Emergence of Order in Leader-Follower Boids-Inspired Systems

Israel Dunk
School of Engineering and Information Technology
University of New South Wales, Canberra
Canberra, Australia
i.dunk@ad.unsw.edu.au

Hussein Abbass
School of Engineering and Information Technology
University of New South Wales, Canberra
Canberra, Australia
h.abbass@unsw.edu.au

***Abstract*—The design of rules governing the behaviour of a follower in a leader-follower system is a non-trivial task. In this paper, we investigate three Boids-like behavioural rules: alignment, attraction and separation. We systematically design and investigate the impact of different reward functions on the three behaviours using evolutionary computation methods. A Learning Classifier System initially starting from a set of random rules is used to evolve the Follower behaviour of agents within a simulated leader-follower environment. We present a series of systemic experiments to reveal and understand the interdependency between the incrementally-designed reward functions and the performance of the Follower in conforming to the three parameters. We demonstrate that an incremental and systemic design of the reward function is sufficient to reproduce Reynolds' rules from zero domain knowledge and that the solution is robust against shifts caused by evolutionary dynamics.**

Keywords*—*Learning Classifier System; Simulation; BOIDS; Evolutionary Computation

I. Introduction

Evolving behaviours, rules and procedures provides an interesting alternative to hard coded solutions; especially in novel environments where autonomous agents could be exposed to environmental conditions that were not available at design time. The ability to design a system that starts with no domain knowledge (an empty or randomly generated rule set), and then allowing the system to evolve the correct knowledge through a series of interactions with the environment, over successive generations, can reveal novel solutions to problems where the relationship between system inputs (agent's perceptions) and output (agent's actions) can be difficult to understand or adequately define. Moreover, it may even shed some light on how nature as a designer has evolved the complex autonomous behaviours we see today.

Our motivation is to understand if and how Boids-like rules can be discovered through simple reward functions and if such behaviours are evolutionary stable strategies. We seek to develop structured Boids-like behaviour in an evolutionary simulation by defining objective functions that serve as a reward mechanism for the simulation. These functions describe the desired behaviour of the system without needing to make any assumptions as to how the system will achieve this behaviour. Rewards under this scheme are apportion by comparing the past system state to the current system state. A Learning Classifier System (LCS) has been chosen as the means of knowledge discovery. LCS systems generally excel at finding a set of rules/Classifiers that describe a required input-output mapping.

The original work from which this paper draws from is Craig Reynolds' paper from 1987 [1]. In his paper, Reynolds described a model that did not seek to represent all the characteristic of a flock of living birds, but instead emulates the elements of flocking which he believed resulted in the visual characteristics of flocking behaviour. These specifically being; the propensity of a member to want to align itself with the direction of movement of the flock (alignment), the desire to centre itself in the flock (attraction/cohesion) and the desire to avoid collision with other members of the flock (separation). It is from these factors that the experimental parameters for this paper have been derived. Though, for this simulation, Followers attempt to arrange themselves around the leader, whom is the centre of the flock for this case. The motivation for this is to show that the system is externally controllable. Recent papers such as those by Benes and Hartman [2] have sought to extend these visualizations by modelling the more complex behaviours of flocks of birds, with the goal again being to make visually appealing simulations of flocking birds. Evolutionary simulation has further been applied to swarming such as the work of Ferrante et.al. [13] which attempts to parallel the natural evolution of task specialized behaviour in swarms.

Work has also been conducted in the area of flock statistics. For instance, Harvey et.al. [11] look at the minimum required parameters for flocking in a Boids model using chaos measures. This in particular pertains to this paper as they indicate that flocking behaviours can be achieved with a limited number of the 3 forces used in Reynolds' model.

Kovacs and Kerber [5], and Takadama et.al. [7] discuss some of the issues facing implementations of LCS', particularly in multi-agent environments. In particular, Takadama et.al. [7] propose a solution using the Organisational Learning Classifier System which implements rule sharing between agents in order to allow the system to reach a solution far faster. It is hoped that the use of a single Classifier set in the simulation that is shared amongst multiple agents will mitigate some of the issues surrounding LCS in multi-agent environments.

