DOI: http://dx.doi.org/10.7551/978-0-262-32621-6-ch145

Feedback Control of Evolving Swarms

Jacob Gold, Adam Wang, Kyle I Harrington
[1]DEMO Lab, Brandeis University, Waltham, MA 02453
kyleh@cs.brandeis.edu

Abstract

We examine the effects of proportional-integral control on the fitness and genetics of an evolving swarm. Introducing controllers with a set point designed to distribute birds equally across the food in a given simulation increases the rate at which the birds accumulate energy, but also increases the rate at which they expire. We find that the amount of food the birds gather is not dependent on the force of the feedback controllers but on the quality of information they transmit. The trends we observe can help to understand and improve the results of a wide variety of systems exhibiting swarm dynamics.

Introduction

Swarming and flocking behavior is ubiquitous throughout biological and physical systems of all scales. Bird flocks, schools of fish, bacteria (Munoz et al., 2007), and even chemical reactions (Sayama, 2011) are prime examples of non-equilibrium dynamical systems. Simulations to describe these phenomena were first developed by (Reynolds, 1987). Remarkably, the overall motion is governed by relatively simple rules detailing behavior of individuals and their interactions with neighbors. Motivated by both the desire to achieve greater biological control of evolving ecosystems (Holt and Hochberg, 1997; Roderick and Navajas, 2003) and recent advances in swarm robotics (Rubenstein et al., 2012), we introduce an environmental control mechanism and explore the evolutionary consequences of environmental feedback control.

In this paper, we explore the interaction between an evolving swarm and an environmental feedback controller. We consider a flock of birds with localized energy sources that provide birds with the energy necessary for their survival. In real-life systems the behaviors of agents evolve over successive generations in order to favor the most successful. Natural selection is accounted for in our model by an energy system where birds expire and are replaced by offspring of the remaining population.

We introduce control into our system in the form of local proportional-integral-derivative (PID) controllers that are capable of attracting and repelling birds. These PID controllers drive the system towards a particular state defined by the PID controller, allowing the system to be optimized. Optimization of heterogeneous swarms has several applications, such as commercial pollination with flying agents (Berman et al., 2011b,a), electric power systems (Fukuyama et al., 1999), and general optimization problems (Eberhart and Kennedy, 1995). Furthermore, recent advances in robotics have made it possible to experiment with large-scale physical swarms of robots (Rubenstein et al., 2012), allowing novel swarm control techniques to be easily tested.

The Evolution of Swarms

Since the original proposal of the Boids algorithm (Reynolds, 1987), the collective dynamics of swarms has been a common topic of study in artificial life. While many studies focus on the effects of swarming algorithms on collective dynamics, we focus specifically on the evolution of swarms. In (Spector and Klein, 2002), the evolution of simple swarming parameters is used in conjunction with a visual simulator to study emergent dynamics. This study was later extended to a radical degree with endogenously evolving computer program controllers for agents in the evolving swarm (Spector et al., 2005). Other work utilizing evolutionary algorithms in conjunction with swarming behavior has mostly been pursued in the context of particle swarm optimization, such as (Zhang and Xie, 2003). We develop our model based upon (Spector and Klein, 2002), to maintain an experimentally tractable degree of complexity.

Model

Simulation is performed in 3D with the Brevis simulator (Harrington, 2014), a scientific and artificial life simulator. Brevis provides simulation and visualization capabilities via the Java JVM and the programming language, Clojure. One particular feature of importance in swarming simulations is neighborhood detection. Brevis provides a nearest neighbor algorithm, allowing for fast lookups of operations commonly used in swarm algorithms.

