Evolutionary Recovery from Sensor Failure in a Trained Multi-objective Swarm

Jon H. Roach, Robert J. Marks II, and Benjamin B. Thompson

***Abstract*— One of the benefits of using Combs disjunctive control in swarm intelligence is the ability of the swarm to continue to operate even when one or more of its sensors are broken. Instead of failing, the swarm continues to perform based on the inputs of the remaining sensors. In this project, a multi-objective scenario is designed for a swarm of agents to complete. Once optimized, the swarm is then modified by removing one of its sensors. Maintaining the original objective, the evolution process is continued to allow the swarm to compensate for the sensor failure. This paper describes the different emergent behaviors of the swarm after the post failure rules are optimized using an evolutionary learning algorithm.**

***Index Terms*—Keywords: swarm intelligence, multi-state, task switching, fuzzy control, emergent behavior, Combs control**

I. Introduction

Swarms in nature are composed of a large number of individual agents, such as bees or ants, that each follows a set of basic rules [3]. Often these rules can be described as a simple cause effect relationship between an external stimuli and an agent's response. Each agent's response contributes to the emergent behavior of the swarm, which can often be unpredictable [2][5][8]. However, what if agents were unable to sense a certain stimulus? For example, what if a group of drones was unable to sense the difference between friend and foe? Or what if they had difficulty sensing certain types of enemy units altogether? How would that reduction in awareness affect the emergent behavior of the swarm?

The robustness of disjunctive control, also called Combs control [9][10][11], has the advantage of seamlessly recovering from failed sensors by exploiting possibly redundant information available from other sensors when available [12][13][14]. A simple example is steering a car to the right. This can be done by a) turning the front wheels of the car to the right, b) turning the back wheels of the car to the left, c) braking the two wheels on the right side of the car, or d) accelerating the rotation of the two wheels on the left side of the car. As long as one of these control actions exists, the car can be steered to the right even when the remaining three functions fail. Redundancy in the information available in sensors is often not as obvious as in this example. In the case of swarms, the failing of one or more sensors may prompt the swarm to adopt a different emergent strategy in order to meet the objective of the collective.

Our goal is to discover what new emergent behaviors occur when certain rules are removed by disabling some of the agent's sensors. A previously tested simulation is used as a baseline for this experiment [1]. In this paper, we demonstrate the emergent behaviors that result from individually removing four of the agents' sensors.

Although we explain swarm behavior in detail, there is no substitute for watching swarming in real-time. The motivated reader is therefore encouraged to view our video which both explains and displays the fascinating and emergent behavior described in this paper .

II. Swarm Intelligence

In the simulation, a swarm is tasked with completing two objectives: guarding a central base from enemy attacks while also searching out and destroying enemy units. In order to encourage recruitment, at least three agents are required to successfully kill an enemy unit. The simulation ends when the base sustains a set amount of damage: in this case, when ten enemy projectiles hit the base. The movement of the agents is controlled by sensors connected to weighting functions. When an agent senses an object, the distance to that object is fed into a function that tells the agent how to move with respect to the object. In addition to a series of object sensors and their corresponding weighting functions, agents are also equipped with a center sensor that tracks how far away the agents are away from the base and pulls them back in if necessary. All of the sensors have limited range, except for this center sensor.

In order to accomplish both objectives, the agents are allowed to take on one of multiple states: defender, scout or recruiter. For each state, there is a different set of sensor weighting functions, including the center sensor functions. When an object switches states, it is changing the set of rules it follows. Since the swarm needs to dynamically adjust its resources, the agents are able to switch between states using threshold functions. If an object senses there are too many agents working on the same task, or too few, the agents are able to change states or request other agents to change states as necessary. How the agents make this decision is determined by a threshold value. Each weighting function and threshold is an adjustable parameter that represents a rule that the agents follow. These rules are optimized through the use of an evolutionary learning algorithm. This paper discusses what happens when one of these rules is removed from the swarm's decision making.

