized temperature parameter of the softmax function (Baseline).

Baseline

For Baseline, we train a team of 10 agents with *pacifist* culture (penalty of -1.0 for firing a laser) for 3000 game episodes, varying the starting temperature parameter and the number of game steps per episode. For each starting temperature, we anneal the temperature of the softmax to 1.0 linearly over the 3000 episodes. We save these agents' models every 500 episodes, thus generating sets of agent models at 500, 1000, 1500, 2000, 2500 and 3000 episodes.

Table 3: Baseline training parameters

Starting Temp	Game steps/episode
1.0	300, 600
1.25	300, 600
1.5	300, 600, 1200
2.0	300, 600, 1200
4.0	300, 600, 1200
8.0	300, 600, 1200

After the agents have been trained, for each set of (starting temp, game steps, episodes trained), we conduct repeated game play (30 episodes at 500 game steps per episode) for a team of 10 agents loaded with their saved models. We average over the results of these games to calculate the metrics defined above.

Followers – Static Target Zone

In static target zone training, we train a team of 10 agents with *pacifist_follower* culture (penalty of -1.0 for firing a laser, reward of +5.0 for each step the agent is in the target zone) to assemble within a stationary target zone over 3000 game episodes at 300 game steps per episodes. We vary the starting temperature and the location of the 5x5 target zone, and we save these agents' models every 500 episodes.

Table 4: *Followers* – Static Target Zone training parameters

Location of 5x5 Target Zone	Starting Temp
(20,0)	1.5, 2,0
(20,1)	1.5, 2,0
(20,2)	1.5, 2,0

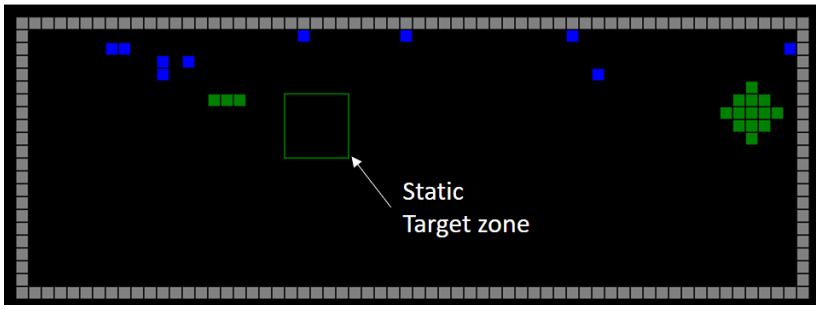

Figure 9: Training *followers* to assemble within a stationary target zone.

After all the agents have been trained, for each set of (starting temp, target zone location, episodes trained), we conduct repeated game play (30 episodes) whereby a team of 10 agents loaded with their saved models follow 8 pre-defined target zone trajectories (illustrated in Figure 10) to the 2nd food pile. Then we average over the results of these games to calculate the metrics for each trajectory.

