AIAA 2003–0608

THE AERODYNAMIC BENEFITS OF SELF-ORGANIZATION IN BIRD FLOCKS

Glen A. Dimock and Michael S. Selig
University of Illinois at Urbana–Champaign
Urbana, IL 61801

41st AIAA Aerospace Sciences Meeting and Exhibit
January 6–9, 2003/Reno, NV

THE AERODYNAMIC BENEFITS OF SELF-ORGANIZATION IN BIRD FLOCKS

Glen A. Dimock* and Michael S. Selig†
University of Illinois at Urbana–Champaign
Urbana, IL 61801

Natural aggregation processes such as the familiar flocking of birds have been accurately modeled using a simple, decentralized controller. Variations on this "boid" controller typically involve three or more control laws, each with an associated control gain and sensor range. In this paper, the boid controller is fitted with an additional rule designed to produce aerodynamically-efficient formations, such as those exploited by migratory birds and hypothetical unmanned aerial vehicles. A simple genetic algorithm is then used to optimize the control parameters for minimum power consumption in a flock of simulated birds. This report focuses on the development and utility of the flocking simulator as a fitness function for the GA. Preliminary results indicate that average power consumption can be significantly reduced with the modified, optimized boid controller.

Introduction

BIRD formations have previously been studied on a macroscopic level, revealing that a V-formation experiences substantial drag savings over other, more disparate arrangements.[1] Empirical results indicate that birds in formation can fly farther than their solitary counterparts,[2] and numerical simulations have confirmed that these formations indeed offer drag benefits to the overall flock. The same studies, however, have also shown that the drag experienced by each bird can vary widely throughout a formation,[3] leading to arguments that cooperation must play an important role in sustaining any formation.[4] While cooperation (and sacrifice) may be necessary for optimal formations, this paper explores the near-optimal global drag savings that can result from purely selfish interests at the local level. As opposed to modeling the flock as a cohesive unit, this approach considers a flock comprised of autonomous, uncooperative agents. For the purposes of this paper, a flock is defined as two or more related but autonomous entities, such as birds or airplanes. Formation flight, which may or may not take place within a flock, is defined as relative positioning that is aerodynamically advantageous to at least one member of the formation.

Numerous schemes have been proposed for modeling decentralized systems such as flocks of birds or unmanned aerial vehicles (UAVs). One method, popular in the engineering community, involves the extension of linear classical control theory to examine groups of independent controllers.[5] This approach is less than ideal for explaining self-organizing behavior in natural systems, however, because it typically assumes some a priori knowledge of the optimal relationship between system agents and often cannot predict global stability. To address real biological issues, such as competition and cooperation between agents, a particle system model developed by Reynolds[6] is used instead. His *boid* model assigns simple rules to each system agent and has been cited in a wide variety of disciplines, including recent controls studies.[7] For the purposes of this research, the system agents, dubbed "boids" by Reynolds, are representative of birds or UAVs.

While Reynolds' boid model was originally conceived to produce more realistic computer animations of aggregate motion in animals, it has recently received attention as a promising method for implementing decentralized control in flocks of UAVs.[8,9] These studies have demonstrated the effectiveness of the boid control concept for managing conflicting goals, such cohesion and separation, in a flock of UAVs while also achieving core mission objectives, such as navigation and targeting. Whereas formation flight is useful for reducing drag and extending the range of flock members, however, existing studies have not considered drag reduction or formation flight as an extension to the conventional boid flocking rules. Because endurance is a critical issue in developing practical UAVs, the conventional boid model is herein expanded to include a rule that includes aerodynamic effects and aims to reduce drag. One aim is to show that a priori knowledge of an optimum formation is unnecessary in the pursuit of global drag savings; individual selfish interests are sufficient to benefit the group and even produce the ubiquitous V-formation commonly found in migrating bird flocks. This paper focuses on the simulation of bird flight, demonstrating that the boid controller, when optimized for drag reduction, is con-

*Graduate Research Assistant, Dept. Aero and Astro Engineering. Student Member AIAA. dimock@uiuc.edu

†Associate Professor, Dept. Aero and Astro Engineering. Senior Member AIAA. m-selig@uiuc.edu

sistent – at least superficially – with the designs nature has evolved over hundreds of millions of years.

Implementing a boid-style controller requires the choice of numerous controller gains and sensor ranges for optimal results, which by hand is a tedious and often intractable process due to the nonlinear nature of the boids system. Instead, this paper proposes the utility of a simple genetic algorithm for optimizing three gains and two sensor ranges specific to the problem of power reduction in formation flight. Because of the nature of this problem, the fitness evaluation is necessarily a simulation of formation flight, suggesting a very computationally-expensive task. Consequently, the development of a simple but valid, consistent fitness function is of paramount importance in this problem and constitutes the bulk of this research. Consistency in the fitness function refers to constant results for various starting conditions and noise inputs.

Related Work

Researchers have previously used genetic programming (GP) to develop decentralized controllers for coordinated group motion, with varying degrees of success. Reynolds began applying GP to autonomous agents in 1992[10] and continued this work for several years.[11–14] In these studies, he examined both homogeneous and heterogeneous populations during fitness trials, as well as the effects of noise in fitness trials, determining that noise plays a major role in evolving robust controllers with GP. Other researchers continued the trend of heterogeneous, competitive fitness environments,[15,16] demonstrating that multiple trials are needed for accurate ranking, due to the strong fitness dependence on the individuals present in each population. In 1996, Zaera et al.[17] attempted to evolve a schooling controller using a homogenous fitness environment and failed. Formulating the fitness function proved to be a problem in this study, as with others.