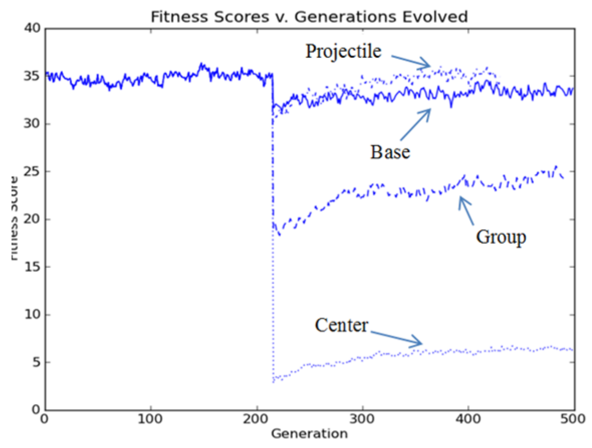

Figure 1. The results of the evolution before and after breaking the four sensors are shown here. Before generation 200, the swarm is evolving with all of its sensors. At the discontinuity, the graph splits to represent the fitness scores after the removal of the sensors. The swarm performs well with its projectile and base sensors removed and okay without its group sensor, but very poorly when the center sensor is disabled.

III. Evolution Process

A predatory swarm's performance can be maximized using an evolutionary learning algorithm [6][7][9][11][12][13]. Performance is measured by tracking the number of enemy kills and the swarms' fitness scores. Each swarm is assigned a fitness score which is equal to the time survived, multiplied by the percentage of surviving agents within the field of play, multiplied by an efficiency factor that represents the reduction of the distance travelled by the swarm, as shown in (1). E is the efficiency factor and C is a constant.

$$F = Time * P_{agents} * E * C \qquad (1)$$

An initial random population of teams, each represented by a set of weighting functions and thresholds, is evolved by simulating the scenario with each member of the population and comparing the results. Teams with higher kill totals and fitness scores survive and advance to the next generation of evolution, while the poorer teams are removed. Each surviving team is copied and mutated by adding random Gaussian noise to the weights and thresholds. For weights with values ranging from negative five to five, the standard deviation of the Gaussian noise added is approximately 1.25. This algorithm allows the population of swarms to "learn" which rules and strategies successfully accomplished both objectives of attacking and defending [4][10]. This process is repeated for approximately 800 generations, resulting in a set of weights that were optimized for the given scenario. A brief description of the emergent behaviors is given below.

To accomplish the first objective of defense, defenders learn to loosely circle the base and intercept enemy projectiles by moving between them and the base. If the number of defenders drops too low, nearby scouts switch tasks to help defend the base. For the second objective, scouts learn to spread out from the base and from each other while looking for enemy units. When an agent finds an enemy, it transforms into a recruiter and return to base. At the base and along the way, scouts join the recruiters until the recruiter senses that the group is strong enough to destroy the enemy unit. The recruiter leads the group back to the enemy in order to eliminate the enemy. Any surviving agents in the group then go back to exploring the map for enemies.

Now that we have a solution for the multi-task scenario, our next step is to begin disabling sensors one at a time in order to determine how the failure of one of the sensors will impact the emergent behavior of the swarms. The first sensor that is removed is the projectile sensor. This sensor is used by defending agents to see incoming enemy projectiles that are headed towards to friendly base. Defenders are normally able to intercept the projectiles. We want to see what would happen if the defenders are no longer able to sense these projectiles. The population of swarms that were evolved with all of their sensors intact is used as the initial population for a second round of evolution without this projectile sensor.

This process is then repeated for another scenario. The projectile sensor is turned back on and the group sensor is removed. The group sensor allows the agents to track how many other agents are nearby. This is used in recruitment since it allows the recruiting agents to know if they have enough friendly agents around them to destroy an enemy unit. The evolving population is reset to the initial group of teams that were evolved with all their sensors and the evolutionary algorithm is run again. This is repeated two more times: once with the center sensor broken and again without the base sensor.