The Boids model of flocking has received much attention over the years as it has shown a robust starting point from which to develop many more complex and intricate systems. The topic of evolution within such behavioural models is also not new as discussed above. However, the application of LCS' to Boids-like simulations appears to have received far less coverage in the literature.

The basic LCS structure was introduced by JH Holland in a 1975 paper [3], [8], [10], in which he referred to an adaptive cognitive system. This system comprised many of the basic elements of a Classifier system and coined much of the common terminology.

Learning Classifier Systems in general describe the use of evolution and learning to discover a set of competitive rules, called Classifiers, for classification and/or control problems. Classifiers are competitively selected, rewarded and bred with the goal being to produce a set of Classifiers that adequately describes a desired/target behaviour. These Classifiers, in their classic form, are represented as propositional rules taking the form of: If-Condition—Then-Action [8], [10], [14]. Each Classifier also contains some form, or forms, of metric that describe its relative strength when competing with other Classifiers for selection, breeding and other rights. Each time a Classifier is selected by the simulation and its action is performed/executed, the outcome is assessed by some reward mechanism. The Classifier is then rewarded or punished by increasing or decreasing its strength metrics; a mechanism known as the credit-assignment problem.

A LCS employs both evolution and reinforcement learning to discover and then reinforce those rules that provide a desired outcome as determined by the objective function. In addition to these basic processes, most LCSs will also have additional mechanisms for rule discovery. The process of rule discovery allows the system to self-discover unique combinations of rule parameters that may provide all or part of the solution to a given problem. The most basic forms of rule discovery include the covering operator, which produces a Classifier to match an input if one does not yet exist in the Classifier set and the use of Genetic Algorithms to produce new rules through cross breeding of the existing rules.

In this paper, we employ a LCS to evolve the set of rules required to achieve the three Boids' behavioural rules. We systematically investigate the role of the objective function as a measure of a Followers' success at completing a task according to some implicit behaviour encoded in the 'reward' concept. The objective function controls the nature of the reward apportioned to a Classifier through the reward mechanism.

The rest of this paper is organized as follows. In Section II, the methodology is explained in details. The experimental protocol is then discussed in Section III followed by Results, Analysis and Discussions in Sections IV to VI, respectively. Future Work is then discussed, and Conclusions are drawn, in Sections VII and VIII, respectively.

II. Methodology

A. Leader Follower Relationship

The agents were modelled as one of two possible types; either a Leader or a Follower. The behaviour of a leader is pre-scripted to generate behavioural patterns consistent with the experimental protocol. The Leader acts as a goal point for the Follower agent type. Follower behaviour is defined by the selection of a Classifier from the Classifier set at each simulation step. The Follower then applies the action corresponding to the selected Classifier through its effectors. Once the action is completed, information on the action is passed to the objective function which then determines the level of reward, positive or negative, to be applied to the Classifier. Figure 1, shows a basic flow chart of the execution process within the simulation.

B. Reward Functions

A primary goal of this paper was to test the effects of the reward function on the evolution of the system. Testing was conducted for a binary reward, i.e. +2 or -2 corresponding to positive or negative behaviours as determined by the objective function/s. By using such a simple reward strategy, the work in this paper demonstrates if, and how far, one can simplify the complexity of the reward system to generate some desired behaviours.

In nature, evolution does not stop. It is a continuous process representing forces acting on species. Despite this, some behaviour continue to exist. In the field of evolutionary dynamics and evolutionary game theory, such behaviours are known to be evolutionary stable strategies/behaviours. This begs an inquiry into Boids-type behaviours to investigate if such behaviours are indeed evolutionary stable. We investigate this phenomenon experimentally by continuing the evolution after a desired behaviour is found.

