Coevolution of Form and Function in the Design of Micro Air Vehicles

Magdalena D. Bugajska and Alan C. Schultz

Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory, Code 5515
4555 Overlook Ave., S.W.
Washington, DC 20375

{magda, schultz}@aic.nrl.navy.mil
tel. (202) 404-4946 fax. (202) 767-2166

Abstract

This paper discusses approaches to cooperative coevolution of form and function for autonomous vehicles, specifically evolving morphology and control for an autonomous micro air vehicle (MAV). The evolution of a sensor suite with minimal size, weight, and power requirements, and reactive strategies for collision-free navigation for the simulated MAV is described. Results are presented for several different coevolutionary approaches to evolution of form and function (single- and multiple-species models) and for two different control architectures (a rulebase controller based on the SAMUEL learning system and a neural network controller implemented and evolved using ECkit).

1. Introduction

This study is motivated by the belief that the natural process of coevolving the form and function of living organisms can be applied to the design of morphology and control behaviors of autonomous vehicles in order to simplify the design process and improve the performance of the system. The work presented here is a continuation of the research published in [2].

In this study, the concept of the coevolution of form and function is applied to the Micro Air Vehicles (MAVs) domain. Due to the size of the aircraft (wingspan on the order of 6 inches) as well as the variety of applications, the design of the sensory payload and the controller of the MAV, is quite complex due to the complex relationships between them. The design issue addressed explicitly in this study is minimization of weight and power requirements. The number of sensors and their sensing capabilities affect these requirements directly and also indirectly through the increase of computational power requirements. The goal of the study is to evolve a minimal sensor suite, which allows for the most efficient task-specific control. The experimental task requires the MAV to navigate to a specified target location, while avoiding collision with obstacles. Previously the coevolution was performed using two cooperating genetic algorithm-based systems, SAMUEL [8] and GENESIS [7]. The current study considers alternative coevolutionary models as well as alternative controller architectures in order to reach a better understanding of the domain and the algorithms, which will guide future research. The single- and multiple-species coevolutionary models are presented as alternative ways of coevolving form and function. The discussed controller architectures include a rulebase controller based on the SAMUEL learning system and a neural network controller based on the ECkit's [19] implementation of multi-layered feed-forward neural networks.

The remainder of this paper briefly outlines the related work and then describes in details our implementation of coevolution of the behaviors and the characteristics of a sensor suite that would allow the MAV to perform collision-free navigation with maximum efficiency. The simulated environment, aircraft, and sensors are described along with the details of the two controllers and the learning systems. Finally, current results are presented, and the future direction of the research is outlined.

2. Related Work

Evolutionary algorithms have been successfully applied to automate the design of robots' morphology as well as the design of the controllers, but the concept of

coevolution of form and function has surfaced only recently.

There has been a great deal of work done in the area of evolution of function for autonomous vehicles. Behaviors have been evolved using a variety of representations such as neural-networks or rule bases, for a variety of tasks including collision-free navigation [17], [22], exploration [9], as well as shepherding [23], [18], and docking and tracking, just to mention a few. While most of the work is done is simulation, the same behaviors can be evolved in real world as shown by [5].

In parallel, research is being done in the area of evolution of form. Evolutionary algorithms have been applied to the design of structures assembled out of parts [6], design of aircrafts [11], as well as to the design of sensors such as a compound eye [13] or auditory hardware [14]. [16] presents a framework for the study of sensor evolution in a continuous 2-dimensional virtual world (XRaptor).

Finally, in recent years, work has began on coevolving form and function for autonomous agents. [3] and [4] present continuing research on concurrent evolution of neural network controllers and visual sensor morphologies, for visually guided tracking. [24] presents a system for the coevolution of morphology and behavior of virtual creatures that compete in a physically simulated three-dimensional world. Similar work is presented in [10] where the body and brain of the creatures are evolved using Lindenmayer systems as generative encoding. In [12] a hybrid genetic programming/genetic algorithm approach is presented that allows for evolution of both controllers and robot bodies to achieve behavior-specified tasks. [15] introduces a LEGO simulator that allows the user to coevolve controllers and body plans using an interactive genetic algorithm in simulation before constructing the LEGO robots. [1] presents the comparative study of evolution of a control system given a fixed sensor suite, and coevolution of sensor characteristics (placement and range) and the control architecture for the task of box pushing.

The work presented in this paper is related to the above work, but differs in several aspects. This study looks at different models of cooperative coevolution as well as the control architectures in hope of achieving a better understanding of the coevolutionary requirements for this domain. The majority of the previous work involved evolution of neural controllers; our approach looks at evolution of stimuli-response rules as well. The sensors' characteristics initially evolved include the number of sensors and the beam width, with the future possibility of evolution of range and explicit placement of each sensor. Also, even though the evolution is performed in simulation, the simulator closely models the real aircraft and its environment. Finally, the control behaviors are not evolved in a specific setup of an environment as in [1], [15], and [12], but rather each single trial is performed in a randomly and dynamically created environment in order to improve generality of the evolved solutions.