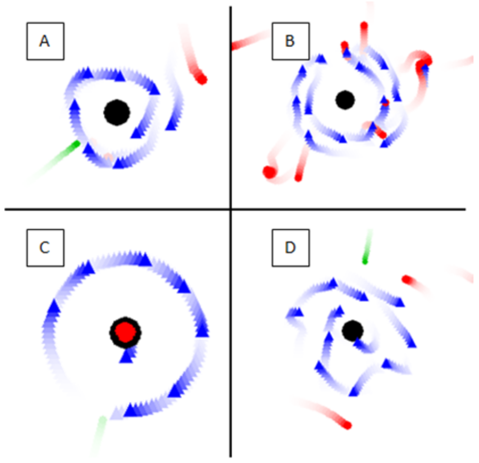

Figure 2. This figure demonstrates the emergent behavior of defending agents evolved with different sensors disabled. (A) shows the tight ring formed when the projectile sensor is removed. (B) shows attackers forming groups before scouting due to the fact that the group sensor is off. (C) shows the ring formed around a group of replacement scouting agents when the center sensor is disabled. (D) shows a mob of defenders surrounding the base with the base sensor broken.

IV. Results

Fitness scores for each of these four evolutionary cycles are shown in Figure 1. In each case, the fitness scores are lower with the sensor removed. However, these scores do improve over time as the swarms' weights are adjusted to maximize the use of the remaining sensors the swarms did have. Even though it is missing one of its sensors, in some cases a swarm is still able to achieve scores almost as high as a swarm with all sensors available. One of the main differences in emergent behaviors that developed is how the defending agents guarded the base. These strategies are depicted in Figure 2.

Combs control subjects each sensor to a nonlinear actuator who shapes are determined through the evolutionary process. Examples of actuator nonlinearities are illustrated Figure 3. The actuator outputs are aggregated to determine agent action [8].

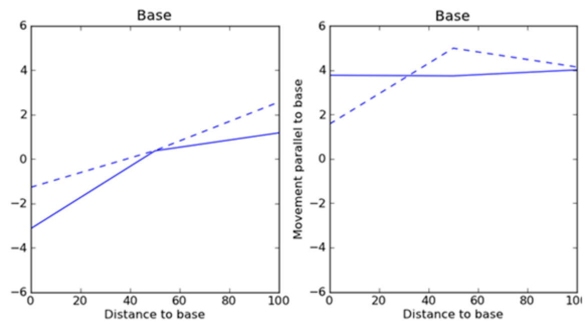

Figure 3. Without the projectile sensor, the swarms adapt by tightening the ring around the base and speeding up their rotations. In the first graph, the old strategy is represented by the solid line and the new function is the dotted line. Initially, the "sweet spot" that the defenders converged to was around 50. This is lowered by evolutionary adaptation to a radius of 40 by the new method. In graph 2, the speed at the "sweet spot" (40) from a little under 4 to almost 5.

A. Projectile Sensor

Removing the projectile sensor does not greatly affect the fitness scores, which drop slightly compared to the results of the previous evolution. This is because the swarms are able to adapt to the loss of the sensor and still accomplish their objectives similarly to before. With the sensor, defending agents circling the base are attracted to projectiles and move in for the kill. Without the sensor, defenders are unable to see the projectiles. The swarm's solution is to make its circling behavior tighter and faster. By rapidly circling around the base, agents are able to intercept most enemy projectiles simply by running into them. The tighter circle uses a smaller group of defenders to effectively defend the base and allows more scouts to look for enemy units. This is an example of a behavior that existed previously, but was adjusted in order to compensate for the loss of a sensor. The behavior of the scouts and recruiters remains the same, since their task does not involve defending the base from projectiles.