From the results of these papers, it was decided that genetic programming would not be attempted in the first iteration of this study, but rather a genetic algorithm to optimize parameters for existing, fixed control laws. Reynolds demonstrated that GP-designed controllers are often "brittle" in the presence of noise, an undesirable trait for any airplane controller. Secondly, although coevolution with a competitive fitness environment was originally considered for this project, this approach was abandoned in favor of homogenous fitness trials. Coevolution, while offering a more realistic simulation of biological systems and allowing for the evolution of multiple optimal controllers, is more computationally expensive than a homogenous approach and would have required significantly more computer time than was available for this research. Finally, the importance of using a consistent fitness function is underscored by the work of Reynolds and others.

Flocking Simulator

The specialized controller presented here is derived from Reynolds' original version, using a subset of the original behavioral rules and adding a new aerodynamics rule. In Reynolds' system, each agent is controlled by a set of rules that generate behavior according to the state of other agents in the system. This behavior usually takes the form of an acceleration vector. In the simplest case, as is used in this study, it is assumed that each boid commands perfect, realtime knowledge of the system state and that acceleration may be applied uniformly in any direction. Of course, a commercial grade controller would need to contend with system, actuator and sensor dynamics.

A modified boid controller is implemented as part of a discrete-time, multi-agent simulation, coded in C++ on a desktop PC. The aerodynamic drag-reduction rule includes a discrete lifting line model, which predicts the induced drag associated with each bird at each time step. While facilitating various calculations associated with the drag-reduction rule, the drag prediction function also accommodates fitness evaluations for the GA, as described below.

Controller Rules

Reynolds' original boid model involves three basic rules, of which two are used in this study: cohesion and collision avoidance. A drag-reduction rule, based on a simple but systems-level accurate aerodynamic model, is added to the original rules. Figures 1a and 1b illustrate the conventional rules of cohesion and collision, respectively. Figure 1c shows the new drag reduction rule, and Fig. 1d provides a key.

For each boid in the flock, each rule produces a velocity demand that is derived from the state of each other boid and weighted according to a constant gain. In addition, each rule (except for drag-reduction) also has an associated sensor range, which determines the relative contribution of each other boid to the rule. The velocity demands are summed and scaled to generate a final acceleration demand, which takes into account the system dynamics of the vehicle. The mathematical implementation of the controller rules, adapted from Reynolds[6] and Crowther et al.,[8] is described below.

Cohesion

Cohesion, the first and most "central" rule to any flock, states that the individual should move toward the centroid of the flock. This rule nominally employs a small gain and a large sensor range, wherein the urge to congregate is just strong enough to prevent lone boids, and the range is large enough to include the entire flock.

The velocity demand on each boid due to the cohe-

sion rule is given as

$$\mathbf{V_C} = K_C \frac{\mathbf{X_C}}{|\mathbf{X_C}|}\left[1 - \exp\left\{-\left(\frac{|\mathbf{X_C}|}{R_C}\right)^2\right\}\right] \qquad (1)$$

where K_C is the gain associated with this rule and R_C is the sensor range. The centroid $\mathbf{X_C}$ is calculated as

$$\mathbf{X_C} = \frac{\sum_{j=1}^{n} w_j \mathbf{X_j}}{\sum_{j=1}^{n} w_j} \qquad (2)$$

where $\mathbf{X_j}$ is the position of each other boid j, and w_j is the "distance" weight associated with each other boid, namely

$$w_j = \exp\left\{-\left(\frac{|\mathbf{X_j}|}{R_C}\right)^2\right\} \qquad (3)$$

Using this method, the sensor range has two effects: to weight the relative location of each other boid (distant boids have little effect) and to weight the location of the flock centroid (incentive to return more quickly to a far-away flock). The rule gain linearly scales the overall effect.

Collision Avoidance

The avoidance rule is essentially the inverse of the cohesion rule, preventing neighboring boids from colliding. The avoidance-rule gain is nominally set high, with a small sensor range; avoiding nearby boids is very important.

The avoidance velocity demand is calculated similarly to the cohesion velocity, except that the rule gain is now inversely proportional to the centroid distance of neighboring boids (the closer the threat, the stronger the rule). Thus, Eq. 1 is adapted for collision avoidance to yield

$$\mathbf{V_A} = K_A \frac{\mathbf{X_A}}{|\mathbf{X_A}|} \exp\left\{-\left(\frac{|\mathbf{X_A}|}{R_A}\right)^2\right\} \qquad (4)$$

and all of the other cohesion equations are reused for this rule, replacing the cohesion C subscript with the avoidance A subscript.

Drag-Reduction Rule

The novel aerodynamics rule aims to reduce each boid's power consumption by computing the induced drag gradient and producing a velocity demand along this gradient. As for physical motivation, this rule assumes that birds are capable of sensing small changes in drag as they move through space. In a system of UAVs, this gradient could be computed based on their relative locations with respect to one another, assuming the presence of a wake model.