C. Simulation Environment

In order to adequately explore the design of the objective function and the emergence of structured behaviour, the development and testing of the simulation was broken down

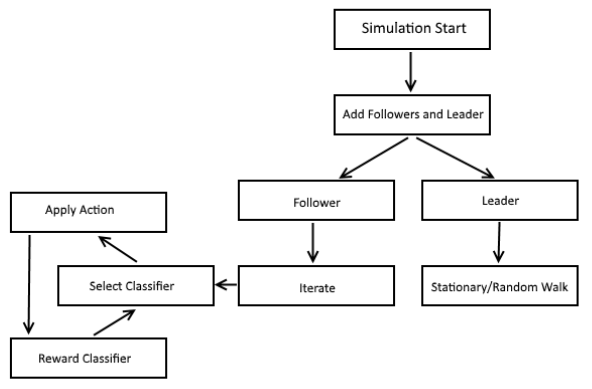

Fig. 1. Representation of Simulation Flow

into multiple phases, with each consecutive phase adding additional functionality to the simulation. The phases were ordered firstly along the lines of the three main attributes previously discussed; these being attraction, alignment and separation. The reward function of the simulation was progressively modified to select for different combinations of these attributes. With the performance of the Followers at meeting these requirements being characterized with each modification of the objective function.

The second stage of development, involved varying the behaviour of the leader once the behaviour of the Followers had been established and characterized. The purpose of this testing was twofold. In the first instance this testing served to characterize how robust the system was to the changing position of the leader, and more specifically how well the reward function handled these variations. In addition to this, a moving leader better approximates the dynamic nature of all agents in the original Boids model.

The simulation was implemented using the Java simulation package called MASON [16], provided under the Academic Free License version 3.0.

D. Simulation Space Abstraction and Simplifications

In order to reduce the number of possible Classifiers in the system to a feasible number for modeling, the simulation space required abstraction to allow for a simplification of the pre-conditions and a reduction in their number. This abstraction reduced the size of the solution space within the simulation as a precondition must logically exist for each of the possible configuration of the simulation space. The total number of possible Classifiers was then total number of pre-conditions, multiplied by the total number of possible actions. This factor was the driving force behind the chosen method of abstraction. For this simulation the space surrounding each of the Followers was broken up into a series of regions centred on the agent, extending to perception boundaries. These regions were generated by dividing the space around each agent first radially into 36 segments providing 10° of arc for each segment. These segments where then further sub-divided by bracketing the distance between the Follower and the limit of its visual range into 20 increments of distance. The 36 segments directly around a given Follower were considered jointly to be the origin sector. This yielded 684 total sectors around each Follower. Figure 2 shows a representation of this space. Decision by the Follower was then made based on how the Leader and other Followers were distributed amongst these sectors.

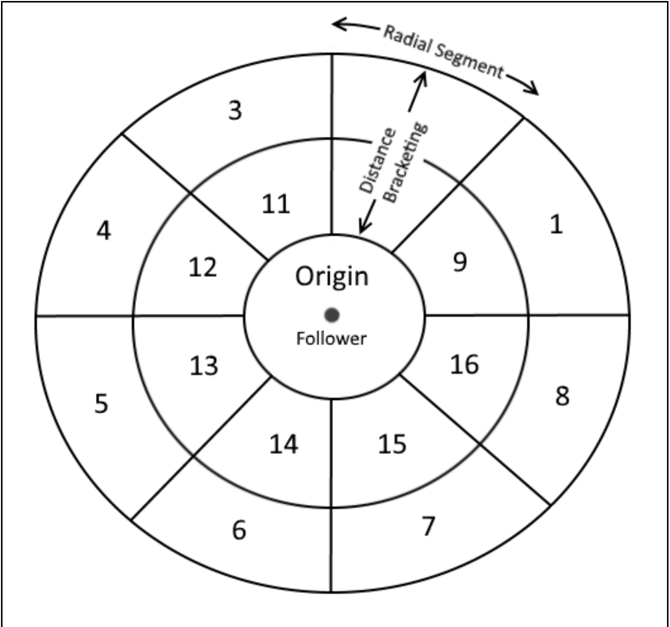

Fig. 2. Representation of Simulation Space Abstraction

E. Metrics for Testing

Several metrics have been defined in order to assess the performance of the objective functions. Each metric correspond to one of the previously identified parameters: attraction, alignment and separation. Each of these metrics was measured with reference to the leader in this paper.

1) Attraction: Attraction was determined by finding the separation between a Follower and the Leader. This value could then be compared to the previous values. If this value had decreased then the system was rewarded. The distance at each step was also recorded with a smaller average distance in this case indicating greater attraction between the Leader and Follower. Distance is measured in pixels, throughout the results section.