The core features of the simulation are a population of

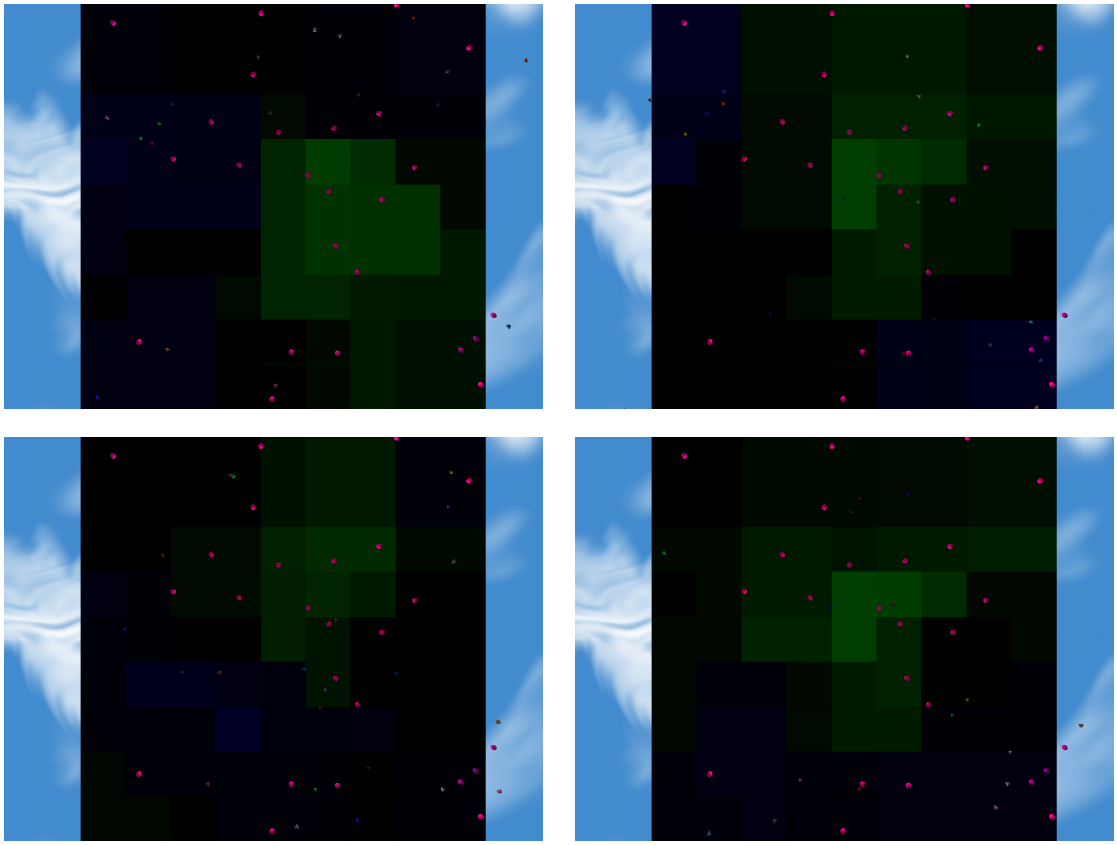

Figure 1: Example of a feedback-controlled evolving swarm and environment simulated with Brevis (Harrington, 2014). Green regions are attractive, blue are repulsive. Pink dots are food, colored cones are birds.

evolving birds, a set of energy sources, and a feedback controller (discussed later). The birds in our simulation use a simple algorithm to update their accelerations. For each bird, we consider the first bird and the first food in its list of neighbors, if any exist within the simulation-defined neighborhood radius (200 units, in this case). For both the neighboring bird and food, the direction vector from the bird to the neighboring object is multiplied by a constant, depending on whether that neighboring object is considered close to or far from the bird. This vector will be 0 if no neighbor of that type exists. The acceleration of the bird is updated to be the sum of these two vectors and passed to Brevis to update their position and velocity.

Energy

Each bird and food has an energy associated with it. A bird's energy will decrease at a constant rate (0.25 units/time), while a food's energy will increase at a constant rate (0.1 units/time). The change in energy in each iteration is proportional to the time step of that iteration. In these simulations we use a fixed time step of 1. If a bird collides with a food it gains energy at a rate of 0.005 units/time, and the food loses energy at a rate of 0.005 units/time. Also, if two birds collide with each other, they will both lose 0.001 units of energy. When a food reaches zero energy, it will be removed from the simulation and a new food will be created at a random position. When a bird reaches zero energy, it will die and be removed from the simulation, and a new mutant bird will be created at a random position to replace it.

Evolutionary Algorithm

A genome is associated with every bird, made up of the information used to update their accelerations. Each bird has a distance associated with food and a distance associated with other birds. If a neighbor is closer than the corresponding distance, it is considered close; otherwise, it is considered far. Each bird also has four coefficients for determining acceleration with respect to neighboring objects: close food, far food, close birds, and far birds. The distance genes range over the nonnegative numbers, while the coefficients can be positive or negative. These genes are all listed below.

Genes

- neighborC: Neighbor coefficient for close behavior
- neighborF: Neighbor coefficient for far behavior

- neighborD: Neighbor distance
- foodC: Food coefficient for close behavior
- foodF: Food coefficient for far behavior
- foodD: Food distance

Every time a bird runs out of energy, its replacement will obtain a mutated copy of a genome of a bird still alive in the simulation. The mutation consists of multiplying the existing value by a random number picked from an even distribution between 0.5 and 1.5. This serves as the fitness evaluation of our simulation. Birds which are fit are able to keep themselves alive longer, and by doing so they are more likely to have their genes copied when a new bird is created.