Although the aerodynamics of birds in flapping flight are decidedly unsteady and complex, a steady-state aerodynamic model is used as a first approximation. It is also assumed that induced drag is the most

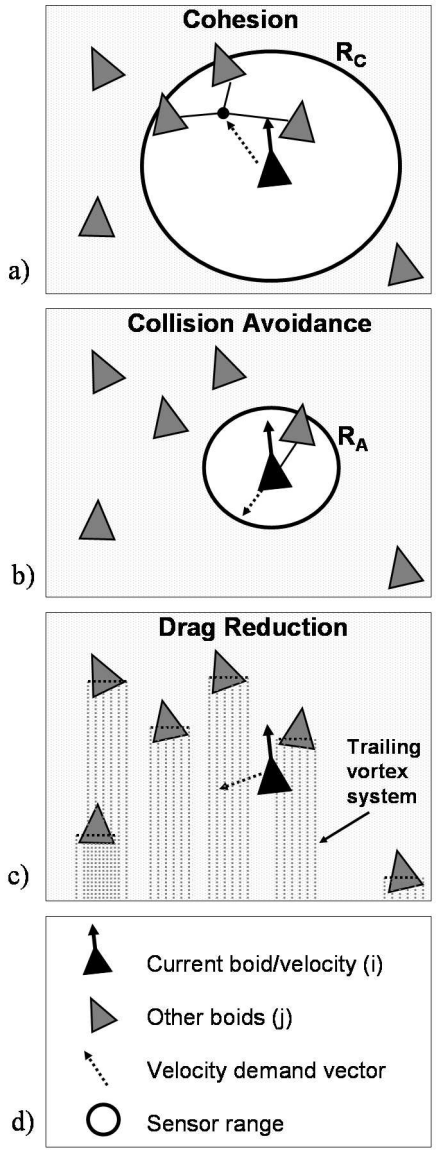

Fig. 1 Flocking rules used in this research: cohesion (a), collision avoidance (b) and drag reduction (c). Symbols are given in (d).

variable and the only significant drag term (within the context of this simulation), that vorticity is shed purely along the x axis (an adequate approximation if the bird's velocity is mostly forward), does not decay and always consists of straight filaments that remain bound to the bird's wings, and that all birds have identical, constant physical parameters.

In classic aeronautical fashion, each boid is modeled as a discrete vortex system, with the wake assumed to lie in the horizontal plane (see Fig. 2). Induced drag (Fig. 3) on a wing in the presence of a vortex is known to vary proportionally to the strength and distribution of the downwash induced by the vortex, and the total drag may be obtained by integrating the effect of every vortex in the system over the span of the wing. A more

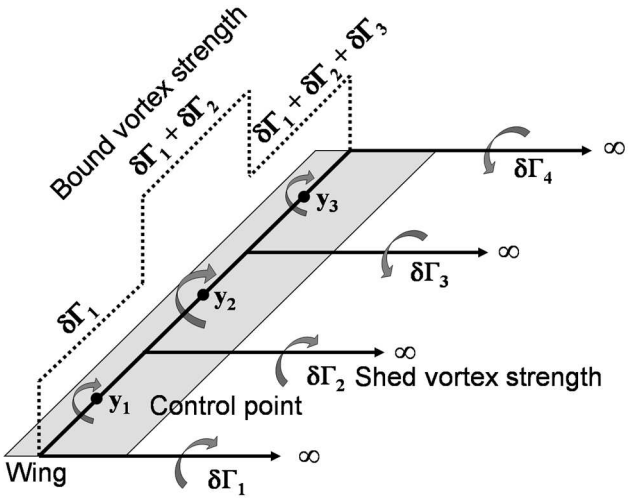

Fig. 2 Discrete vortex system for a three-dimensional wing. The vortex system has been simplified for clarity of illustration; 10 discrete filaments were used in the simulations.

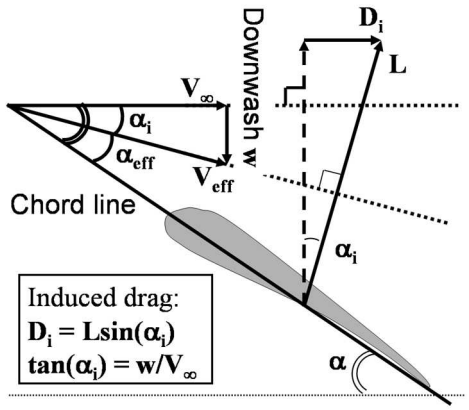

Fig. 3 Induced drag for a three-dimensional wing.

comprehensive discussion of discrete vortex theory and induced drag is given in Ref. 18.