2) Alignment: Within the simulation a Follower was said to be in alignment with the leader when the Follower's velocity vector was aligned to a vector drawn between the Follower and the leader within a margin of error equal to half the angle bracketing span i.e. 5.0° (see simulation space abstraction for details of bracketing). Alignment is measured as a percentage of the total simulation time for which this criterion was met. E.g. 3.45% would indicate that for 3.45% of the indicated time interval the Follower was aligned with the leader as specified above.

3) Separation: In order to measure separation, the simulation recorded all instances where-in the Follower violated a predetermined boundary around the Leader. This boundary was set to be one interval of distance as defined in the simulation space abstraction. This interval size was the minimum permissible distance that could reasonably be perceived by a Follower given this abstraction scheme. The number of violations of this space was then averaged over the duration of the simulation to yield a measure of the success of a Follower in maintaining separation away from the Leader. Separation was specified as the total time for which the Follower did not violate the boundary specified above a percentage of the interval indicated. E.g. 99.98% would indicate that the Follower did not approach the leader closer than one interval of distance for 99.98% of simulation time.

TABLE I. REPRESENTATION OF THE CLASSIFIER STRUCTURE USED IN THE SIMULATION

Description	Representation of Classifier Structure					
Sector	1	2	3	...	origin	leader
Min	0.1	-1	0	...	0	27
Max	0.7	0.5	0.2	...	0	unused

F. *Classifier Design and Representation*

The Classifiers used in the simulation where directly modelled to reflect the representation shown in Figure 2. Each Classifier consisted of a 2-Dimentional array of numbers with a span equal to (684+2) and a depth of 2. Table I shows a general representation of the Classifier structure used in this simulation. Each column represents a one to one mapping of the sectors around the Follower; numbered as shown in Figure 2. The origin was then appended at the end of this array. The final element of the array was the location of the Leader relative to the Follower was recorded in the last cell. Each sector, aside from the element containing the leader location, was given a minimum and maximum value corresponding to a total number of other Followers in that sector, normalized over the total Followers population. In addition to a positive number between zero and one, a value of -1 (or #) was included as the ‘don't care’ case for this Classifier structure. Table I shows a representation of the classifier structure.

Each Classifier also contained an action which comprised a direction in which to move and a magnitude/speed at which it should move in the given direction. Thus the effectors of each Follower were its X and Y velocity which could be increased or decreased corresponding to the action of a selected Classifier within a predefined velocity limit.

III. EXPERIMENTAL PROTOCOL

The experiments as a part of this paper were conducted in the following stages.

A. *Establishing a Baseline*

In order to establish a baseline for the performance metrics to be used in later experiments, testing was first conducted to characterize the baseline performance of the system. This was done by measuring the distance, alignment and separation parameters for the Follower when conducting a random walk in the environment for both a static and dynamic Leader.

A. *Alignment*

B. *Progressive Testing of Objective Function*

Once the baseline was established testing was then conducted using a number of objective functions, both individually and in combination to characterize their performance.

C. *Simulation Parameters*

Aside from the parameters of the simulation space abstraction, the key parameters of the simulation are as follows: the simulation was conducted on a 600x600 cell grid. In the case of static testing, the leader was placed at the centre of this grid. One follower and one leader were simulated using a classifier set of size 5000. This set size was based on the number of sectors in the simulation space abstraction (684), plus a generous margin to allow for multiple potential solutions to be maintained and compete simultaneously. Each generation in the simulation spanned 1000 simulation steps with the GA implementing one point crossover with a mutation rate of 0.05. In each simulation step, all classifiers having strength of one, the minimum possible strength were "culled" from the classifier set and replaced with a new random classifier. This random-replacement strategy generally resulted in a more rapid evolution of the system as non-ideal classifiers without the random-replacement strategy took far longer to be removed by replacement from the GA and Covering operator alone.