Feedback Control

Feedback control serves a variety of purposes in engineered systems, as it allows for real-time correction. Many algorithms exist which allow the feedback controller to calculate or learn what control parameter will move the system towards the set point. We adopt a proportional-integral-derivative (PID) algorithm as our feedback controller (Astrom and Hagglund, 1995; Astrom and Murray, 2008).

Many biological systems naturally self-organize, and often utilize control techniques to achieve this organization. Insects, such as ants, bees, and termites, deposit pheromones to indirectly coordinate with their colony. This stigmergic communication aids in collective behaviors such as navigation, defense, and brood care. In previous work, we found that stigmergic communication can improve the performance of teams of heterogenous agents in real-time strategy games by communicating agent density and state information (Olsen et al., 2008). However, in the work at hand, we consider a top-down environmental feedback controller that regulates agent density.

PID Controller

Originally developed to control the heading of large ships, the PID control algorithm is now used in a variety of controllers, such as thermostats and automobile cruise controls (Minorsky, 1922). We specify the desired state of the system, or set point, and the PID controller modifies parameters of the system until it is in that state. Formally, the output of the PID as a function of time is given by

$$u(t) = k_p e(t) + k_i \int_0^t e(\tau)d\tau + k_d \frac{de(t)}{dt}, \qquad (1)$$

where $e(t)$ is the error, defined as the set point minus output:

$$e(t) \equiv S - u(t). \qquad (2)$$

The constants k_p, k_i, and k_d simply adjust the weight of the proportional, integral and derivative terms, respectively.

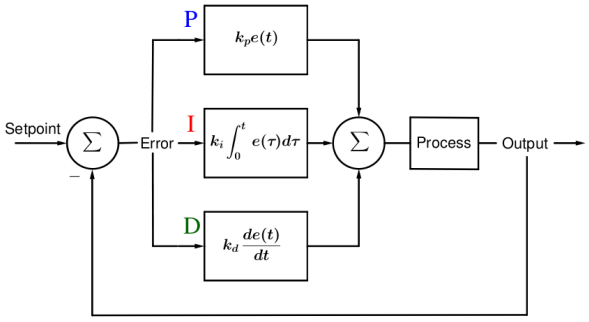

Figure 2: Illustration of the PID Controller.

The goal of the PID controller is to obtain the output such that the set point is obtained, or equivalently when the error is zero.

In our simulation we have a tile floor, and each tile is associated with a feedback controller. The input to the controller is the number of birds within a certain radius of the center of the tile. The set point S_{tile} of the controller in each tile specifies the number of birds which should be in that radius, which we have defined as

$$S_{\text{tile}} \equiv \frac{B_{\text{total}}}{F_{\text{total}}} F_{\text{rad}}, \qquad (3)$$

where B represents birds and F represents food. Our simulation ensures that, $B_{\text{total}}/F_{\text{total}}$ is constant, so $S_{\text{tile}} \propto F_{\text{rad}}$. We see then that the more food within a specified radius of a PID controller, the higher the set point and thus number of desired birds in that tile will be. It is important to note here that our simulation does not use periodic boundary conditions, so birds that fly far from our tile floor will not be subject to PID control, but still expire when they reach zero energy.

For our control algorithm, we consider only the first two terms of the PID algorithm. The output of the controller as a function of time becomes

$$u(t) = k_p e(t) + k_i \int_0^t e(\tau)d\tau. \qquad (4)$$

The proportional term makes a tile attractive if there are fewer birds locally than the set point, and repulsive if there are more birds than the set point. The integral term helps to overcome any steady-state error that may exist in our system due to overshoot and undershoot caused by the proportional term alone. For example, a bird outside an attractive tile's radius may circle that tile as it experiences a constant force; the integral term means the accumulated error will gradually make the tile more attractive in this situation, and the bird will be pulled inwards.