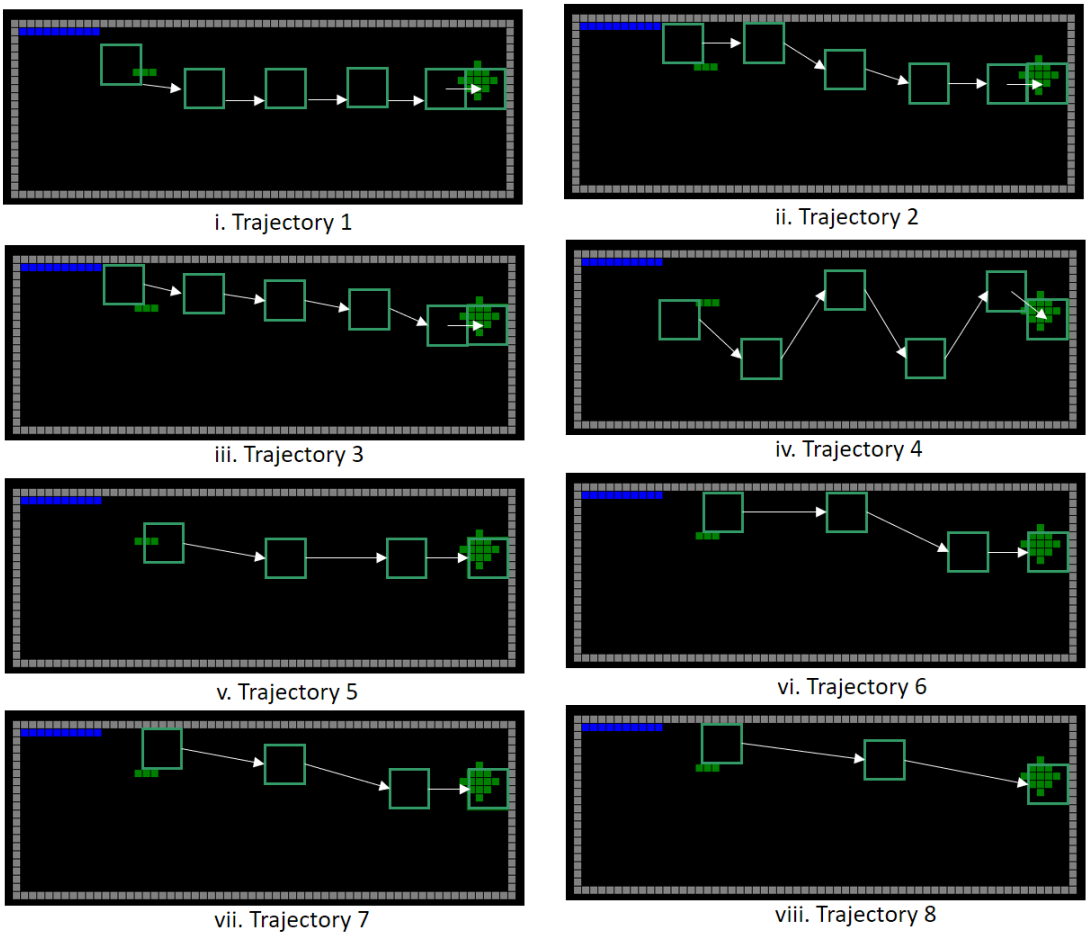

Figure 10: 8 pre-defined trajectories used in repeated game plays.

The target zone will pause 100 steps at each point of the trajectory, except for the last point, where it will pause for 400 steps. This is because while the agents are assembled in the target zone on their way to the 2nd food pile, they are not gathering any apples. So we give the agents 400 games steps to gather apples at the last point of the trajectory, so that a fair comparison can be made with the Baseline agents with respect to the 2nd metric (average apples gathered per agent).

Followers – Zone Trajectory

In zone trajectory training, we train the team of 10 *pacifist_follower* agents to follow a moving target zone to the 2nd food pile over 3000 game episodes. We vary the trajectory of the target zone, the starting temperature and the number of game steps per episode. We save these agents' models every 500 episodes.

For the two trajectories used during training, we allow the target zone to pause 100 steps at each stop on its way to the 2nd food pile, and 300 steps on the last stop atop the food pile, so that the agents can learn to gather apples while inside the target zone.

Table 5: *Followers* – Zone Trajectory training parameters

Zone Trajectory	Temperature	Game steps/episode	Trajectory Detail
Trajectory 7	1.5, 2,0	600	stop #1 – 100 steps stop #2 – 100 steps stop #3 – 100 steps stop #4 – 300 steps
Trajectory 8	1.5, 2,0	500	stop #1 – 100 steps stop #2 – 100 steps stop #3 – 300 steps

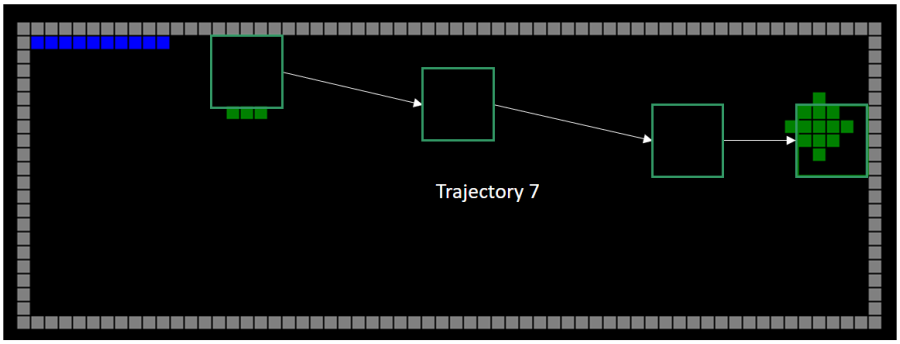

Figure 9: Training *followers* using a pre-defined target zone trajectory.