3. Evolution of Sensor Design and Control for MAV

The objective of the study is to evolve a sensor suite with a minimal number of sensors, which allows for the most efficient task-specific control. This section gives an overview of the system architectures used to coevolve the sensor characteristics and the control of the MAV whose task is a collision-free navigation to a specified target location.

In [2] the learning system used for coevolution of form and function was composed of two cooperating genetic algorithm-based systems, SAMUEL and GENESIS. SAMUEL evolved the stimuli-response rules to control the MAV, while GENESIS was used to evolve characteristics of the sensors for the aircraft. The two systems created a loop in which the output from one learning system is the input to the other one. For each member of the population being evaluated by GENESIS representing a specific sensor configuration, SAMUEL had to evolve the best collision-free navigation behavior. Due to the inefficiency of the implementation of this architecture, the need arose for alternative architectures. The single- and multiple-species coevolutionary models were considered for this study.

3.1 Single-Species Coevolution

In a single-species coevolutionary model for coevolution of form and function, the individual (chromosome) in the population, contains the genetic material describing the information of both the morphology and the control behavior of the autonomous agent. During each generation, each individual in the population is evaluated in turn based on its task performance and quality of the morphology, and then children solutions are produced using evolutionary operators such as mutation and crossover. This cycle is performed until a satisfactory solution is found or the evolution stagnates. In this model, only the evaluations can be performed in parallel.

In this work, the single-species coevolutionary model has been used to coevolve form and function with a rulebase controller based on the SAMUEL learning system and with a neural network controller implemented using ECkit libraries. The chromosome in the population contains a floating-point vector, which describes the sensor suite of the MAV, and a set of stimulus-response rules in case of SAMUEL controller or a vector of

floating-point values representing weights of a neural network, which implement the collision-free navigation behavior. The results of the experiments using this coevolutionary model for both controller types are presented in Section 6.1.

3.2 Multiple-Species Coevolution

Our multiple-species coevolutionary model is based on the model of the cooperative coevolution [20]. In this model as applied to the coevolution of form and function, the genetic material describing the morphology and the control behavior is decomposed into separate species. The individual in one population contains the genetic material describing the morphology of the agent while the individual in the other population contains genetic material of a control behavior. Each population is evolved separately, but in terms of the same global fitness function that is based on the performance of the task and the quality of the morphology of the agent. The evolutionary cycle for each species is the same as for the population in the single-species model except that the member of one population is evaluated in terms of the best behavior (as defined by fitness) of the other species. Such decomposition of the problem allows for better understanding of the problem and simplifies the search space. Also, in this model both the learning and the evaluations can be parallelized.

Currently, the multiple-species coevolutionary model has only been used to coevolve form and function with a neural network controller implemented using ECkit. The first population contains individuals, implemented as floating-point vectors, whose genetic material describes the sensor suite of the MAV. The second population contains individuals that implement the collision-free navigation behavior as a two-layer feed-forward neural network. The results of the experiments using this coevolutionary model for a neural network controller are presented in Section 6.2.

3.3 Fitness Function

The morphology of the sensor suite and the control behavior of the MAV are evolved in simulation. During each evaluation, a number of episodes are performed that begins with placement of the MAV at a random distance away from the target facing in a random direction, which is followed by a random placement of trees in the environment. The episodes end with either a successful arrival of the MAV at the target location, a loss of the MAV due to energy/time running out, or a loss of the MAV due to collision with an obstacle. The fitness of the individual is based on the quality of the sensor suite and execution of the task and is defined as follows:

if (got to goal)
  payoff is based on
    the distance MAV traveled (see Section 5.3.3)
    PLUS
    the quality of the sensor suite (see Section 4.3.3)
else if (crashed or run out of time)
  payoff based on
    the distance away from target (see Section 5.3.3)

It should be noted that the contribution due to the quality of the sensor suite is considered only once the task performance is satisfactory and that payoff is only assigned once the episode has been completed.

The following sections will discuss the details of the evolution of form (Section 4.0) and function (Section 5.0). The goals of each learning task will be reviewed, followed by implementation details and a short description of the learning method, the representation and the specific fitness function used.

4. EVOLUTION OF FORM

In this section, the details of the MAV's sensor suite configuration and its evolution are discussed.

4.1 Problem Description

There are a wide variety of sensors that could be implemented on the MAV, but the final make up of the sensor suite is constrained by the size, weight, and power capacity of the vehicle. The objective of this study is to evolve a most power-efficient sensor suite that guarantees an efficient task-specific control. Power efficiency is assumed for this study to be inversely proportional to sensor coverage (beam width and range).