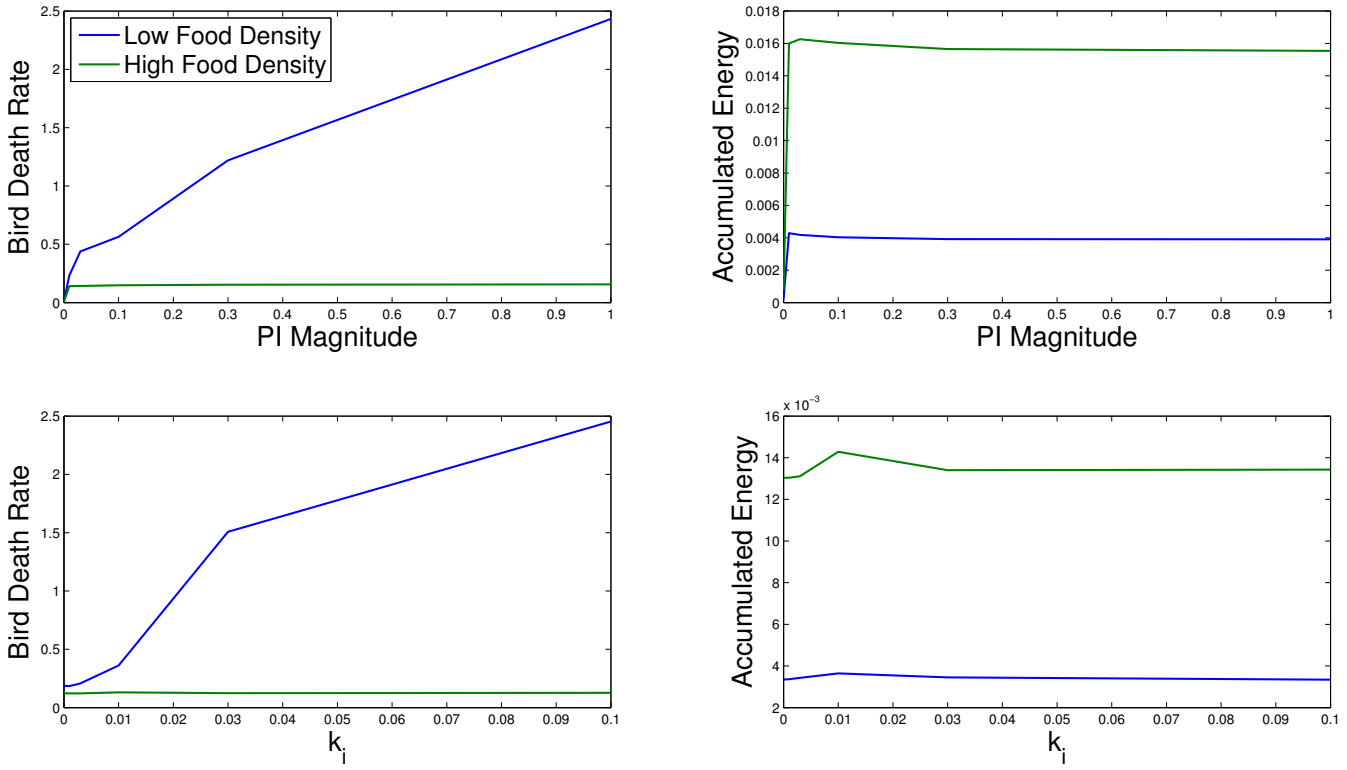

Figure 3: Average fitness for varying values of PI magnitude and k_i. Note the sharp increase in accumulated energy between PI magnitudes of 0 and 0.01

Experiments

All of the following experiments are conducted in Brevis (Harrington, 2014) for 50,000 timesteps. We sample over 3 parameters of the feedback controller: strength of feedback control, integral magnitude, and range of control. Each parameter set is tested on 10 unique random seeds.

Strength of Feedback Control

Our first set of experiments consist of varying the parameters of our PI control on simulations with low (5 food) and high (25 food) food density. Each experiment has 50 birds. Our control experiment has no feedback. When we introduce feedback, we have a parameter which we multiply the output of our PI controller by to modify the magnitude of its effect. We considered a somewhat logarithmic series of values for this parameter: 0.01, 0.03, 0.1, 0.3, and 1. The greatest of these values roughly corresponds to the maximum force felt by birds in the no-feedback experiment.

Integral Magnitude

We also consider a range of values for our integral constant k_i of 0.001, 0.003, 0.01, 0.03, and 0.1, holding the proportional constant k_p at a fixed value $k_p = 1$. At a k_i of 0.1 the accumulated error can saturate extremely quickly, so we do not consider values greater than this.

Range of Feedback Control

The final parameter we vary is the size of our feedback controller. To do this, we select three different radii of increasing size, corresponding to: the circle circumscribing the tile associated with the controller, the circle inscribed in the 3x3 neighborhood of tiles around the controller, and the circle circumscribing the neighbors in the cardinal directions.

Results

There are two primary statistics in our simulation that indicate the fitness of the birds: the rate at which they expire, and the rate at which they accumulate energy. A lower death rate and a higher energy gathering rate would correspond to fitter birds. We expect that having a feedback controller to distribute the birds evenly across the food could improve all aspects of their fitness. However, the results that we see indicate that introducing control increases the rate at which the birds accumulate energy, but it also increases the rate at which they expire.