IV. RESULTS

Listed below are the results from the testing of each of the reward functions, as well as the combined function. Also shown is an example for the output of the combined objective function for a static and moving leader. All graphs represent a fitted function in the measured values. We do not show the original values for two reasons. First, the reward functions come with different magnitude, making it almost impossible to see the trend which visualizing multiple functions on the same graph. Second, the fitted function smooth out the noise from stochastic evaluation of the rewards. Section A shows the curve-fitted measured alignment for each of the tested objective functions. Section B shows the curve-fitted measured attraction for each of the tested objective functions. Section C shows the curve-fitted measured separation for each of the tested objective functions. Lastly section D shows the curve-fitted outputs of the reward function for the combined reward function. The results shown here are for the single follower characterisation of the system.

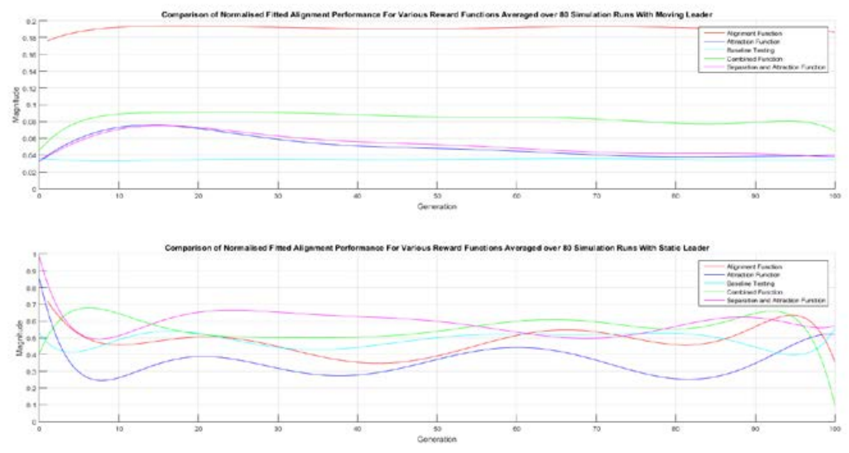

Fig. 3. Changes in Alignment with different reward functions.

B. Attraction

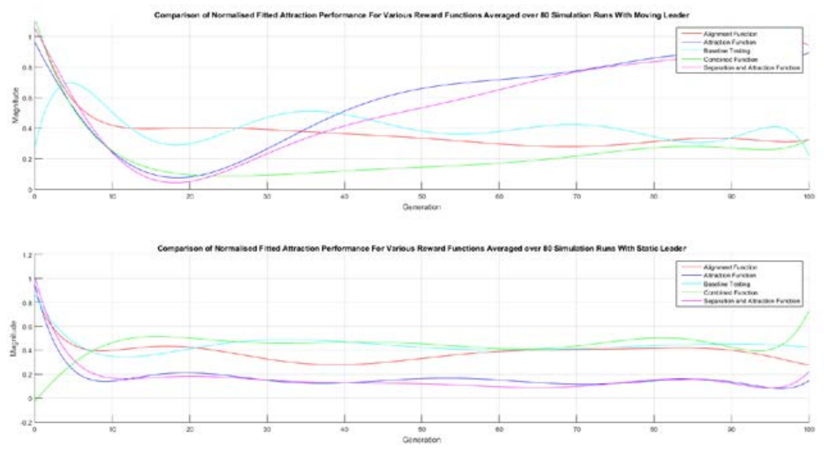

Fig. 4. Changes in attraction with different reward functions.