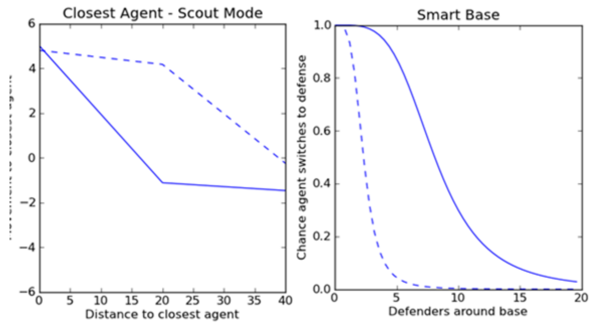

Figure 4. Since the group sensor is disabled, the swarms have to adapt. Originally, scouts are repelled from each other. The value of the solid line is positive (indicating attraction) at small distances. However, since the agents are repelled at larger distances, the agents should never get close enough to each other for that to matter. The new function causes agents to be more attracted to each other. At large distances the attraction is weak, but still positive, and increases as the agents move closer. The second graph shows how many defenders the base is expecting. The swarm learns to use a smaller number of defenders in order to allow more agents to explore.

B. Group Sensor

The group sensor is proven to have more of an effect on the swarms' behavior. This sensor is crucial for the recruitment techniques developed previously since it allows recruiters to count the number of nearby agents. When this ability is taken away, the agents are no longer able to determine if their group is large enough to destroy an enemy. Through the evolutionary process, the swarms respond by modifying their strategy. Instead of scouts spreading out, over time the scouts form up in clumps and scout in groups. This removes the need for recruiters to return to the base to find other agents. When an enemy is found by a group, recruitment is no longer needed. Instead, assuming a group of at least three agents, the group can simply move in immediately and destroy the enemy. The defensive strategy remains the same, but with a smaller number of defenders to allow more scouts to explore. Compared to the original evolution, scores drop since the effectiveness of the search is reduced when the scouts search in groups. However, considering the previous recruitment strategy no longer works at all, this emergent behavior is a reasonable alternative and allows the swarm to still find and destroy enemy units while maintaining the previous defensive strategy of surrounding the base in a rotating ring.

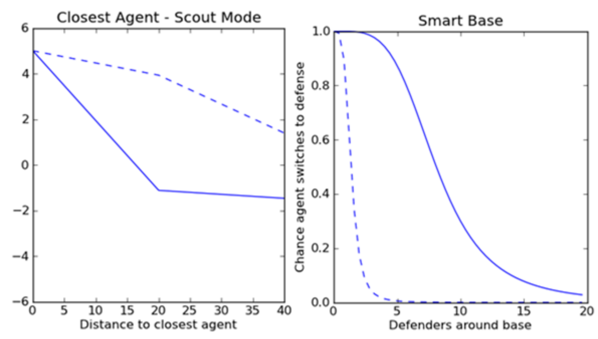

Figure 5. These graphs look similar to those in Figure 4. However, there are some significant differences. On the left, the new strategy developed without the center sensor is represented by the dotted line. Scouting agents are always attracted to each other since the function is positive for all distances, including an attraction of 1.5 at the maximum distance of 40. This is a much stronger attraction than that in Figure 4. Also, the second graph shows how the swarms have learned to use fewer defenders, even less than when the group sensor is removed.

C. Center Sensor

When the center sensor is removed, the swarm scores drop off drastically. The center sensor is of foundational importance for the swarm mission. Without it, there is nothing to prevent exploring agents from wandering off the edge of the map. Since the swarm is unable to perform well in its task of finding and destroying enemies, it focuses on the second task of defending its base. The unexpected emergent behavior that develops from evolving without this sensor is the clumping of scouts in the base. When the simulation begins, some of the scouts form groups and begin searching, but most of the scouts form one large group and converge right on top of the base. These scouts, while not scouting, are used as reserves to replace destroyed defenders. The defending ring is similar to before, but with a much larger radius and a group of eight evenly spaced defenders surrounds the base. As the defenders detonate to take out incoming projectiles, they are replaced by scouts in the base. The defenders are far enough away from other defenders and the base (which is full of scouts) so that detonations do not take out any friendly units. This strategy allows a large portion of the swarm to stay defending the base which keeps the base alive for as long as possible. While the swarm is unable to properly complete both objectives, this modification does allow the swarm to perform well in at least one mission objective.