4.2 Problem Representation

The model (Figure 1) of the range sensor is based on a simple range sensor. It returns the range to the closest obstacle in its field of view. The evolvable sensor characteristics include:

1. range of the individual sensor
2. beam width of the individual sensor
3. placement of individual sensor on the vehicle

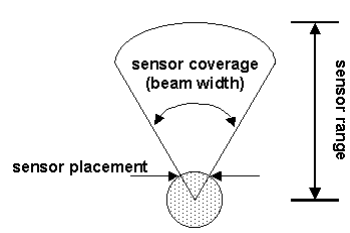

Figure 1. Sensor Model.

In this study, only the number and the beam width of each of the sensors are being evolved. The number of sensors is evolved implicitly since values of beam width and/or range equal to zero imply that the sensor doesn't exist. Nine sensors are placed symmetrically along the direction of flight in increments of 22.5 degrees with the maximum sensor range of 200.0 tenths of feet.

4.3 Implementation of Evolution

4.3.1 Representation. The sensor suite characteristics are represented as a vector of nine floating-point values each in [0 .. 1] range. Each gene value is mapped to 0 to 45 degrees range that defines the beam width of the sensor.

4.3.2 The Learning Method. The basic genetic operators, mutation and crossover, are independent of the coevolutionary model used to evolve the characteristics of the sensor suite of the MAV as well as the controller being used. In all cases, a Gaussian mutation (mu = 0 and sigma = [0.01 .. 0.15 .. 0.2]) and a two-point crossover are used. Currently, the selection operator used in the evolution of the sensor suite characteristics, is specific to a technique used to evolve the controller.

SAMUEL uses standard genetic algorithms and other competition-based heuristics to evolve the solution. It specifically uses a fitness-proportional selection method to choose the individuals out of the population, which means that the number of offspring is proportional to the parent's fitness. Also, the sigma of the Gaussian mutation of genes is fixed at 0.15 during learning.

The evolutionary technique chosen from the ECkit library for evolution of the MAV's sensor suite is a (μ , λ) evolution strategy (μ equal to 10 and λ to 100) [21]. The sigma of the Gaussian mutation operator is evolved along with the individuals, but it cannot be higher than 0.2.

4.3.3 Fitness Function Contribution. The fitness of the sensor suite is inversely proportional to its coverage and contributes [0.0 .. 0.2] to the global fitness functions, but only if the agent behavior allows it to complete the task, i.e.: navigate safely to the target location. The contribution is calculated as follows:

$$f_{FORM}(x) = 0.2 * (1.0 - (C(x) / C_{EXP}))$$

where x is an individual or part of the individual whose genetic code contains only the information on the characteristics of the sensor suite; C(x) is the coverage of the sensor suite calculated as the sum of the beam widths of individual sensor; and C_{EXP} is the maximum possible sensor coverage for the experiment; C_{EXP} is currently equal to 405.0 (9 * 45.0).

5. Evolution of Function

In this section, the details of the MAV's control task and its evolution are discussed. Experimental details of the simulated environment, aircraft, and sensors are provided along with the details of the learning systems used.

5.1 Problem Description

The MAV must be able to efficiently and safely navigate among obstacles (trees) to a target location. The desired behavior should maximize the number of times the MAV reaches the target location while minimizing the distance traveled to that location. The generality of evolved control should be ensured due to a random setup of the environment for every evaluation.

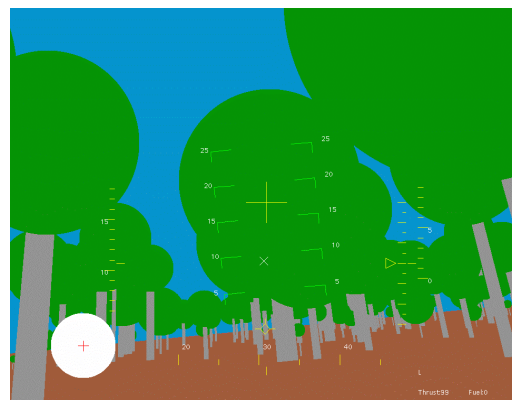

Figure 2. The screenshot of the 3-D simulated environment used for the experiments. The white sphere marks the target and dark gray (or green) spheres with light gray cylinders mark the obstacles (trees).