C. Separation

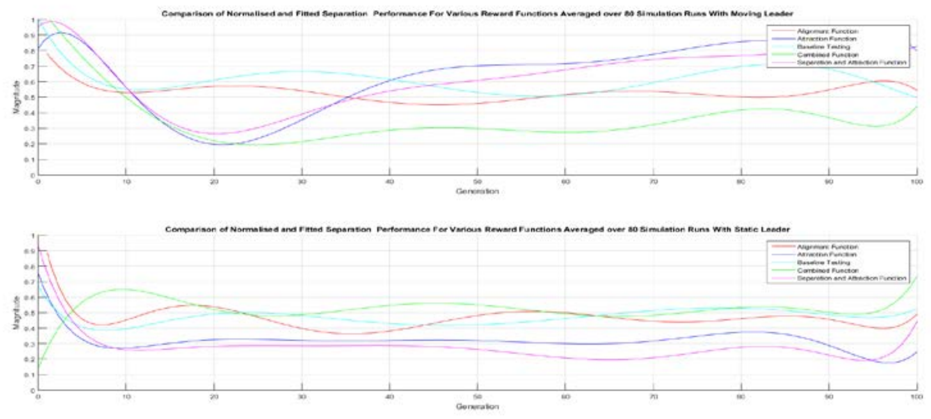

Fig. 5. Changes in separation with different reward functions

D. *Reward Function Output*

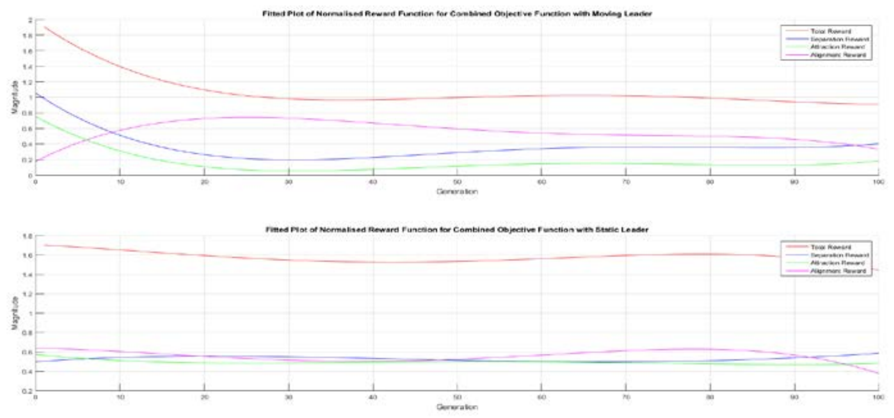

Fig. 6. Changes in separation with different reward functions

V. ANALYSIS OF RESULTS

From analysis of the results above the following information can be drawn. The baseline measurements shown in the figures above indicate that some level of alignment, attraction and separation is intrinsic to the simulation even when run with both Leader and Follower in random walk. This is a byproduct of the random nature of the simulation. Therefore in subsequent result sets, with the application of the LCS, if the performance is no better or perhaps worse than the baseline, then objective function can be said to have failed to meet the design criteria or that some other more complex interaction is effecting the simulation. It is important when observing the figures above that some apply to a stationary leader and others apply to a moving leader, transposing the two may result in confusion.

Figures 3-6 show the results from testing of the objective functions, in the case of separation this testing was combined with the attraction function, as it was believe that it would be meaningless to test the separation function on its own as the baseline measurements indicate that the Follower will rarely violate the boundary of the leader if left to random chance alone.

Figure 3 shows the results of testing for a purely attractive objective function for a moving and static leader. The attractive objective function shows a twofold reduction in the time average distance of the follower from the leader in the moving case when compared to the baseline. In addition to this the combined and separation functions also show a marked performance improvement when compared to the baseline. A similar performance improvement is seen in the static leader case for the attractive and separation function but not so for the combined function. This is likely due to the inclusion of the alignment element within the combined function as the alignment function also shows poor performance in the static case. Ultimately this probably stems from issues with velocity approximation as discussed below

Figure 4 shows the alignment measurements for both a static and moving leader. As can be seen in the moving case, the alignment only function shows a fourfold improvement in the time average alignment when referenced to the baseline. Other reward functions such as the combined reward function show some improvement but only on the order of approximately to 2 times at most. This clearly shows the performance of the alignment function in improving the time average alignment. This performance is carried through to the static test case as shown in the figure. Also of note is that the performance of the combined function is markedly better for the static case than for the moving case. This shows that the other elements of the combined function do not interfere to much in the performance of the combine function in this case.

Figure 5 shows the results for the time average separation of the follower from the leader in the moving and static leader case. The results for separation are somewhat less conclusive in the moving case, though a reduction in separation can be observed for the cases of the attractive reward functions (Combined Function, attraction Function and separation function). In the static case a clear change in the level of leader/follower separation can be seen based on the reward function chosen.