Figure 4 illustrates the downwash induced at a given point on one wing due to a discrete trailing vortex originating on another wing. It can be shown that the vertical z component of the downwash at an offset y_p on a wing centered at P in space, due to a discrete vortex filament of strength $\delta\Gamma$ trailing from an offset y_q on a wing originating at point Q is:

$$w = \frac{-\delta\Gamma\left(P_y - y_q + y_p - Q_y\right)\left(-P_x + Q_x + A\right)}{4\pi\left(\left(P_y - y_q + y_p - Q_y\right)^2 + \left(P_z - Q_z\right)^2\right)A} \quad (5)$$

where

$$A = \sqrt{\left(P_x - Q_x\right)^2 + \left(P_y - y_q + y_p - Q_y\right)^2 + \left(P_z - Q_z\right)^2}$$

and that the vertical component of the downwash at the same point, due to a bound vortex of strength Γ stretching between y_1 and y_2 on the same wing originating at Q is:

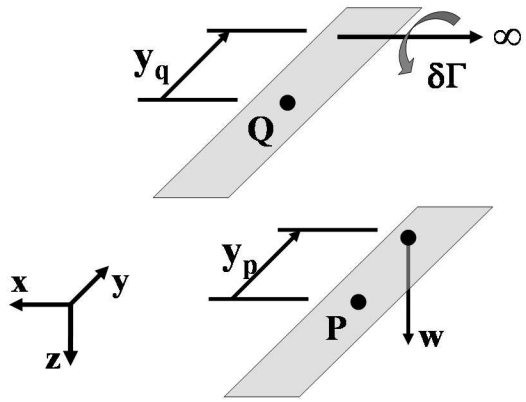

Fig. 4 Induced downwash on a wing at point P due to a trailing vortex from another wing at point Q.

$$w = \frac{\Gamma\left(-P_x + Q_x\right)\left(B_1 + B_2\right)}{4\pi\left(\left(P_x - Q_x\right)^2 + \left(P_z - Q_z\right)^2\right)} \quad (6)$$

where

$$B_1 = \frac{P_y - y_1 + y_p - Q_y}{\sqrt{\left(P_x - Q_x\right)^2 + \left(P_y - y_1 + y_p - Q_y\right)^2 + \left(P_z - Q_z\right)^2}}$$

and

$$B_2 = \frac{-P_y + y_2 - y_p + Q_y}{\sqrt{\left(P_x - Q_x\right)^2 + \left(P_y - y_2 + y_p - Q_y\right)^2 + \left(P_z - Q_z\right)^2}}$$

Note that when computing downwash, points P and Q are constrained to a grid of spacing Δy, which corresponds to the distance between vortex filaments. This constraint is necessary in discrete lifting line theory. In the equations above, P_* refers to the $*$ component of point P, i.e. $P_x = P \cdot \hat{i}$.

Summing these two expressions for downwash at a single station on a wing due to all vortex filaments and bound vortices in the system of wings, it is possible to iteratively solve for the bound vorticity distribution $\Gamma(y)$ on each wing.[18] In this study, a rectangular wing with zero twist is assumed, and the geometric angle of attack is solved iteratively as an inner loop while solving for $\Gamma(y)$. The angle of attack α is determined from the constraint $L = W$.

After solving for $\Gamma(y)$ at each time step, the induced drag on each wing may be expressed as

$$D_i = \rho V \sum_{j=1}^{k} \Gamma(y)\alpha_i \Delta y \quad (7)$$

The drag-reduction rule accelerates each boid in a direction of decreasing induced drag, which requires the calculation of the induced drag gradient. This procedure is performed numerically, by perturbing each

boid plus and minus one grid space Δy in each orthogonal direction and computing the induced drag, as described above, at each location. To compute the velocity demand, the negative gradient is taken to yield a direction of decreasing induced drag.

$$\mathbf{V_D} = -K_D \mathbf{V_d} \tag{8}$$

where $\mathbf{V_d}$ is the drag gradient computed above.

Acceleration

Upon calculating the weighted velocity demand for each individual rule, the final velocity demand is a simple sum of these parts:

$$\mathbf{V} = \mathbf{V_C} + \mathbf{V_A} + \mathbf{V_D} \tag{9}$$

This simulation treats the flock as a particle system and simply assigns the new velocity at each time step. These assumptions are reasonable when the final solutions is expected to be steady state.

Fitness Evaluation

For the purposes of fitness evaluation, the above rules are implemented in a C++-based simulation of five identical boids, evaluated over 20 sec of flight in time steps of 0.1 sec. The boids' physical parameters are as follows: wingspan $b = 5$ ft, chord $c = 1$ ft, weight $W = 10$ lb, freestream velocity $V = 30$ ft/sec, zero-lift angle of attack $\alpha_0 = -1$ deg, lift curve slope $a = 2\pi$, 10 discrete vortex filaments, standard sea-level atmospheric conditions. Because the ultimate objective of the research is to produce a controller that results in the lowest average power consumption for a flock of boids, (negative) average power consumption is used as a fitness measure. The fitness of any individual controller is then

$$f = -\frac{1}{n}\sum_{i=1}^{n}\frac{w}{t} \tag{10}$$

where w is the total work expended by each of n boids over the simulation run, and t is the time elapsed (20 sec). At each time step, the work w is incremented by $D_i V \Delta t$ for each boid.

In addition, the boids are subject to the constraint that they may not collide. While the collision avoidance rule is used for this purpose, it provides no guarantee; a small control gain or large sensor range may render this rule ineffective. A collision is defined as a separation distance of less than one grid space Δy between any two individuals and is checked for at each time step. Should a collision occur, the simulation is halted, and a fixed fitness value of -1000 (lower than any expected normal fitness value) is assigned. It is not possible to differentiate between the fitness values of controllers suffering from collisions, because their simulations are necessarily incomplete. Although the power consumption up until the time of collision could be used as a fitness measure, this practice could improperly favor controllers that tend to increase their power consumption (but collide) as time progresses. Therefore, controllers that suffer collisions are effectively thrown out during tournament selection by using an artificially-low fitness value. In the event that two unfit controllers are paired in a tournament, one is selected at random.