The feedback control manages to help some birds locate and stay near the food, so they are able to collect more energy overall. This can be seen Fig. 3 with a sharp increase in accumulated energy as soon as the magnitude of PI control becomes greater than zero. As the force of the PI control continues to increase, the amount of energy gathered by the

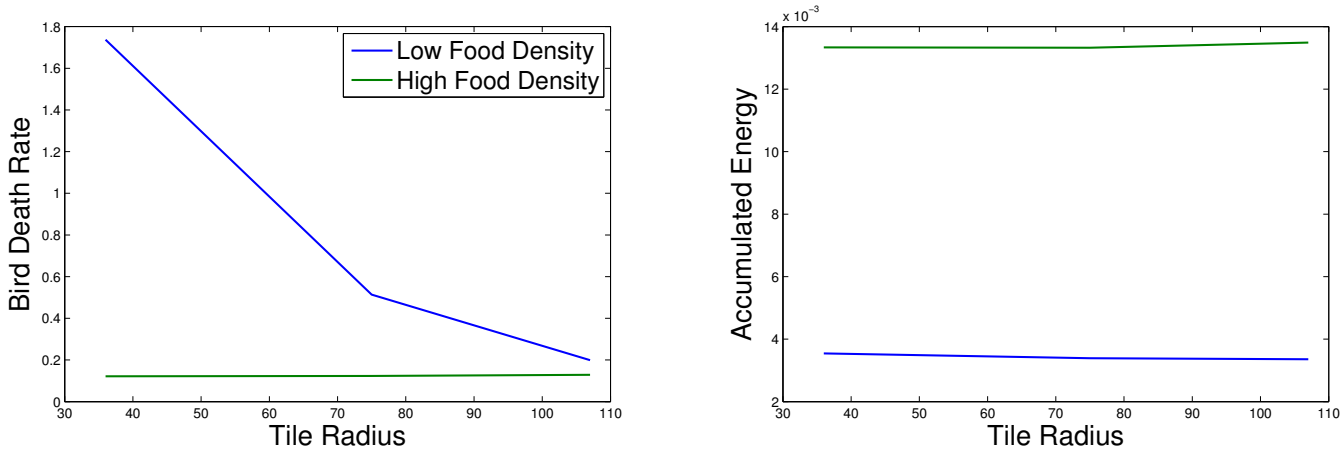

Figure 4: Average fitness for varying tile radii.

birds remains roughly constant. If we imagine a given layout of food, it is likely that there is an optimum distribution of birds across that layout. As we increase the force exerted on the birds by the feedback controller, the genes of the birds will change to compensate for the increased force in such a way that they still distribute themselves evenly across the food. Going from no control to some level of control provides additional information to the birds, allowing them to locate food more easily. Increasing the force of that control does not provide them with any new information, so it does not improve their fitness.

Varying the integral constant of the PI control informs us of what the optimum constant should be. For both low and high food density, we see a maximum of food accumulated when $k_i = 0.01$, shown in Fig. 3. When k_i is too low, the accumulated error barely provides any information compared to the proportional term about what force a controller should be exerting. When k_i is too high, the accumulated error will quickly saturate, resulting in behavior akin to a proportional term with some time lag. When k_i is at a near-optimum value, it will allow each controller to compensate for any steady-state error which may occur. While increasing the magnitude of the feedback does not improve the fitness because it does not provide additional information, the integral constant affects the quality of the information provided by the controllers, which is why it has a more noticeable impact on the amount of energy the birds accumulate.

Increasing the magnitude of the PI control does negatively impact the fitness of the birds as well. Particularly in the low food density case, it will cause the birds to expire at a higher rate. The same trend exists for increasing values of k_i, since higher integral constants will allow for a greater force to be exerted by the controllers, meaning it will have similar effects. We believe increasing the amount of feedback increases the death rate because there exists the possibility that there will be large regions of the simulation without any food. Since the set point of controllers in this region will be 0, if they exert a force on the birds it will only ever be repulsive, due to accumulated error. If birds are introduced into the simulation in these large repulsive regions, it is likely that they will be pushed away from the inner part of our simulation where all the food is located. This decrease of fitness as a result of the introduction of feedback control is not one which needs to exist, but simply a result of our specific implementation. It may be avoided with a change such as only allowing the controllers to exert a positive force of varying magnitude on the birds.

The effects of the radius around a feedback controller for which a bird or food is considered nearby are shown in Fig. 4 and are rather straightforward. In the low food density case, we see a decrease in the bird death rate for increasing tile size. This agrees with our earlier analysis that large regions of repulsive tiles will cause the birds to expire at a higher rate. When the tile radius is larger, more tiles will have food nearby. This will result in more tiles which will tend to be attractive and fewer tiles which will tend to be repulsive, so fewer birds will be pushed outside the region containing the food. Tile radius does not have an effect on the rate at which the birds accumulate energy. This is likely because the birds are able to get the same amount of information from the feedback controllers regardless of their radius. If neighboring controllers overlap, the information about where the bird should go exists in the difference between their outputs. Tile radius may have more of an effect in a system where the controllers are not equidistant from one another.