After all the agents have been trained, for each set of (starting temp, trajectory, episodes trained), we conduct repeated game play (30 episodes) whereby a team of 10 agents loaded with their saved models follow the same 8 pre-defined target zone trajectories to the 2nd food pile. Then we average over the results of these games to calculate the metrics for each trajectory.

5. Results

Baseline

Detailed results of Baseline are presented in Table 5. By optimizing the starting temperature and game steps per episode, we manage to induce 6.3 agents (out of 10) to cross over to the 2nd larger food pile. We get the best result at starting temp=1.0 and game steps=300. To our surprise, fewer agents cross over to the 2nd food pile when we increase either the starting temperature parameter or the game steps parameters.

When temperature becomes too high, the agents appear to no longer learn effectively and we witness increasing number of agents who become motionless. Increasing the number of games steps per training episodes also adversely affects our results. It is possible that the agents become over-trained in longer episodes and end up learning less effective and general policies.

Table 5: Baseline Results – Green cells represent instances when more than 5 agents have crossed over to the 2nd food pile to gather apples

Av. Apples/agent

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=1.0/300steps	27.5	35.4	36.1	36.0	31.4	35.5
Temp=1.0/600steps	29.7	34.1	33.3	35.3	35.3	37.2
Temp=1.25/300steps	24.6	34.1	35.6	35.2	34.3	35.5
Temp=1.25/600steps	31.2	34.9	35.6	37.1	36.8	27.3
Temp=1.5/300steps	18.0	34.2	36.5	36.5	33.7	37.5
Temp=1.5/600steps	29.5	33.0	33.1	34.9	35.6	9.3
Temp=1.5/1200steps	27.7	30.0	31.1	34.7	35.5	32.9
Temp=2.0/300steps	19.6	27.7	34.4	35.9	35.2	36.8
Temp=2.0/600steps	26.2	32.7	36.1	37.3	36.4	35.6
Temp=2.0/1200steps	32.1	37.1	38.8	35.4	14.1	35.9
Temp=4.0/300steps	7.3	31.1	35.5	37.2	33.8	30.8
Temp=4.0/600steps	10.1	30.2	32.1	36.8	33.6	18.9
Temp=4.0/1200steps	9.0	15.5	14.0	25.4	10.7	9.2
Temp=8.0/300steps	1.3	15.2	30.3	33.0	32.4	29.3
Temp=8.0/600steps	5.7	29.6	35.0	33.3	23.8	9.3
Temp=8.0/1200steps	7.7	25.2	30.9	35.4	30.9	13.7

Agents Crossed (2nd food pile)

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=1.0/300steps	6.3	6.0	5.1	4.8	4.9	4.3
Temp=1.0/600steps	3.5	3.4	4.9	5.0	4.8	5.2
Temp=1.25/300steps	3.7	5.1	5.3	4.5	3.4	3.4
Temp=1.25/600steps	3.6	4.9	5.6	4.1	3.5	1.1
Temp=1.5/300steps	4.5	4.7	5.1	5.6	2.7	4.5
Temp=1.5/600steps	2.3	2.6	1.0	1.2	1.0	0.0
Temp=1.5/1200steps	1.9	2.0	1.8	2.9	2.7	1.1
Temp=2.0/300steps	2.8	3.6	3.5	4.1	3.2	3.4
Temp=2.0/600steps	3.4	3.1	3.0	2.8	2.0	2.0
Temp=2.0/1200steps	4.0	3.6	3.1	1.3	0.5	1.0
Temp=4.0/300steps	1.1	3.6	4.0	3.6	2.7	1.8
Temp=4.0/600steps	1.2	2.9	2.4	1.9	1.8	0.6
Temp=4.0/1200steps	0.1	0.5	0.4	2.1	0.1	0.2
Temp=8.0/300steps	0.0	1.3	2.7	3.5	3.2	2.7
Temp=8.0/600steps	0.3	2.9	3.1	1.7	1.1	0.0
Temp=8.0/1200steps	0.4	1.2	1.1	1.9	1.3	0.2

Followers – Static Target Zone

Detailed results of *followers* trained using static target zones are presented in Table 8 of Supplementary Materials. Even though the original intent of this training scenario is for the agents to just assemble within a stationary target zone, repeated game play results indicate that the agents have actually learned the following skills:

1. Stop firing their lasers
2. Look for a target zone and assemble within it
3. Follow a moving target zone and assemble within it
4. Gather apples

As shown in Table 6 and Table 8, placing the stationary target zone at a slight angle to the agents at position (20,1) and (20,2) during training reliably get more than 6.0 agents to cross over to the 2nd food pile. In some cases, up to 9.0 agents cross over. This is demonstratively better than the 6.3 agents achieved under optimal temperature setting in Baseline. It appears that learning simply to follow a leader can help a team of agents overcome local optima and sparse reward.