The boids are also subject to the obvious constraint that the aerodynamic equations must converge at each simulation step. In the rare event that convergence does not occur, a fixed fitness value of -1000 is assigned, for the same reasons as with any collision. The equations have been observed to diverge under very specific circumstances, but this happens so rarely (less than 0.5% of all cases) as to not significantly affect the performance of the GA.

For consistency, the boids are always started from a V-formation. This general formation is known to be a near-optimum aerodynamic shape,[1] so it is expected that optimum controller gains and sensor ranges will produce a stable V-formation for best power consumption. Allowing the boids to produce a V formation when started from random or non-V formations presents additional challenges and is left for future work.

Evolutionary Optimization

The genetic algorithm (GA) is a type of evolutionary optimization often applied to problems with large, nonlinear design spaces that cannot be effectively attacked using conventional, calculus-based methods.[19] A simple genetic algorithm (SGA), a C++ adaptation of Goldberg's Pascal SGA, is used to optimize five parameters in this reserach: cohesion gain K_c, collision avoidance gain K_a, drag-reduction gain K_d, cohesion sensor range R_c, and collision avoidance sensor range R_a. To enforce bounds on the values the parameters may take, each floating-point parameter is mapped to an integer with 10 bits of precision, such that an integer value of zero corresponds to the minimum acceptable floating-point parameter value, and 2^10 represents the maximum acceptable value. The five integer mappings, stored as 10-bit binary strings, are concatenated in the order given above into a chromosome of length 50. From this representation, each parameter may assume one of 1024 (2^{10}) possible values, yielding a design space of 1125899906842624 (2^{50}) possible unique strings.

Before any optimization is performed using integer mapping, it is necessary to determine the acceptable bounds on the control parameters. The simulation is run with hand-selected parameter values, and "reasonable" values are determined by noting regions of undesirable results, including instability and unrealistic velocity demands. This process yields a range of 0–100 for each control gain, 3–100 for the cohesion sen-

sor range, and 2.5–50 for the collision avoidance sensor range. It should be noted that the minimum collision avoidance sensor radius is chosen to be nearly large enough to encompass the entire wing, and that the cohesion sensor range should not fall within the collision avoidance range.

It is expected that the most significant building blocks in this problem will encompass the high bits for each of the five parameters. These bits determine the general size of each value (large or small) and will likely converge early in the GA run. Also, because some of the parameters are clearly linked (cohesion gain and sensor range for example), the schemata describing the combined high bits of each pair (or triplet) will be important in determining fitness. In fact, the three gain parameters are located adjacent to one another in the chromosome for this reason; it is expected that the most significant coupling will take place between these parameters. If this is so, then important building blocks encompassing multiple gains will stand a higher chance of survival if they can be kept short.

Three basic operators are used in this simple GA application: selection, crossover and mutation. This set is chosen to keep the GA portion of the study relatively simple while focusing on the fitness evaluation, with the expectation that a competent GA or hybrid method will be attempted in a future study. Furthermore, previous related studies have successfully used SGAs and performed trade studies to determine appropriate parameters for this problem class. Based on the works of Reynolds et al., tournament selection is used with a selection pressure of 2, crossover probability is set at 1, and the mutation rate is 0.01. Population size is set at 50, a value determined by the available CPU time (see below). These values are not varied, and a trade study of GA parameters is not performed as part of this research.

Using the SGA and fitness function described above, one controller evaluation requires approximately 30 sec of processing time on a 1.8 GHz Pentium 4 processor. This figure suggests a run-time of 25 min per generation, assuming 50 individual controllers, which adds to several days when considering a typical number of generations (more than 100). Because this study aims to refine the fitness function by performing multiple GA runs, a parallel processing scheme was developed using existing C++ network functions developed as part of another project. Briefly, this system consists of a GA "host" machine that farms out fitness evaluations to many networked "client" machines. By installing the client application on 10 or more PCs with Internet access, it is possible to decrease the GA runtime from several days to several hours.

Results

The fitness evaluation simulation underwent several major iterations, as weaknesses were discovered and exploited by the GA. For example, the original simulation modeled each boid as a horseshoe vortex system, wherein only two trailing vortices exist for each wing. The GA quickly revealed that boids with high drag-reduction gain tend to form a row in the direction of flight, exploiting an artifact of the (overly) discrete nature of the system. The model was eventually refined to its current state through similar encounters, along the way adding multiple vortex filaments and the discrete gradient method, and solving for $\Gamma(y)$ and α iteratively. The resulting simulation is much less prone to exploitation by the GA.