Figure 6 shows a representation of the output of the reward functions for the system for the combined functions in the static and moving leader case. As can be seen in both figures the total reward shown in purple remains relatively unchanged whilst the rewards for Attraction, Alignment and Separation might vary. Due to the scale required to show each function though this can be somewhat difficult to pick out.

VI. Discussion

From the results it can be concluded that the Classifier system has improved the ability of the follower to flock around the leader. When the individual reward functions were combined to fully replicate Reynolds the performance as measured by the metrics was observed to be favorable. Additional empirical observations of the simulation show that the visual characteristics of the simulation resemble those seen in Reynolds Boids model.

As can be seen throughout the results there tends to be variation between the result shown for the static leader and those shown for a moving leader. This can be attributed to multiple factors but in particular this is seen as a result of the classifier structure not containing any means by which the follower might represent the velocity of the leader or other followers. This means that the there will in general be an error between the velocity of the follower and the leader. This error is exacerbated in the static case due to the inability of the follower to have zero velocity. The result of this is an essentially infinite error between the velocity of the follower and the leader when the leader is static. In the moving case this is less of an issue as the leader will always have some velocity but this error is none the less present in most cases. The velocity errors are most clearly observable when a comparison is made between the separation results for the moving and static leader. As can be seen in general for a given function type the separation in the static case is worse than that for the moving case. In the visual representation this manifested itself as a propensity for the follower to overshoot the static leader and then dart back again overshooting. This had a noticeable impact on the separation measurements as discussed.

Further analysis of the results from the testing of the separation function shows a mixed outcome. In the moving leader case the separation is defiantly effected by the application of the reward functions but sees comparable performance between the separation function and the other functions. The results for the static case are a bit more confusing. It can be seen that the separation function actually performs worse in the separation criteria then other function not selecting for separation. It is hypothesized that this is an effect of the reward function design in this case. In particular for the attraction function there is a tendency for overshoot due to the inability of the follower to approximate the leader speed. It is believed that this overshoot means that the follower actually spend very little time at a proximity that would result in punishment. The separation function however slows the follower and forces the follower to reverse direction when it enters the separation region around the leader. This process actually the takes longer than simply "shooting" through the separation region. This means that the follower ends up receiving more negative reward with the application of the separation function, but also worse separation performance as seen.

With regards to the output of the combined reward functions shown in Figure 6, it can be seen that the total reward remains relatively constant as the other values vary. Interestingly the attraction output value is very strongly affected by the chosen design of the attraction element of the objective function. This is also what gives the total reward its apparently low value even though the simulation can be observed to be performing quite well. Essentially this is due to the fact that the attraction objective function reward only rewards based on whether the follower has moved closer to the leader, so once the follower is at a stable distance the reward will tend to zero for attraction. Some windowing of the reward function for attraction can reduce this effect and raise the average reward to just over two on the scale shown. Figure 9 shows the results of the reward function for a static leader. Interestingly the output of each of the reward function elements is very similar to that of the moving case, but there appears to be much less improvement over time. This is mirrored to some extent in the other graphs for the static leader case. It is believed that this is a result of the static leader case being a simpler task for the classifier system to learn resulting in the system achieving a maximum value rapidly. Any improvement would then likely be masked within the average of the first few generations and thus not be explicitly visible.

Throughout the testing it was seen that the resulting behaviours was also evolutionarily stable as the behaviour once evolved was maintained until the end of the simulations.

In addition to the work conducted to characterize the simulation with a single follower, multi follower simulations were also conducted to test the scalability of the system. This testing showed that the behaviors observed in the single agent testing carried over to the multi agent simulations.

VII. Future Work

Possible future work on this topic could include integrating a representation of the current speed and heading of the agents within the environment into the Classifier structure. This would then allow for better velocity matching between the Follower and the Leader as well as other Followers. Although, it has been noted that the current Classifier structure is not suitable for this modification as it cannot distinguish between individual followers in the simulation. Achieving this would require either a new Classifier structure or a different representation of the velocity of the followers in the simulation. It is therefore suggested that the total velocity and heading of the agents in each sector could be calculated as a group velocity and added to the classifier structure as a third row in future simulation. This would likely reduce overshooting and instability as seen in this simplified model. Additional objective functions could also be designed to test the effect of various weightings of both heading matching and velocity matching. In terms of gauging the quality of flocking resulting from such simulation, future work could include a metric specifically oriented towards gauging cohesion of a group of agents in this environment, such as those mentioned in [11].