Looking at the genomes of the birds for varying parameters provides us with more information about the effects of PI control. In practically every simulation the distance for considering food to be far or close converged to 0 for all birds; the ability to have two coefficients did not grant enough extra information to be necessary. This makes it

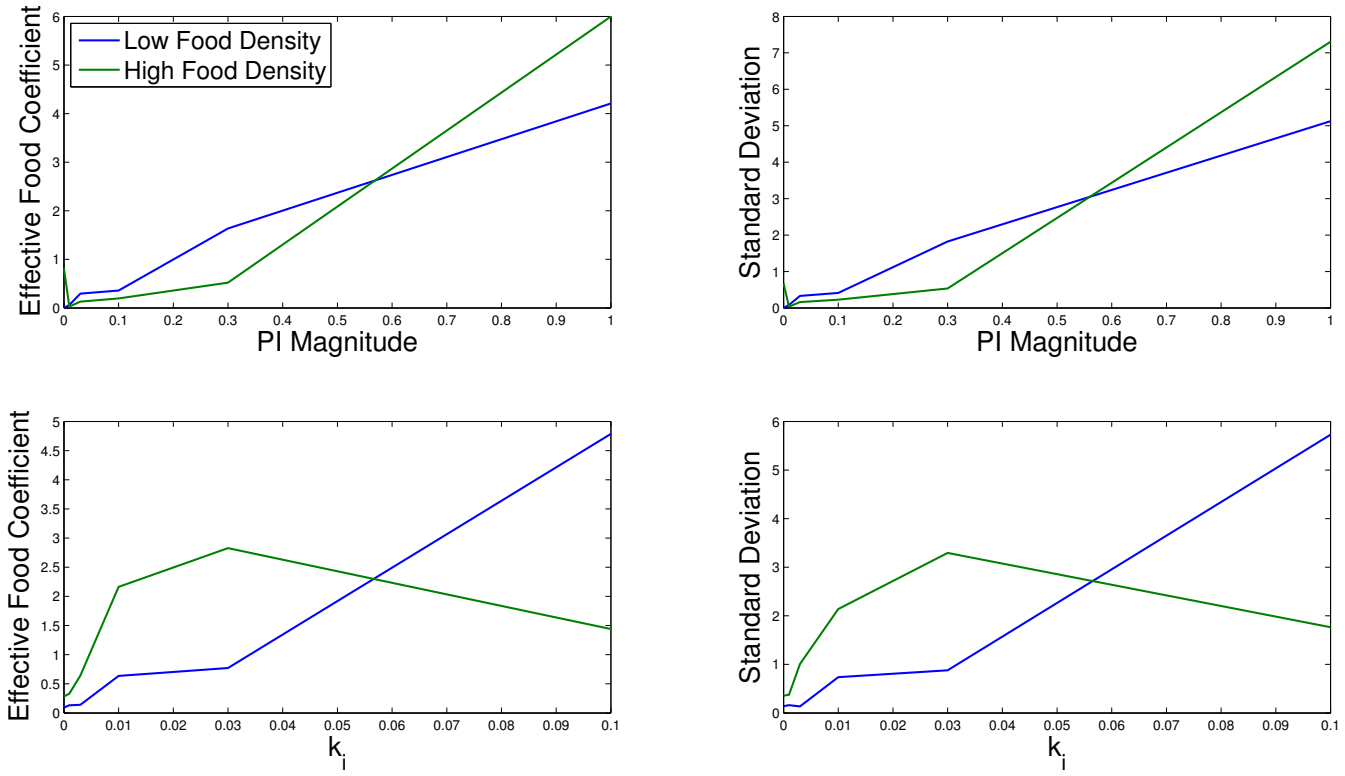

Figure 5: Effective food coefficient gene observed for varying values of PI magnitude and k_i, along with its standard deviation. The effective food coefficient is the value the birds actually use, as in nearly all simulations the food distance converged to 0.

easy to analyze the food coefficient, as we can simply average across the far coefficient. For every parameter we varied we can see a trend that this food coefficient increases as that parameter increases. Since we performed mutation by multiplying the value for a gene by a random number centered at 1, simulations which converge to a greater value will have a greater standard deviation.