D. Base Sensor

Without the base sensor, the swarms are able to still achieve fitness scores on par with previous results. Since the defenders are unable to sense the base, the previous rotating ring strategy is impossible. To compensate for this loss, the defenders forms a mob and surrounded the base. The defenders also learn to spread out from each other instead of linking up in the circling method. Previously, defenders used the attraction from the base sensor to stay close to the base. After evolution, the swarm learns to use the center sensor to accomplish the same objective by pulling in defenders that wander off more than 50 units away. This group of randomly moving defenders makes it difficult for enemy projectiles to reach the base, even though it is not as efficient as the ring strategy. The scouting and recruiting methods are not affected by the loss of the base sensor, so overall fitness scores remain high.

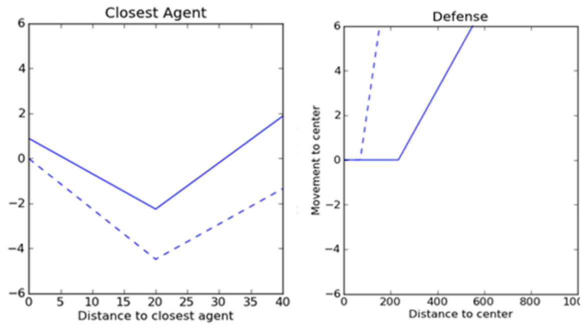

Figure 6. These graphs represent some of the changes in rules that develop when the base sensor is turned off. The first weighting function shows how defenders react with respect to other agents. Initially, defenders are attracted to the "sweet spot" at the zero-crossing of the graph: around 30 units away. After evolving without the base sensor, the defenders learn to spread out and are always repelled based on the negative value of the function. The second graph shows the changes in the center sensor. Initially the sensor pulls back defenders that get more the 250 units away from the base. After evolution, the defenders are kept in a tighter radius of closer to 50.

V. Conclusion

We have shown that while decreasing a trained swarm's abilities will negatively affect its performance, an evolutionary algorithm can allow the swarm to learn a new strategy that utilizes its remaining strengths. In four separate cases, a swarm is evolved to complete a multi-objective scenario, each time with the loss of one sensor. Sometimes an emergent behavior is refined to maintain swarm performance. An example of this is when the rotating defenders form a smaller, tighter circle around the base to better defend it when their projectile sensors were removed. Other times, a completely different strategy emerges, such as when scouts began exploring in groups to remove any need for recruitment. Also, if a swarm is unable to accomplish one objective due to sensor loss, then it would put more of its focus into the other objective. For instance, disabling the center sensor greatly reduces the swarm's ability to search for enemies so instead the swarm began to keep a large percentage of its agents near the base in order to survive for as long as possible. In each case, even though the fitness scores drop initially, the swarms are able to use other sensors in order to at least partially compensate for the sensor reduction.

Acknowledgment

Special thanks to Baylor University, the Applied Research Laboratory at the Pennsylvania State University and especially the Office of Naval Research's University Laboratory Initiative for funding for this project.

References

[1] Jon Roach, R.J. Marks II & Benjamin B. Thompson, "Tactical Task Allocation and Resource Management in Nonstationary Swarm Dynamics," to be published

[2] Jon Roach, Winston Ewert, Robert J. Marks II and Benjamin B. Thompson, "Unexpected Emergent Behaviors From Elementary Swarms," Proceedings of the 2013 IEEE 45th Southeastern

Symposium on Systems Theory (SSST), Baylor University, March 11, 2013

[3] E. Bonabeau et al, Swarm Intelligence: From Natural to Artificial Systems. Oxford, NY: Oxford University Press, 1999.