The average number of apples gathered by these *follower* agents is lower than that of the Baseline agents. We believe this is because the training scenario focuses on training the agents to assemble within a target zone, not gathering apples.

Table 6: Average results for *follower* agents trained with static target zone at (20,1) and (20,2), compared against Baseline (temp=1.0, 300 game steps).

Agents Crossed (2nd food pile)

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=2.0	6.2	7.4	7.5	7.0	6.9	6.2
Temp=1.5	7.6	7.9	7.1	7.2	6.8	6.1
Baseline	6.3	6.0	5.1	4.8	4.9	4.3

Average Agent Rewards

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=2.0	17.8	18.8	19.5	17.1	17.1	17.7
Temp=1.5	20.9	19.6	17.1	16.8	15.6	13.9
Baseline	27.5	35.4	36.1	36.0	31.4	35.5

Followers – Trajectory

Detailed results of *followers* trained using zone trajectories are presented in Table 9 in Supplementary Materials. The more elaborate training scenario results in even better metrics. As illustrated in Table 7 and 9, it reliably gets 8.0 – 9.0 agents to cross over and gather apples at the 2nd food pile. In one case, 9.8 agents or essentially every agent in the team, cross over [3].

While the average number of apples gathered by these *follower* agents improves with respect to agents trained using static target zones, it is still lower than that of the Baseline agents.

Table 7: Average results for *follower* agents trained with zone trajectories 7 and 8, compared against Baseline (temp=1.0, 300 game steps).

Agents Crossed (2nd food pile)

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=2.0	8.0	9.2	8.8	9.1	8.7	8.2
Temp=1.5	8.9	8.9	8.8	8.6	8.1	8.9
Baseline	6.3	6.0	5.1	4.8	4.9	4.3

Average Agent Rewards

	Episodes Trained					
	500	1000	1500	2000	2500	3000
Temp=2.0	23.1	21.8	20.7	21.4	19.7	16.3
Temp=1.5	22.3	22.3	18.9	19.0	17.6	20.3
Baseline	27.5	35.4	36.1	36.0	31.4	35.5

5. Discussions

Our experiments demonstrate that a team made of up *leader* and *followers* can very reliably and effectively solve a problem with local optima and sparse reward, provided that the *leader* knows how to guide the *followers* to the area where global optimum can be achieved, as illustrated in the following video (https://youtu.be/2u9SN0EoQYo). On the other hand, the traditional approach of increasing agent exploration by optimizing the softmax function's temperature parameter does not work as reliably and effectively in a multi-agent setting.

The power of being able to bring an entire team of agents from a poor region to a good region in the game space is very critical to dominating a game. Suppose the 2nd food pile is not located 37 paces away from the 1st food pile, but is instead located in a wall-enclosed area with a single opening, and suppose there are now 3 teams competing in the game, then the first team that reaches the 2nd food pile with all its agents can use a few of its agents to seal off the wall opening with laser fire, and the rest of the team will have all the apples in the 2nd pile to themselves. This will leave the other 2 teams in a poor region of the game space competing for the 3 apples of the 1st food pile. Under the CTMA framework, we can easily train a *leader-followers* team to accomplish the above strategy.

Thus, we have demonstrated in this paper the ability of a leader to reliably lead its followers from a poor region of the game space to a good region. All that is required now is for the leader to learn how to plan an optimal trajectory balancing between reaching the 2nd food pile fast while ensuring that the majority of its agents cross over.

References

[1] J. Z. Leibo, V. F. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel. Multi-agent reinforcement learning in sequential social dilemmas. CoRR, abs/1702.03037, 2017.

[2] L. Liem. Team-based multi-agent reinforcement learning. 2019. https://github.com/LUKELIEM/team_marl/blob/master/report/CSE293_Report_Team-based_MARL.pdf

[3] https://youtu.be/6ylSxvhC5fY

Supplementary Materials

Table 8: Results of *follower* agents trained with static target zones – Green cells represent instances when more than 6 agents have crossed over to the 2nd food pile to gather apples. Blue cells represent instances when more than 8 agents have crossed.