For increasing PI magnitude, the forces exerted by the feedback controllers will vary more rapidly with time as the birds move around and the proportional term quickly adjusts. This means that for birds to compensate to this rapidly changing environment, they must be able to accelerate rapidly. This is why we see the positive relationship between PI magnitude and food coefficient in Fig. 5. For increasing values of k_i, we see two different trends for low and high food density. Increasing the integral constant will increase the magnitude of the feedback controller's output. In the low food density simulations the large regions which become repulsive do so because of accumulated error, so for high k_i the birds need to accelerate quickly towards the food to cross these regions. In the high food density case k_i will similarly increase the overall output of the feedback, but the birds do not have to deal with large repulsive regions. For lower values of k_i we still see a positive trend as the accumulating error will result in changes that the birds need to react to, albeit more slowly than the proportional term. However, for higher values of k_i the accumulated error will quickly become saturated, resulting in what will often be a constant attractive or repulsive force. As the strength of an attractive force of this nature will be large relative to the other forces on a bird, that bird will be more successful if it allows itself to be pulled toward the food by the control instead of accelerating towards it on its own and overshooting the target.

The genes for neighbor coefficients display a similar behavior to that of the food coefficient, though the magnitude of these values is considerably lower, as seen in Fig. 6. As all the other birds will be attempting to find and stay close to food, moving towards them will often result in moving towards the food they are near. Since increasing the magnitude of control increases the food coefficient, it follows that it will increase the bird coefficient as well.

Conclusions

We have introduced an environmental feedback controller to a model of swarm evolution. This introduction of control has a significant effect on simulation dynamics; however, the benefit of control is clearly dependent upon the goal of the evolving swarm. In this case, control does not reduce the rate at which birds expire, but instead leads to an increase in energy accumulation. This type of behavior is desirable in a

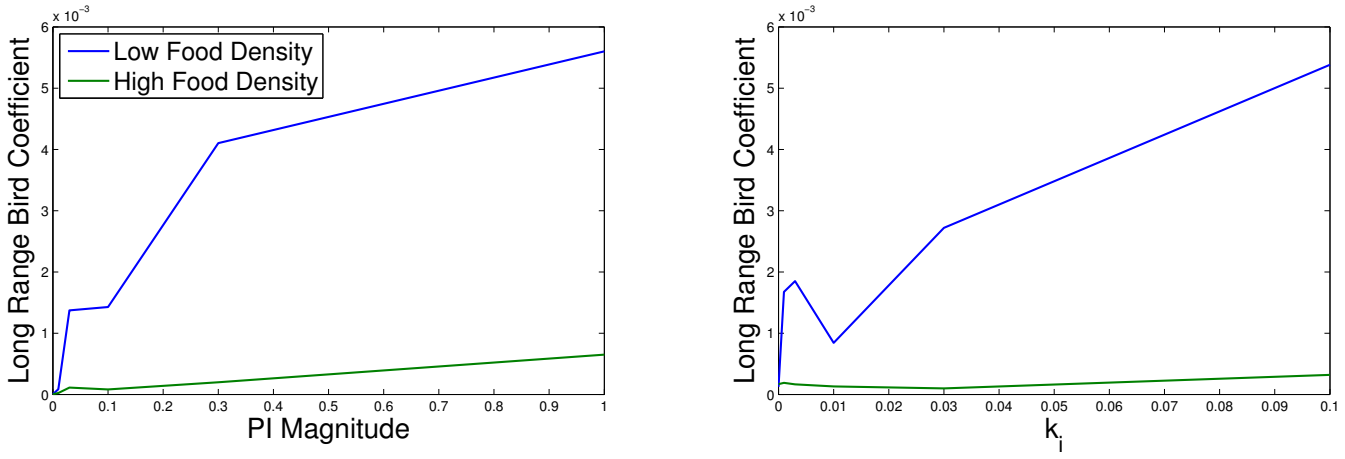

Figure 6: The coefficient birds to determine their acceleration with respect to other birds. Since the bird distance gene did not converge to 0, these values have been averaged over the coefficient birds would use at distance greater than 50, which is higher than the average value of bird distance in any simulation.

number of situations, for example search and rescue, reconnaissance, and crop pollination. Our model can be readily extended to test other ideas relating to the control of evolving swarms, and opens new possibilities for studying the interface between control theory and artificial life.

Future Work

In previous work we have introduced the use of gene regulatory networks to actively tune the parameters of a reinforcement learning-based control algorithm, where the tuning allows the behavior of controllers to change over time (Harrington et al., 2013). Introducing this additional layer to the controller may allow controllers to adopt different modes of control based on context.