5.2 Problem Representation

5.2.1 Environment. The world as well as the aircraft itself is modeled in a high fidelity, 6-DOF flight simulator (Figure 2), which includes an accurate parameterized model of a 6-inch MAV and a model of the task environment. Although the simulator allows for accurate modeling of sensor noise, winds, and wind gusts, this initial study does not take advantage of these capabilities. The low-level control for the MAV is implemented using a number of PID controllers, which allow the user to control the aircraft by specifying only the turn rate values; the PID controllers adjust speed and altitude of the plane appropriately. The trees (obstacles) are modeled as spheres on top of cylinders in order to decrease the computational complexity of the environment. Any

contact between the plane and the tree constitutes a collision. The density of trees is user-defined as a number of trees per square foot assuming uniform distribution and was set to 2.5 trees per hundred square feet. At the beginning of each simulated flight, the MAV is placed in a random location within a specified area away from the target. The target is stationary and reachable during every trial.

5.2.2 Sensors. It is assumed that the MAV has a sensor, which returns the relative range and bearing to the target. Also, the aircraft is equipped with a number of range sensors. Each sensor is capable of detecting obstacles and returning the range to the closest object within its field of view. The exact makeup of the sensor suite is evolved as described in Section 4.3.

5.2.3 Actions/Effectors. There is a discrete set of actions available to control the MAV. In this study, the only action that is considered specifies discrete turning rates for the MAV. The control variable turn_rate is between –20 and 20 degrees in increments dependent on the learning method used. As mentioned, the altitude and the speed of the plane are adjusted as necessary by underlying PID controllers.

5.3 Implementation of Evolution

Due to the choice of the controllers for this study, the methods for evolution of the control behavior for the MAV for the task of collision-free navigation are architecture dependent. The following sections discuss the details of the evolution for both, the rulebase controller evolved using SAMUEL learning system and a neural network controller evolved using ECkit.

5.3.1 Rulebase Controller. SAMUEL implements behaviors as a collection of stimulus-response rules. Each stimulus-response rule consists of conditions that match against the current sensors of the autonomous vehicle, and an action that suggests action to be performed by it. An example of a rule (gene) might be:

RULE 122
IF bearing = [-20, 20] AND
range4 < 45
THEN SET turn_rate = -100

Each rule has an associated strength with it as well as a number of other statistics. During each decision cycle, all the rules that match the current state are identified. Conflicts are resolved in favor of rules with higher strength. Rule strengths are updated based on rewards received after each training episode. For this study, SAMUEL uses the following sensors:

- range1 .. range9: Value between [5 .. 200] in 10 tenths of feet increments specifies the distance to the closest obstacle within sensor's field of view.
- range: Value between [5 .. 2000] in 20 tenths of feet increments specifies the distance to the target.
- bearing: Value between [-180 .. 180] in 45 degree increments specifies the bearing to the target.

The action parameter, turn_rate, specifies the turn rate for the MAV in the range [-20 .. 20] in 5 degree increments.

The system must learn a behavior for navigating the MAV to the target location while avoiding obstacles. The behaviors, which are represented as a collection of stimulus-response rules, are learned in the SAMUEL rule

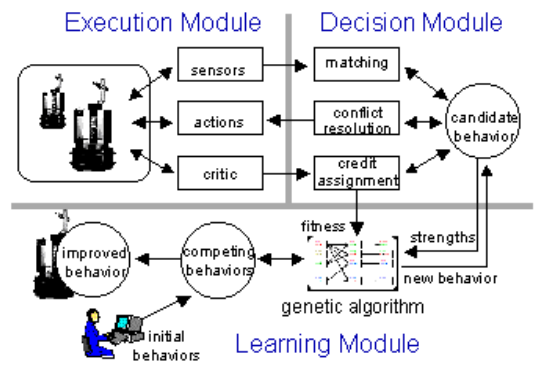

Figure 3. SAMUEL Learning System.

learning system (Figure 3).

SAMUEL uses standard genetic algorithms and other competition-based heuristics to solve sequential decision problems. It features Lamarckian operators (specialization, generalization, merging, avoidance, and deletion) that modify decision rules on the basis of observed interaction with the task environment. SAMUEL has to perform a number of evaluations in order to provide history for Lamarckian operators, to coalesce rule strengths, and to account for the noise in the evaluations. The original system implementation is described in greater detail in [8].

5.3.2 Neural Network Controller. The ECkit library [Potter WWW] contains various representations for organisms, members of the species under evolution. For this study, organisms use a floating-point vector representation. The organisms contain the genetic code, which describes the connection weights of the neural network controller (Figure 4), which implements the collision-free navigation behavior.

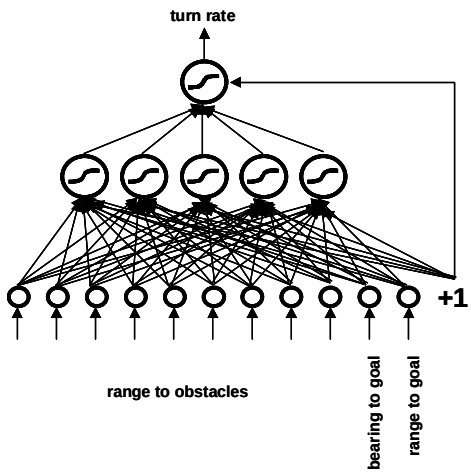

Figure 4. Neural network controller.