Agents Crossed (2nd food pile)

	Episodes Trained					
	500	1000	1500	2000	2500	3000
-	For Trajectory 1					
temp=2.0, Loc=(20,0)	6.1	7.4	6.0	6.5	4.7	4.2
temp=2.0, Loc=(20,1)	7.1	8.0	7.2	6.5	6.3	5.8
temp=2.0, Loc=(20,2)	5.8	7.9	8.0	8.0	7.2	5.5
temp=1.5, Loc=(20,0)	6.9	5.4	6.2	5.6	5.8	4.3
temp=1.5, Loc=(20,1)	8.0	8.2	7.7	7.2	7.6	5.9
temp=1.5, Loc=(20,2)	7.4	7.7	7.1	7.5	7.2	7.1
	For Trajectory 2					
temp=2.0, Loc=(20,0)	6.4	7.4	7.1	6.3	5.4	4.9
temp=2.0, Loc=(20,1)	7.5	7.9	7.4	7.0	6.9	6.5
temp=2.0, Loc=(20,2)	6.1	8.2	8.3	8.0	7.9	7.2
temp=1.5, Loc=(20,0)	6.7	5.1	5.8	5.0	3.7	3.5
temp=1.5, Loc=(20,1)	8.4	8.9	8.6	7.6	7.0	6.3
temp=1.5, Loc=(20,2)	8.1	8.4	7.4	7.8	7.2	7.0
	For Trajectory 3					
temp=2.0, Loc=(20,0)	6.4	7.0	6.8	6.5	5.2	4.7
temp=2.0, Loc=(20,1)	7.7	7.9	7.4	6.9	7.3	6.6
temp=2.0, Loc=(20,2)	6.3	8.1	8.3	7.7	8.1	6.7
temp=1.5, Loc=(20,0)	7.1	5.2	6.0	5.4	5.2	3.6
temp=1.5, Loc=(20,1)	8.1	9.0	7.9	7.9	7.4	6.1
temp=1.5, Loc=(20,2)	8.0	8.5	7.3	7.6	7.1	6.8
	For Trajectory 4					
temp=2.0, Loc=(20,0)	6.1	7.3	6.7	6.2	4.7	3.9
temp=2.0, Loc=(20,1)	6.5	6.7	7.0	5.8	6.4	5.4
temp=2.0, Loc=(20,2)	5.4	7.2	7.6	7.6	7.1	6.4
temp=1.5, Loc=(20,0)	6.2	4.2	4.0	3.9	3.8	3.0
temp=1.5, Loc=(20,1)	7.1	7.8	6.3	6.5	6.9	5.3
temp=1.5, Loc=(20,2)	5.9	5.4	4.6	5.4	5.7	5.7
-	For Trajectory 5					
temp=2.0, Loc=(20,0)	5.5	6.9	6.3	6.1	4.3	4.1
temp=2.0, Loc=(20,1)	6.0	7.3	6.5	5.6	5.8	5.7
temp=2.0, Loc=(20,2)	5.1	6.7	7.8	7.5	6.8	6.1
temp=1.5, Loc=(20,0)	5.1	4.4	5.5	5.2	5.4	3.5
temp=1.5, Loc=(20,1)	7.4	8.1	6.9	6.8	7.4	5.8
temp=1.5, Loc=(20,2)	7.1	6.8	7.1	7.4	6.6	6.7
	For Trajectory 6					
temp=2.0, Loc=(20,0)	6.0	7.2	6.8	6.1	5.0	4.1
temp=2.0, Loc=(20,1)	7.1	7.3	7.3	5.9	6.5	5.8
temp=2.0, Loc=(20,2)	5.0	6.9	8.0	7.8	7.0	6.6
temp=1.5, Loc=(20,0)	5.8	4.9	4.0	4.2	3.1	3.6