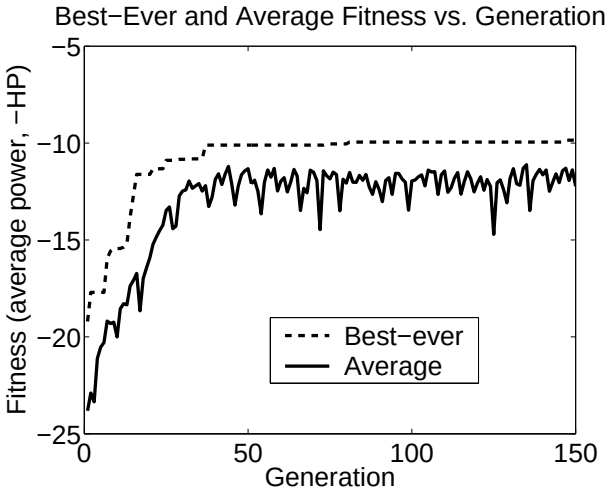

Fig. 5 Average and best fitness vs. generation (controllers unfit due to collision or non-convergence omitted)

Figure 5 shows the average and best-ever fitness results over 150 generations using the coding, operators and fitness evaluation described above. Note that the plotted fitness values do not include the artificially-low numbers of unfit controllers, i.e. those who collided or did not converge. The GA clearly converges, reaching a best-fitness value of −9.8 HP. Furthermore, this value is significantly improved over the best fitness the authors were able to produce by hand (−13 HP).

Figures 6–8 illustrate the properties of the best-ever controller early in the run, midway through, and after 150 generations. As discussed above, it is expected that an optimum controller will fly in a stable V-formation. Indeed, by studying the position traces at these various stages of development, it is observed that the early controller flies in a chaotic pattern, whereas the best-ever controller only adjusts the angle of the V and maintains the formation thereafter. The results of this flocking behavior are reflected in the plots of average induced drag, which is expected to decrease over time for a controller forming a beneficial formation. Such is the case for the best-ever controller, who refines his V formation and enjoys reduced drag as a result.

One major problem with the current fitness evalu-

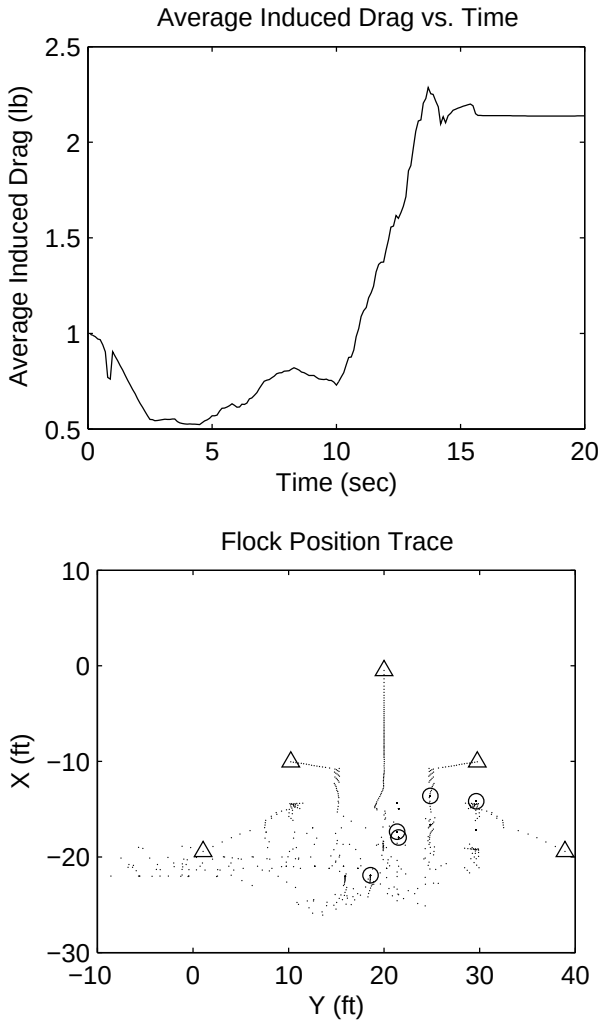

Fig. 6 Position trace and average drag for the best controller after generation 5 (EARLY). Boids are moving from △ to ◯.

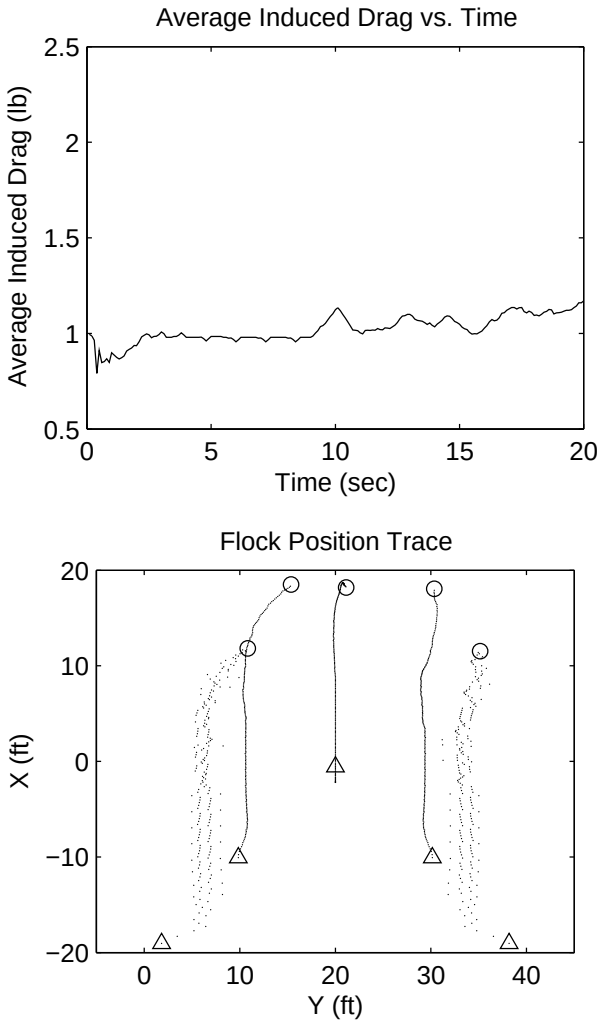

Fig. 7 Position trace and average drag for the best controller after generation 13 (MIDWAY to convergence). Boids are moving from △ to ◯.