For this study, the MAV's controller is implemented as a two-layer feed-forward neural network. There are 11 input nodes (9 range sensors, bearing and range to the target), 5 hidden nodes, and one output node. All hidden nodes and the output node use a standard sigmoid trigger function. The output of the controller is mapped to the range [-20 .. 20] in 1-degree increments and defines the turn rate of the MAV. The network is fully connected and each hidden and output node has a bias associated with it, hence the floating-point vector contains (I+1)*H + (H+1)*O values in range [-MAX_DOUBLE, MAX_DOUBLE], where I is the number of the inputs, H is the number of the hidden nodes, and O is the number of the output nodes.

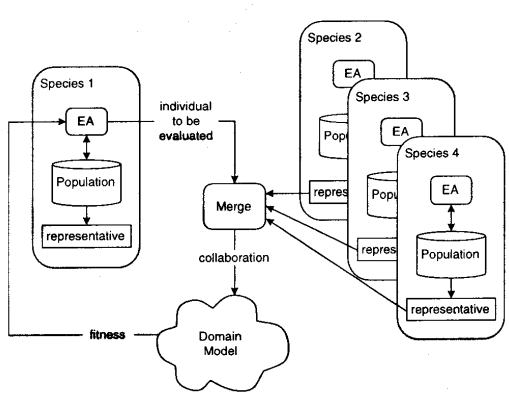

Figure 5. Coevolutionary model of ECkit.

ECkit is based on the multiple-species coevolutionary model [20] as briefly described in Section 3.2 and as shown in Figure 5. Each species is evaluated based on the global fitness function in terms of the representative, in this case the best individual, from all the other species in the ecosystem. ECkit provides the user with a variety of evolutionary operators whose parameters can be tuned for the application. The user is required to implement the domain for the ecosystem. Details of the system implementation are described in [19].

The same evolutionary algorithm was chosen to evolve the neural network controller for collision-free navigation as for evolution of sensor suite characteristics, that is a (μ , λ) evolution strategy (μ equal to 10 and λ to 100) with Gaussian mutation (mu = 0 and adaptive sigma = [0.01 .. 1.0]) and two-point crossover. To account for the noise in the evaluations, a number of trials are performed during each evaluation.

5.3.3 Fitness Function Contribution. The fitness of the controller is proportional to the distance MAV traveled during the successful trial or the minimum distance away from the target during an unsuccessful trial, and contributes [0.0-0.3 .. 0.5-0.8] to the global fitness functions. The contribution is calculated as follows:

$$f_{FUNC}(x) = \begin{cases} 0.8 * (1.0 - D_S/D(t)), \text{ if successful trial} \\ 0.3 * (1.0 - D_A/D_S), \text{ if unsuccessful trial} \end{cases}$$

where D_S is a initial distance away from the target, D(t) is total distance traveled during the trial, and D_A is the minimum distance away from the target during the trial.

6. Current Results

In this section, the results of the experiments performed are presented. The results are discussed in terms of the internal fitness function (Section 3.3) as well as the external performance, which is defined as number of times the MAV arrived at the goal. The quality of the evolved solutions is also evaluated in harder and easier environments, to get a feel for their ability to generalize. A random behavior (random turn rate values) failed to perform the task under all the conditions considered.

6.1 Single-Species Coevolution

This section describes the single-species approach to coevolution of form and function for rulebase and neural network controllers.

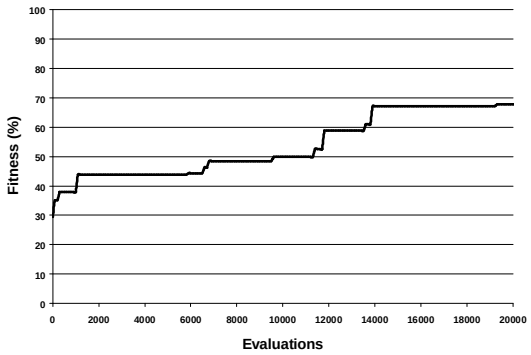

Figure 6. Best-so-far internal fitness (average of 100 evaluations) curve for coevolution of form and function using a single-species model based on SAMUEL.