Average Agent Rewards

	Episodes Trained					
	500	1000	1500	2000	2500	3000
-	For Trajectory 1					
temp=2.0, Loc=(20,0)	16.2	15.5	14.7	12.6	16.5	12.5
temp=2.0, Loc=(20,1)	19.2	19.8	19.7	16.1	15.9	14.3
temp=2.0, Loc=(20,2)	20.0	21.2	21.3	20.0	20.6	18.4
temp=1.5, Loc=(20,0)	16.6	14.1	14.7	13.8	13.7	11.6
temp=1.5, Loc=(20,1)	23.9	21.5	19.1	20.3	16.1	13.2
temp=1.5, Loc=(20,2)	21.4	21.2	20.7	15.7	15.9	17.0
	For Trajectory 2					
temp=2.0, Loc=(20,0)	16.6	16.5	15.9	13.3	16.6	12.5
temp=2.0, Loc=(20,1)	18.8	18.9	17.9	16.4	16.3	15.0
temp=2.0, Loc=(20,2)	21.2	20.1	21.9	19.5	18.9	22.9
temp=1.5, Loc=(20,0)	15.3	12.6	13.6	11.1	9.1	11.2
temp=1.5, Loc=(20,1)	22.8	20.7	17.9	18.2	16.8	13.5
temp=1.5, Loc=(20,2)	21.7	19.3	18.1	16.0	17.2	16.8
	For Trajectory 3					
temp=2.0, Loc=(20,0)	15.4	15.0	16.2	13.5	16.4	11.5
temp=2.0, Loc=(20,1)	18.7	18.9	18.9	16.1	16.3	15.0
temp=2.0, Loc=(20,2)	20.3	19.7	20.5	19.5	18.6	20.3
temp=1.5, Loc=(20,0)	16.1	13.6	13.0	12.3	10.7	12.4
temp=1.5, Loc=(20,1)	23.0	20.4	16.8	18.3	16.2	11.8
temp=1.5, Loc=(20,2)	21.5	20.4	17.8	15.7	15.6	15.4
	For Trajectory 4					
temp=2.0, Loc=(20,0)	14.9	15.4	15.8	14.3	14.7	11.4
temp=2.0, Loc=(20,1)	16.6	18.7	18.7	14.9	16.7	16.5
temp=2.0, Loc=(20,2)	21.1	24.0	27.5	25.9	24.6	26.9
temp=1.5, Loc=(20,0)	15.9	12.0	9.7	12.7	11.3	6.3
temp=1.5, Loc=(20,1)	23.4	22.2	21.0	22.3	19.2	14.5
temp=1.5, Loc=(20,2)	19.1	20.4	16.1	14.8	14.7	16.7
-	For Trajectory 5					
temp=2.0, Loc=(20,0)	12.9	13.4	13.7	13.1	13.9	10.6
temp=2.0, Loc=(20,1)	14.5	18.5	18.4	14.2	13.4	13.9
temp=2.0, Loc=(20,2)	18.0	18.9	19.8	18.5	19.4	19.0
temp=1.5, Loc=(20,0)	12.3	11.9	12.5	12.4	11.4	10.6
temp=1.5, Loc=(20,1)	20.7	19.3	16.2	17.8	15.4	12.0
temp=1.5, Loc=(20,2)	19.2	17.7	17.6	15.2	14.5	16.4
	For Trajectory 6					
temp=2.0, Loc=(20,0)	14.3	14.5	14.0	11.8	14.1	9.5
temp=2.0, Loc=(20,1)	15.5	16.0	18.1	14.0	14.9	12.2
temp=2.0, Loc=(20,2)	17.5	17.4	19.4	18.4	17.9	22.3
temp=1.5, Loc=(20,0)	13.8	12.2	9.6	11.2	9.8	11.3

temp=1.5, Loc=(20,1)	7.5	8.4	7.7	6.7	6.6	5.0
temp=1.5, Loc=(20,2)	8.0	7.7	6.8	8.0	6.6	6.8
	For Trajectory 7					
temp=2.0, Loc=(20,0)	5.4	7.1	6.4	5.4	4.8	4.1
temp=2.0, Loc=(20,1)	6.8	7.6	7.3	6.4	6.7	6.1
temp=2.0, Loc=(20,2)	5.2	7.3	7.5	7.7	6.8	6.4
temp=1.5, Loc=(20,0)	5.4	5.0	5.2	4.3	3.2	3.2
temp=1.5, Loc=(20,1)	8.2	8.4	7.9	7.0	6.9	5.1
temp=1.5, Loc=(20,2)	7.7	7.6	6.9	7.7	6.9	6.8
	For Trajectory 8					
temp=2.0, Loc=(20,0)	5.1	7.4	6.5	5.1	5.2	4.1
temp=2.0, Loc=(20,1)	6.9	6.8	7.4	5.6	5.7	5.4
temp=2.0, Loc=(20,2)	4.9	7.3	7.6	8.0	7.4	6.5
temp=1.5, Loc=(20,0)	4.9	5.2	4.2	4.4	2.9	2.7
temp=1.5, Loc=(20,1)	7.4	7.9	6.8	5.9	5.0	4.8
temp=1.5, Loc=(20,2)	7.3	7.8	6.7	7.4	6.9	6.2