ation, not evident in these plots, is that of collision avoidance. The parameters of the best-ever controller are as follows: $K_c = 31.05$, $K_a = 0.29$, $K_d = 7.42$, $R_c = 33.37$, $R_a = 3.52$. Here, it may be observed that the collision avoidance gain is too small to prevent most collisions, and the collision avoidance sensor range is too small to detect most nearby boids. This result presents a problem, for the controller is too aggressively pursuing drag-reduction at the risk of possible collisions. Unfortunately, the fitness evaluation fails to detect most collisions for two reasons: the total simulation time is short so as to reduce GA run times, and the boids always start from the same positions, leading to a somewhat "brittle" solution. As with previous model problems, the GA took full advantage of this weakness. The small collision avoidance gain took hold within 10 generations, causing over 75% of all controllers to have $K_a < 2.0$ by this time. Collisions began to rise, but remained too sporadic to weed these members from the population.

A possible solution for this problem would be to restrict the collision avoidance gain to higher values, but such an action would not necessarily prevent all collisions. Both of the other rules generally lead to cohesion, causing ambiguity in the proper minimum value for this gain and associated sensor range. A better solution would involve additional simulations designed to encourage collisions within each fitness evaluation, possibly by starting the boids in tightly-packed formations. Alternatively, the collision avoidance velocity demand could be made even more nonlinear with respect to separation distance. It currently increases exponentially with respect to distance, but a stronger demand at close range could be achieved by using values larger than e for the exponential term.

Despite the lingering problems with collision avoidance, it is clear from these results that the GA is capable of optimizing a modified boid-style controller

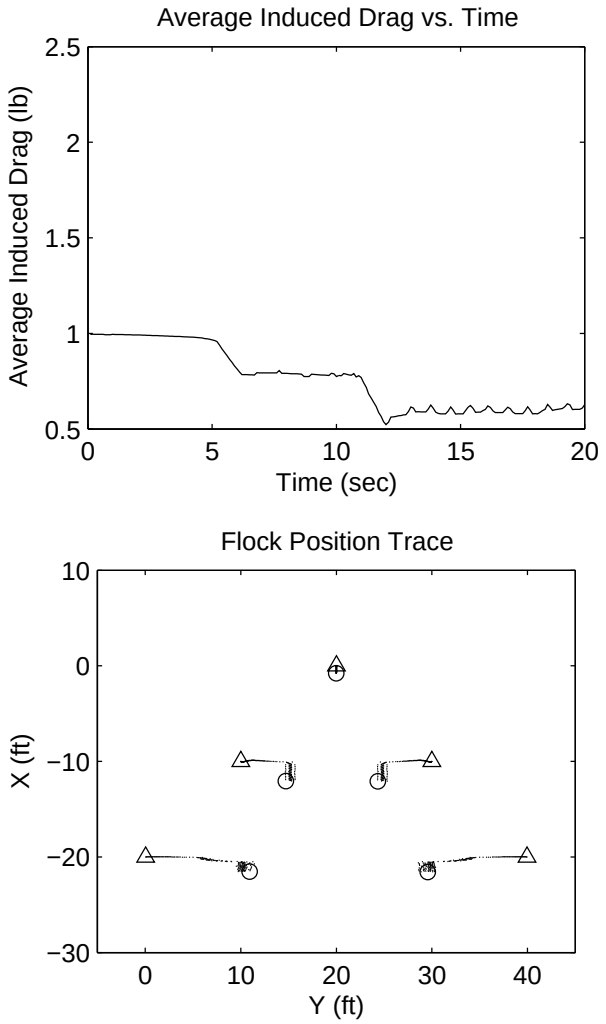

Fig. 8 Position trace and average drag for the best controller after generation 150 (BEST overall). Boids are moving from △ to ◯.

to produce energy-efficient boids capable of maintaining V-formations.

Future Work and Applications

Several avenues exist for future research into this topic. First, ongoing work continues to refine the fitness evaluation, including improved constraint-checking. Collisions are an obvious candidate for the new checks, but additional constraints may also be introduced. These include no steady-state oscillations (the current best controller tends to oscillate about other birds' trailing vortices when in the rear of a stable V) and a requirement for a stable formation after some fixed time period.

The next obvious improvement to the fitness evaluation would be to replace the current particle system with a full, 6-DOF aircraft model for each boid, possibly including sensor models. This step would add relevance for the control scheme to physical systems, and plans are currently underway to use an existing helicopter model for this purpose.

Given that the original motivation for this project was to study the flocking behavior of geese, an unsteady aerodynamic model would greatly enhance the applicability of this controller to actual birds. Low Reynolds number models of flapping flight are, themselves, topics of current research and would be well-applied to this topic.