6.1.1 SAMUEL Controller. The learning curve, plotting the internal fitness against the number of evaluations, is shown in Figure 6. Given the simplicity of seeding SAMUEL with initial heuristic rules, an initial population was created which consisted of simple hand-coded rule sets such as random walk, emergency obstacle avoidance, and going towards the goal. All the initial sensor suites contained 9 sensors each with 45-degree beam width. To obtain a better estimate of the solutions fitness in face of high variance, the fitness was averaged over 100 trials. The initial solution as described above, obtained a 29.18% level in terms of internal fitness with external performance around 23%. The best evolved individual (generation 192) had internal fitness of 67.73% and 68% external performance. The evolved sensor suite had eight out of the nine sensors (no sensor at 67.5 degree location) with total beam width of 104.8 degrees. The results for this experiment are summarized in Table 1.

	Fitness		# of sensors (total coverage)
	Internal	External	
Initial	29.2%	23%	9 (405)
Best	67.7%	68%	8 (104.8)

Table 1. Summary of the results for the single-species coevolutionary model with rulebase controller. The internal fitness, external performance (averaged over 100 runs), and the characteristics of the sensor suite are shown for all conditions.

The quality of the solution was also evaluated in simpler and more complex environments to see how well it generalizes. In the simpler environment (approx. 1.25 trees per hundred square feet), the solution obtained internal fitness of 80.6% and external performance of 84%. In the more complex environment (approx. 5 trees per hundred square feet), the solution's internal fitness decreased to 39.9% and external performance to 33%. These results suggest the solution's ability to generalize.

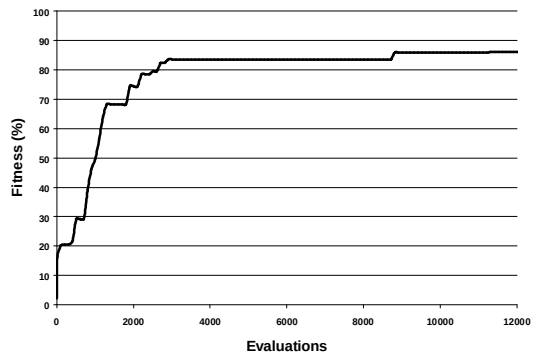

Figure 7. Best-so-far internal fitness (average of 10 evaluations) curve for coevolution of form and function using a single-species model based on neural network controller evolved using evolutionary strategy in ECkit.

6.1.2 Neural Network Controller. The learning curve, plotting the internal fitness against the number of evaluations, is shown in Figure 7. The initial population of neural network controllers was randomly initialized, as were the sensor suites. Due to time constraints and inability to currently parallelize the evolution of the neural network controller, each member of the population was evaluated only 10 times, which given the high variance of the internal fitness function, introduced discrepancy between the internal fitness as seen by the learning algorithm and the actual internal fitness of the solution. Since the quality of learning performed by the evolutionary algorithm was based on the internal fitness, but the actual fitness allows for better comparison to external performance, both the internal fitness and the actual fitness of the solutions are reported. A randomly generated solution obtained internal fitness of 19.6% while in fact it's value (averaged over 100 runs) was only 2.27% with an external fitness of 0%. The best evolved solution (generation 113) had an internal fitness of 86.04%, better estimated at 43.5%, and was able to safely navigate the MAV to the target 46% of the time. The evolved senor suite consisted of all nine sensors with total beam coverage of 159.2 degrees. The results for this experiment are summarized in Table 2.

As before, the quality of the best individual was evaluated in simpler and more complex environments to see how well it generalizes. In the simpler environment,

the solution obtained internal fitness of 70.63% and an external performance of 82%. In the more complex environment, the solution's internal fitness decreased to 23.2% and external performance to 17%. These results again show that the solution is most likely able to generalize.

	Fitness		# of sensors (total coverage)
	Internal (Actual Internal)	External	
Initial	19.61% (2.27%)	0.00%	9 (215.6)
Best	86.04% (43.48%)	46%	9 (159.2)

Table 2. Summary of results for the single species experiments with a neural network controller. The internal fitness as seen by the learning algorithm is given as well as the fitness of solution averaged over 100 trials, the external fitness, and the make up of the sensor suite are shown.

6.2 Multiple-Species Coevolution

This section describes the multiple-species approach to coevolution of form and function for currently only a neural network controller. Implementation of the multiple-species coevolutionary model to be used with SAMUEL is underway.

6.2.1 Neural Network Controller. The learning curve, plotting the internal fitness against the number of evaluations, is shown in Figure 8. Both, the behavior and the sensor suite populations were initialized with random individuals. As in the previous experiment (Section 6.1.2), inadequate number of evaluations (only 10), introduced discrepancy between the internal fitness and the actual value of the solution. The baseline for this experiment was the same as in the single-species coevolution with neural network controller (Section 6.1.2). A randomly generated solution obtained internal fitness of 19.6% with more accurate estimate of 2.27% and no ability to reach the goal. The best evolved solution (generation 186) had a 87.14% internal fitness, which was closer to 52.7% with external performance at 53%. The evolved sensor suite makes use of eight out of nine sensors (no sensor at 67.5 degree location) with a total beam width of 174.7 degrees. The results for this experiment are summarized in Table 3.