temp=1.5, Loc=(20,1)	20.2	19.8	15.9	14.6	15.5	10.3
temp=1.5, Loc=(20,2)	20.2	18.2	16.2	17.6	16.5	16.7
	For Trajectory 7					
temp=2.0, Loc=(20,0)	12.3	13.9	14.0	11.4	13.3	10.5
temp=2.0, Loc=(20,1)	16.1	17.2	17.6	14.8	14.7	13.4
temp=2.0, Loc=(20,2)	17.3	18.4	19.1	17.3	17.5	21.2
temp=1.5, Loc=(20,0)	12.5	12.2	11.6	11.1	9.8	10.3
temp=1.5, Loc=(20,1)	21.3	19.4	16.3	16.6	14.9	10.1
temp=1.5, Loc=(20,2)	19.3	17.6	16.2	16.2	15.2	14.5
	For Trajectory 8					
temp=2.0, Loc=(20,0)	11.4	13.9	13.3	11.6	12.5	8.9
temp=2.0, Loc=(20,1)	14.1	14.9	16.0	12.0	11.8	11.2
temp=2.0, Loc=(20,2)	16.5	18.0	17.5	16.2	15.8	21.1
temp=1.5, Loc=(20,0)	11.9	10.4	8.1	9.6	7.4	10.3
temp=1.5, Loc=(20,1)	19.1	18.6	14.4	14.3	12.8	9.3
temp=1.5, Loc=(20,2)	17.8	17.1	13.7	14.7	13.4	13.7

Table 9: Results of *follower* agents trained with zone trajectories – Green cells represent instances when more than 8 agents have crossed over to the 2nd food pile to gather apples. Blue cells represent instances when more than 9 agents have crossed.

Agents Crossed (2nd food pile)

	Episodes Trained					
	500	1000	1500	2000	2500	3000
	For Trajectory 1					
temp=2.0, traj 7	8.0	9.7	9.0	8.9	8.8	8.0
temp=2.0, traj 8	8.7	9.4	8.5	9.3	8.5	8.0
temp=1.5, traj 7	9.2	8.7	8.8	8.1	8.1	8.4
temp=1.5, traj 8	8.9	9.2	8.6	7.9	7.8	8.7
	For Trajectory 2					
temp=2.0, traj 7	8.1	9.6	9.0	9.4	9.7	8.7
temp=2.0, traj 8	9.1	9.3	8.8	9.4	8.6	8.4
temp=1.5, traj 7	9.3	8.8	9.0	9.1	8.9	9.2
temp=1.5, traj 8	9.2	9.8	9.0	9.2	8.3	9.3
	For Trajectory 3					
temp=2.0, traj 7	7.8	9.6	9.8	9.4	9.4	8.1
temp=2.0, traj 8	8.7	9.2	8.7	9.5	8.0	8.4
temp=1.5, traj 7	9.3	8.9	9.1	9.0	8.7	9.0
temp=1.5, traj 8	9.3	9.7	9.4	9.0	8.3	9.6
	For Trajectory 4					
temp=2.0, traj 7	7.3	9.5	8.1	7.8	8.3	7.6
temp=2.0, traj 8	7.3	8.6	7.6	8.9	9.0	8.1
temp=1.5, traj 7	8.0	6.4	7.6	6.5	6.7	6.8
temp=1.5, traj 8	8.0	8.5	8.3	8.2	8.0	8.8
	For Trajectory 5					
temp=2.0, traj 7	7.4	9.3	9.0	9.0	9.1	7.7
temp=2.0, traj 8	8.2	8.9	8.5	9.1	8.5	8.5
temp=1.5, traj 7	8.6	8.6	9.0	8.7	8.2	8.5
temp=1.5, traj 8	8.7	9.1	8.6	8.0	7.9	8.4
	For Trajectory 6					
temp=2.0, traj 7	7.4	9.3	9.3	9.2	9.0	8.3
temp=2.0, traj 8	8.1	8.9	8.4	9.5	8.1	8.3
temp=1.5, traj 7	9.0	8.6	9.3	9.0	7.6	8.7
temp=1.5, traj 8	8.8	9.5	8.2	8.8	8.1	9.5
	For Trajectory 7					
temp=2.0, traj 7	7.4	9.1	9.1	9.5	9.5	8.5
temp=2.0, traj 8	8.6	8.8	8.5	9.3	7.9	8.3
temp=1.5, traj 7	9.0	8.6	9.4	9.0	8.0	8.6
temp=1.5, traj 8	8.8	9.5	8.2	9.0	8.6	9.4
	For Trajectory 8					
temp=2.0, traj 7	7.1	9.3	9.1	9.1	9.2	7.7
temp=2.0, traj 8	8.0	8.9	8.5	9.0	8.3	8.4
temp=1.5, traj 7	9.0	8.6	9.4	9.4	7.9	9.2
temp=1.5, traj 8	8.7	9.8	8.4	9.0	8.5	9.8