In addition to these potential areas of future research, there exists the possibility of applying the boid controller to the control of a flock of UAVs. Unmanned aircraft are currently a popular research topic, especially with respect to formation and swarming flight. While a more rigorous stability analysis would be required before using such a controller on any physical system, this scheme and elements thereof offer the very real possibility of reducing power consumption in a flock of autonomous airplanes.

Conclusions

Birds are simulated using Reynolds' flocking rules, which are designed to reproduce the aggregate motion of animals. A new aerodynamics rule is added to the conventional set in order to encourage birds to fly in a direction of decreasing induced drag within the flock. A simple genetic algorithm is used to optimize the gains and sensor ranges of the various rules, with the goal of reducing the average power consumption in the flock. Results show that systems of autonomous agents playing by Reynolds' rules and seeking only to reduce their own induced drag, can achieve a global decrease in average induced drag and power consumption. While these results may help to better understand migratory bird flocking behavior from an aerodynamic perspective, a more practical application would be a control strategy for flocks of autonomous UAVs. Whereas previous flocking studies of UAVs have focused on cohesion and collision avoidance, the addition of this simple aerodynamics rule could significantly increase the range of a UAV flock.

References

[1]Filippone, A., "Heuristic Optimization Applied to an Intrinsecally Difficult Problem: Birds Formation Flight," AIAA Paper No. 96-0515, Reno, NV, 1996.

[2]Alexandre, P., Clerquin, Y., Jiraskova, S., Martin, J., and Weimerskirch, H., "Energy Saving in Flight Formation," *Nature*, Vol. 413, 2001, pp. 697–698.

[3]Lissaman, P. B. S., "The Facts of Lift," AIAA Paper No. 96-0161, Reno, NV, 1996.

[4]Cutts, C. and Speakman, J., "Energy Savings in Formation Flight of Pink-Footed Geese," *J. Exp. Biol.*, Vol. 189, 1994, pp. 251–261.

[5]Chichka, D., Speyer, J., and Wolfe, J., "Decentralized Controllers for Unmanned Aerial Vehicle Formation Flight," AIAA Paper No. 96-3833, San Diego, CA, 1996.

[6]Reynolds, C. W., "Flocks, Herds, and Schools: A Distributed Behavioral Model," *Computer Graphics*, Vol. 21, No. 4, 1987, pp. 25–34.

[7]Jadbabaie, A., Lin, J., and Morse, A. S., "Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules," *IEEE Transactions on Automatic Control*, submitted 2002.

[8]Crowther, B. and Riviere, X., "Flocking of Autonomous Unmanned Air Vehicles," *Proc. 17th Bristol Int'l Conf. on Unmanned Air Vehicles*, Bristol, UK, 2002.

[9]Anderson, M. and Robbins, A., "Formation Flight as a Cooperative Game," AIAA Paper No. 98-4124, Boston, MA, 1998.

[10]Reynolds, C. W., "An Evolved, Vision-Based Behavioral Model of Coordinated Group Motion," *From Animals to Animats 2*, edited by J.-A. Meyer, H. L. Roitblat, and S. W. Wilson, Proc. Second Int'l. Conf. on Simulation of Adaptive Behavior, MIT Press, 1992, pp. 384–392.

[11]Reynolds, C. W., "An Evolved, Vision-Based Model of Obstacle Avoidance Behavior," *Artificial Life III*, edited by Christopher G. Langton, Vol. 17 of *SFI Studies in the Sciences of Complexity, Proc.*, Addison-Wesley, 1994, pp. 327–346.

[12]Reynolds, C. W., "Evolution of Corridor Following Behavior in a Noisy World," *From Animals to Animats 3*, edited by D. Cliff, P. Husbands, and J.-A. Meyer, Proc. Third Int'l. Conf. on Simulation of Adaptive Behavior (SAB94), MIT Press, 1994, pp. 402–410.

[13]Reynolds, C. W., "Competition, Coevolution and the Game of Tag," *Artificial Life IV*, edited by R. Brooks and P. Maes, Vol. 18 of *SFI Studies in the Sciences of Complexity, Proc.*, MIT Press, 1994, pp. 59–69.

[14]Reynolds, C. W., "Evolution of Obstacle Avoidance Behavior: Using Noise to Promote Robust Solutions," *Advances in Genetic Programming*, edited by K. E. J. Kinnear, MIT Press, 1994.

[15]Angeline, P. J. and Pollack, J. B., "Competitive Environments Evolve Better Solution for Complex Tasks," *Genetic Algorithms: Pro. Fifth Int'l. Conf.*, edited by S. Forrest, Morgan Kauffman, San Mateo, Calif., 1993, pp. 264–270.

[16]Sims, K., "Evolving 3D Morphology and Bahavior by Competition," *Artificial Life IV*, edited by R. Brooks and P. Maes, Vol. 18 of *SFI Studies on the Sciences of Complexity*, MIT Press, 1994, pp. 28–39.

[17]Zaera, N., Cliff, D., and Bruten, J., "(Not) Evolving Collective Behaviours in Synthetic Fish," *From Animals to Animats 4*, edited by P. Maes, M. J. Mataric, J.-A. Meyer, J. Pollack, and S. W. Wilson, Proc. Fourth Int'l. Conf. on Simulation of Adaptive Behavior, MIT Press, 1996, pp. 635–644.

[18]Anderson, J., *Fundamentals of Aerodynamics*, McGraw-Hill, Inc., New York, 1991.

[19]Goldberg, D., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.