The quality of the solution was again evaluated in simpler and more environments to see how well it

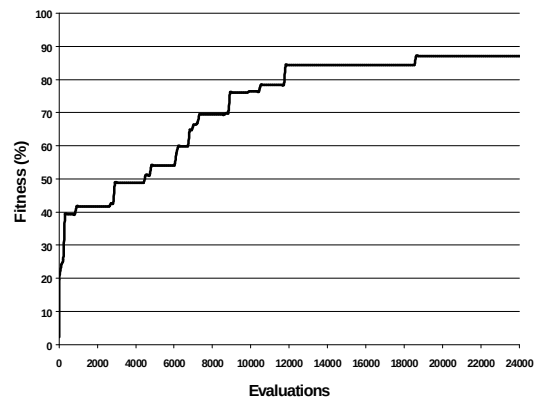

Figure 8. Best-so-far internal fitness (average of 10 evaluations) curve for coevolution of form and function using a multiple-species model based on neural network controller evolved using evolutionary strategy in ECkit.

generalizes. In the simpler environment (approx. 1.25 trees per hundred square feet), the solution obtained internal fitness of 68.1% and external performance of 75%. In the more complex environment, the solution's internal fitness decreased to 29.14% and external performance to 33%. Those results show solution's aptitude for generalization.

	Fitness		# of sensors (total coverage)
	Internal (Actual Internal)	External	
Initial	19.61% (2.27%)	0.00%	9 (215.6)
Best	87.14% (52.69%)	53%	8 (159.3)

Table 3. Summary of results for the multiple species experiments with a neural network controller. The internal fitness as seen by the learning algorithm is given as well as the fitness of solution averaged over 100 trials, the external fitness, and the make up of the sensor suite are shown.

7. Conclusions and Future Work

This paper discussed approaches to the cooperative coevolution of form and function for autonomous vehicles, specifically evolving the morphology (the sensor suite) and the control (goal seeking and collision avoidance behaviors) for an autonomous micro air

vehicle. This research is significant, because it can result in more efficient synergistic designs of autonomous vehicles.

Alternative models of cooperative coevolution were presented, including single- and multiple-species models, for two different control architectures, a rulebase controller evolved using the SAMUEL learning system and a neural network controller evolved and implemented using ECkit. Experimental results were presented demonstrating that both models and both control architectures could learn to coevolve a minimal sensor suite and corresponding behaviors, and that the resulting evolved systems were tolerant to changes in environment complexity.

Once the implementation of the multiple-species coevolutionary model combined with the SAMUEL learning system is complete, additional data will be collected in order to establish statistical significance of the experimental results. Given that data, it should be possible to draw some conclusions about the preferred representation and coevolutionary model for this domain. In the follow-up work, additional characteristics of the sensor suite such as explicit placement of the sensors on the airframe body and the ranges of the sensors, will be evolved.

While this study has specifically emphasized the coevolution of *sensors* and control, this general methodology is also applicable to design parameters of the vehicle structure. In future work, other aspects of the parametric model that define the vehicle platform, specifically ones that will result directly in design decisions for the airframe structure, will be considered. Aspects of the airframe can be optimized for classes of missions and expected behaviors. Future work might also consider reconfigurable hardware to allow for changes in the system as missions change over time. Effects of sensor noise and variability in the environmental conditions such as wind speed and direction on the evolved system will be considered as well.

Acknowledgments

The work reported in this paper was supported by the Office of Naval Research under work order N0001402ZWX30005.

References

[1] Balakrishnan, K. and V. Honovar. "On Sensor Evolution in Robotics." In Koza, Goldberg, Fogel, and Riolo (eds.), Proceedings of 1996 Genetic Programming Conference – GP-96; MIT Press, pp. 455-460; 1996.

[2] Bugajska, Magdalena D. and Alan C. Schultz. "Co-Evolution of Form and Function in the Design of Autonomous Agents: Micro Air Vehicle Project." GECCO-2000 Workshop on Evolution of Sensors in Nature, Hardware, and Simulation, Las Vegas, NV; July 8, 2000.

[3] Cliff, D., I. Harvey, and P. Husbands "Explorations in Evolutionary Robotics", *Adaptive Behaviour* (MIT Press) 2(1):71--108, 1993.

[4] D. Cliff and G. F. Miller "Co-evolution of Pursuit and Evasion II: Simulation Methods and Results" in: P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. Wilson (editors) *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (SAB96)*. MIT Press Bradford Books, pp.506--515, 1996.