Average Agent Rewards

	Episodes Trained					
	500	1000	1500	2000	2500	3000
	For Trajectory 1					
temp=2.0, traj 7	25.2	25.5	25.6	21.9	21.0	18.6
temp=2.0, traj 8	24.4	21.9	19.6	21.8	21.5	17.1
temp=1.5, traj 7	24.1	28.1	21.0	20.9	18.4	18.2
temp=1.5, traj 8	24.3	21.2	19.3	17.1	19.5	23.4
	For Trajectory 2					
temp=2.0, traj 7	23.5	21.1	20.7	20.6	19.9	15.2
temp=2.0, traj 8	25.5	21.8	19.7	23.0	18.8	18.3
temp=1.5, traj 7	22.6	22.9	21.1	20.8	20.3	20.5
temp=1.5, traj 8	24.9	20.7	17.8	17.4	15.7	21.4
	For Trajectory 3					
temp=2.0, traj 7	23.5	21.9	22.2	21.3	19.6	14.8
temp=2.0, traj 8	24.7	20.0	18.7	22.7	17.4	18.3
temp=1.5, traj 7	23.1	23.3	20.2	21.0	19.4	19.4
temp=1.5, traj 8	23.8	20.2	18.8	17.4	15.2	20.5
	For Trajectory 4					
temp=2.0, traj 7	25.7	30.1	27.9	28.1	27.0	19.0
temp=2.0, traj 8	25.9	25.5	23.5	28.2	24.0	18.1
temp=1.5, traj 7	25.1	28.4	21.9	24.6	22.7	21.3
temp=1.5, traj 8	23.3	26.5	21.3	19.5	20.5	23.9
	For Trajectory 5					
temp=2.0, traj 7	23.1	22.1	22.1	19.6	19.7	16.1
temp=2.0, traj 8	21.1	20.5	19.8	19.3	18.3	16.0
temp=1.5, traj 7	22.2	25.5	20.0	20.9	19.2	18.9
temp=1.5, traj 8	21.4	20.0	16.8	16.2	17.0	21.7
	For Trajectory 6					
temp=2.0, traj 7	22.2	19.6	19.6	19.4	19.4	14.5
temp=2.0, traj 8	21.4	20.3	19.5	21.3	17.6	13.9
temp=1.5, traj 7	20.5	21.1	19.5	20.8	15.9	18.4
temp=1.5, traj 8	20.8	19.4	15.8	16.6	15.0	20.5
	For Trajectory 7					
temp=2.0, traj 7	21.6	19.8	19.1	19.4	19.9	15.1
temp=2.0, traj 8	22.0	19.2	17.3	19.2	16.3	14.7
temp=1.5, traj 7	20.8	21.5	19.0	19.4	16.5	17.7
temp=1.5, traj 8	21.4	19.7	15.4	16.9	15.1	21.3
	For Trajectory 8					
temp=2.0, traj 7	19.8	19.9	18.8	18.7	18.6	16.2
temp=2.0, traj 8	19.8	19.3	18.2	18.5	15.6	15.4
temp=1.5, traj 7	19.2	20.0	18.6	19.0	16.1	18.2
temp=1.5, traj 8	19.7	18.8	16.2	15.4	14.7	20.1