VIII. Conclusion

This report has explored the application of a LCS to the development of Boids like rules in simulation. The simulation had both successes in replicating the individual parameters of the simulation, such as alignment as well as demonstrating that a combination of the three objective functions also performs analogously to the Boids model. The results were limited by the inability of the Followers to adequately match velocity with that of the leader using alignment, attraction and separation metrics alone. Additionally while limited testing was done with multiple agents in the environment, further work is needed to adequately explore the effects of multiple agents on the evolution of rule set, particularly given the issues often experienced with LCS' in multi agent environment. As such further work is needed on this topic to refine the model.

References

[1] C. Reynolds, “Flocks, Herds and Schools: A Distributed Behavioural Model,” *Computer Graphics*, vol. 21, no. 4, pp.25-34, 1987.

[2] B. Benes and C. Hartman, “Autonomous Boids,” *Computer Animation and Virtual Worlds*, vol. 17, no. 3-4, pp. 199-206, 2006.

[3] J. Holland, "An Adaptive Cognative System," *Progress in Theoretical Biology,* vol 4, no 1, pp.279-293, 1976

[4] John H. Holland, "Genetic Algorithms," University of Columbia, New York City, Informative 01.04.2003 10:02 Uhr, 2003

[5] T. Kovacs, M. Kerber, "What Makes a Problem Hard for XCS?," *Lecture Nnotes in Artificial Intelligence,* vol. 1996, pp. 80-99, 1996

[6] R. Richards, "Classifier Systems & Genetic Algorithms," in *Zeroth-Order Shape Optimization Utilizing a Learning Classifier System.* Stanford, United States of America: Stanford Univercity, 1996, ch. 3

[7] K. Takadama, T. Terano, and S. Katsunori, "Learning Classifier Systems Meet Multiagent Environments," *Lecture Notes in Artificial Intelligence*, vol. 1996, pp. 192-210, 1996

[8] R. Urbanowicz, J.. Moore, "Learning Classifier Systems: A Complete Introduction, Review, and Roadmap," *Journal of Artificial Evolution and Applications*, pp. 1-25, 2009

[9] L. Bull, "On ZCS in Multi-agent Environments," *PPSN V Proceedings of the 5th International Conference on Parallel Problem Solving from Nature,* vol. 1498, pp. 471-480, 1998

[10] L. Bull, "A Brief History of Learning Classifier Systems: From CS-1 to XCS and its Varients," *Evolutionary Intelligence,* vol. 8, no. 2, pp. 55-70, 2015

[11] J. Harvey, K. Merrick and H. Abbass, “Application of Chaos Measures to a Simplified Boids Flocking Model,” *Swarm Intellegence,* vol. 9, no. 1, pp. 23-41, 2013

[12] Z. Zatuchna and A. Bagnall, “A learning Classifier system for mazes with aliasing clones,” *Natural Computing,* vol. 8, no. 1, pp. 57-99, 2009

[13] E. Ferrante, A. E. Tugut, D.-G. Edgar, D. Marco and T. Wenseleers, “Evolution of Self-Organized task Specialization in Robot Swarms,” *PLoS Computaional Biology,* vol. 11, no. 8, pp. 1-21, 2015

[14] J. H. Holland, L. B. Booker, M. Colombetti, M. Dorigo, D. E. Goldberg, S. Forrest, R. L. Riolo, R. E. Smith, P. L. Lanzi, W. Stolzmann and S. W. Wilson, “What is a Learning Classifier System,” *Lecture Notes in Computer Science,* vol. 1813, pp. 3-32, 2000

[15] J. H. Holland, “Studying Complex Adaptive Systems,” *Journal of Systems Science and Complexity,* vol. 19, pp. 1-8, 2006

[16] S. Luke, G.C. Balan, L. Panait, C. Cioffi-Revilla, and S. Paus, “MASON: A Java multi-agent simulation library,” In Proceedings of Agent 2003 Conference on Challenges in Social Simulation, vol. 9, p. 9. 2003.