[4] D. Fogel, Blondie24. San Francisco, CA: Morgan Kaufmann Publishers, 2002.

[5] I. Gravagne and R. Marks II, “Emergent Behaviors of Protector, Refugee, and Aggressor Swarms,”IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, vol. 37, no. 2, pp.471-476, Apr, 2007.

[6] Z. Yuan, “Continuous Optimization algorithms for tuning real and integer parameters of swarm intelligence algorithms,” ANTS 2010, pp. 203-214, 2010.

[7] M. Clerc and J. Kennedy, “The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 1, pp. 58-73, Feb, 2002.

[8] W. Ewert, R.J. Marks II, B.B. Thompson & Albert Yu, “Evolutionary Inversion of Swarm Emergence Using Disjunctive Combs Control,” IEEE Transactions on Systems, Man & Cybernetics, (prepint available at IEEE Xplore February 1, 2013.)

[9] William E. Combs. Reconfiguring the fuzzy rule matrix for large time-critical applications. in 3rd Annu. Int. Conf. Fuzzy-Neural Applicat.,Syst., Tools, Nashua, NH, Nov. 1995, pp. 18:118:7.1

[10] William E. Combs and J. E. Andrews. Combinatorial rule explosion eliminated by a fuzzy rule configuration. IEEE Transactions on Fuzzy Systems, vol. 6, no. 1, pp. 1-11, Feb. 1998.

[11] Jeffrey J. Weinschenk, William E. Combs, Robert J. Marks II, "On the avoidance of rule explosion in fuzzy inference engines," International Journal of Information Technology and Intelligent Computing, vol.1, #4 (2007).

[12] Sreeram Narayanan, R.J. Marks II , John L. Vian, J.J. Choi, M.A. El-Sharkawi & Benjamin B. Thompson, "Set Constraint Discovery: Missing Sensor Data Restoration Using Auto-Associative Regression Machines," Proceedings of the 2002 International Joint Conference on Neural Networks, 2002 IEEE World Congress on Computational Intelligence, May12-17, 2002, Honolulu, pp. 2872-2877.

[13] Benjamin B. Thompson, Robert J. Marks II, and Mohamed A. El-Sharkawi "On the Contractive Nature of Autoencoders: Application to Missing Sensor Restoration," 2003 International Joint Conference on Neural Networks, July 20-24, 2003 , Portland , Oregon (pp. 3011-3016)

[14] M.A. El-Sharkawi and R.J. Marks II, "Missing sensor restoration for system control and diagnosis," Sympoium on Dyagnostics for Electric Machines, Power Electronics and Drives, Atlanta, GA 24-26 August 2003, pp. 338-341.

[15] D. Cvetkovic and I. Parmee, “Evolutionary Design and Multi-ojbective Optimisation,” Plymouth Engineering Design Centre, University of Plymouth. Drake Circus, Plymouth PL4 8AA, U.K.

[16] K. Liang et al, “Dynamic Control of Adaptive Parameters in Evolutionary Programming,” Computational Intelligence Group, School of Computer Science. University College, The University of New South Wales. Australian Defence Force Academy, Canberra. ACT, Australia 2600.

[17] C. Fonseca and P. Fleming, “An Overview of Evolutionary Algorithms in Multiobjective Optimization,” Dept. Automatic Control and Systems Eng. University of Sheffield, Sheffield S1 4DU. U.K. July, 1994.

[18] F. Kursawe, “A Variant of Evolution Strategies for Vector Optimization,” University of Dortmund, Department of Computer Science XI, D 44221 Dortmund, Germany.

[19] S. Carlson, “A General Method for Handling Constraints in Genetic Algorithms,” University of Virginia, Charlottesville, VA.