To further explore the effects of a PI controller, we plan to incorporate energy sinks in the form of stationary obstructions that take away energy upon collision. Currently, the average bird death rate increases as the magnitude of the controller increases. We expect with the addition of sinks that their associated PI controllers could repel the birds away, and thus decrease the death rate.

We also plan to allow the PI controller to regulate motion in all three dimensions, rather than only the horizontal plane. It would then be possible to place PI controllers on the sources and sinks of energy themselves. The function of the PI controller could also be modified, with the goal of sustainability in mind, pushing birds more strongly towards food sources with high energy as opposed to the nearest food source.

In contrast to our PI controller that promotes convergence toward the food, it would be interesting to implement a pesticide that aims to repel birds from energy sources. We could examine the genes that characterize fit birds in such a system, and if a particular combination of genes can overcome the pesticide. Furthermore, we could construct an optimization problem of how to apply pesticide by introducing a cost proportional to the pesticide strength. We would then search for the minimum pesticide strength that successfully repels birds.

We believe our PI control mechanism which functions on an input of how evenly the birds are distributed could have applications for particle swarm optimization (Zhang and Xie, 2003). Using feedback to try to distribute the particles evenly through the search space could help to maintain diversity while still utilizing the benefits of swarm behavior.

Acknowledgements

KIH is funded by the Brandeis University School of Computer Science. Computing support was provided by the Brandeis University HPC cluster. We thank Seth Fraden, and Jordan Pollack for advice and support. We also thank the UMass BINDS lab for comments, in particular Patrick Taylor.

References

Åström, K.J., and Hägglund, T. (1995). *PID Controllers* (2nd ed.). Research Triangle Park, N.C.: International Society for Measurement and Control.

Åström, K. J., and Murray, R.M. (2008). *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton: Princeton University Press.

Berman, S., Kumar, V., and Nagpal, R. (2011a). Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–385.

Berman, S., Nagpal, R., and Halász, A. (2011b). Optimization of stochastic strategies for spatially inhomogeneous robot swarms: A case study in commercial pollination. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3923–3930.

Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43.

Fukuyama, Y., Takayama, S., Nakanishi, Y., and Yoshida, H. (1999). A Particle Swarm Optimization for Reactive Power and Voltage Control in Electric Power Systems. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., and Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1523–1528, Orlando, Florida, USA. Morgan Kaufmann.

Harrington, K. I. (2014). Brevis (Version 0.7.17).

Harrington, K. I., Awa, E., Cussat-Blanc, S., and Pollack, J. (2013). Robot Coverage Control by Evolved Neuromodulation. In *IJCNN 2013*, page 543-550.

Holt, R. and Hochberg, M. (1997). When is biological control evolutionarily stable (or is it)? *Ecology*, 78:1673-1683.

Minorsky, N. (1922). Directional Stability of Automatically Steered Bodies. *Journal of the American Society for Naval Engineers*, 42(2):280-309.

Munoz, M., Lopez, J., and Caicedo, E. (2007). Bacteria swarm foraging optimization for dynamical resource allocation in a multizone temperature experimentation platform. In *Analysis and Design of Intelligent Systems using Soft Computing Techniques*, pages 427–435.

Olsen, M., Harrington, K., and Siegelmann, H. (2008). Emotions for Strategic Real-Time Systems. In *AAAI Emotion, Personality, and Social Behavior Technical Report*, pages 104–110. Março.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH Computer Graphics*, volume 21, pages 25–34. ACM Press.

Roderick, G. and Navajas, M. (2003). Genes in new environments: genetics and evolution in biological control. *Nature Reviews Genetics*, 4:889-899.

Rubenstein, M., Ahler, C., and Nagpal, R. (2012). Kilobot: A low cost scalable robot system for collective behaviors. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3293–3298.

Sayama, H. (2011). Seeking Open-Ended Evolution in Swarm Chemistry. In *Artificial Life (ALIFE), 2011 IEEE Symposium on*, pages 186–193.

Spector, L. and Klein, J. (2002). Evolutionary dynamics discovered via visualization in the breve simulation environment. In *Workshop Proceedings of the 8th International Conference on the Simulation and Synthesis of Living Systems*, pages 163–170.

Spector, L., Klein, J., Perry, C., and Feinstein, M. (2005). Emergence of Collective Behavior in Evolving Populations of Flying Agents. *Genetic Programming and Evolvable Machines*, 6(1):111–125.

Toner, J. and Tu, Y. (1998). Flocks, herds, and schools: A quantitative theory of flocking. *Physical review E*, 58(4):4828–4858.

Zhang, W.J. and Xie, X.F. (2003). DEPSO: Hybrid Particle Swarm with Differential Evolution Operator. *IEEE International Conference on Systems Man and Cybernetics*, pages 3816–3821.