[5] Floreano, D. and F. Mondada. "Evolution of Homing Navigation in a Real Mobile Robot." In IEEE Transactions on System, Man, and Cybernetics – Part B; 26(3) 396-407; 1996.

[6] Funes, P. and J. Pollack. "Computer Evolution of Buildable Objects." Husbands and Harvey (eds.), Proceedings of The Fourth European Conference on Artificial Life, MIT Press, pp.358-367; 1997.

[7] Grefenstette, J. J. "The User's Guide To GENESIS." Technical Report CS-83-11, Computer Science Department, Vanderbilt University, Nashville, TN; 1984.

[8] Grefenstette, J. J. "The User's Guide to SAMUEL, Version 1.3." NRL Memorandum Report 6820, Naval Research Laboratory, 1991.

[9] Harvey, I., P. Husbands, and D. Cliff. "Issues in Evolutionary Robotics." Proceedings of the Second International Conference on Simulation of Adaptive Behaviour; MIT Press Bradford Books, 1993.

[10] Hornby, G. S. and J. B. Pollack (2001). "Body-Brain Co-evolution Using L-systems as a Generative Encoding." In Proceedings of the Genetic and Evolutionary Computation Conference 2001, San Francisco, CA; pp. 868-875.

[11] Husbands, P., G. Jermy, M. McIlhagga, and R. Ives. "Two Applications of Genetic Algorithms to Component Design." In selected papers from AISB Workshop on Evolutionary Computing; Fogarty, T (ed.), Springer-Verlog; Lecture Notes in Computer Science, 1996.

[12] Lee, Wei-Po, J. Hallam, and H. H. Lund. "A Hybrid GP/GA Approach for Co-evolving Controllers and Robot Bodies to Achieve Fitness-Specific Tasks." In Proceedings of IEEE Third International Conference on Evolutionary Computation, IEEE Press, NJ,;1996.

[13] Lichtensteiger, L. and P. Eggenberger. "Evolving the Morphology of a Compound Eye on a Robot." Proceedings of the Third European Workshop on Advanced Mobile Robots (Eurobot '99); 1999.

[14] Lund, H. H., J. Hallam, and W-P. Lee. "Evolving Robot Morphology." Proceedings of IEEE Fourth International Conference on Evolutionary computation; IEEE Press, NJ, 1997.

[15] Lund, H. H. and O. Miglino. "Evolving and Breeding Robots." In Proceedings of First European Workshop on Evolutionary Robotics; Springer-Verlag, 1998.

[16] Mark, A., D. Polani, and Thomas Uthmann. "A Framework for Sensor Evolution in a Population of Braitenberg Vehicle-like Agents." In C. Adam, R. Belew, H. Kitno, and C. Taylor (eds.), Proceedings of Artificial Life IV; pp. 428-432; 1998.

[17] Nolfi, S., Floreano, D., Mighno, O., and F. Mondada. "How to Evolve Autonomous Robots: Different Approach in Evolutionary Robotics." In R. Brooks and P. Maes (eds.), Proceedings of the International Conference Artificial Live IV; MIT Press, pp.190-197; 1994.

[18] Potter, Mitchell A., Lisa A. Meeden, and Alan C. Schultz. "Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergence of Specialists." In Proceedings of The Seventeenth International Conference on Artificial Intelligence, Seattle, WA; August 4-10, 2001.

[19] Potter, Mitchell A. "Evolutionary Computation Toolkit" [online]. Fairfax, Virginia: George Mason University, Department of Information Technology and Engineering, 1999 [cited March 5, 2002]. Available from World Wide Web at <http://cs.gmu.edu/~mpotter/eckit>.

[20] Potter, M. A. "The Design and Analysis of a computational Model of Cooperative Coevolution." Ph.D. Thesis, George Mason University, Fairfax, VA; 1997.

[21] Rechenberg, I. Evolutionsstrategie – Optimierung technisher Systeme nach Prinzipien der biologischen. Stuttgart-Bad Cannstatt: Frommann-Holzboog; 1973.

[22] Schultz, A. C. "Using a Genetic Algorithm to Learn Strategies for Collision Avoidance and Local Navigation." Proceedings of the Seventh International Symposium on Unmanned Untethered Submersible Technology, University of New Hampshire Marine Systems Engineering Laboratory, pp. 213-215; 1991.

[23] Schultz, A. C., J. J. Grefenstette, and William Adams. "RoboShepherd: Learning a complex behavior." Presented at RobotLearn96: The Robotics and Learning workshop at FLAIRS ’96; 1996.

[24] Sims, K. "Evolving 3D Morphology and Behavior by Competition." In R. Brooks and P. Maes (eds.), Proceedings of the International Conference Artificial Live IV; MIT Press, Cambridge, MA, pp.28-39; 1994.