Autonomous Controller Design for Unmanned Aerial Vehicles using Multi-objective Genetic Programming

Gregory J. Barlow

Center for Robotics and Intelligent Machines, Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695-7911

Abstract. Autonomous navigation controllers were developed for fixed wing unmanned aerial vehicle (UAV) applications using multi-objective genetic programming (GP). Four fitness functions derived from flight simulations were designed and multi-objective GP was used to evolve controllers able to locate a radar source, navigate the UAV to the source efficiently using on-board sensor measurements, and circle around the emitter. Controllers were evolved for three different kinds of radars: stationary, continuously emitting radars, stationary, intermittently emitting radars, and mobile, continuously emitting radars. In this study, realistic flight parameters and sensor inputs were selected to aid in the transference of evolved controllers to physical UAVs.

1 Introduction

The field of evolutionary robotics (ER) [1] combines research on behavior-based robot controller design with evolutionary computation. A major focus of ER is the automatic design of behavioral controllers with no internal environmental model, in which effector outputs are a direct function of sensor inputs. Most of the controllers evolved in ER research to date have been developed for simple behaviors, such as obstacle avoidance, light seeking, object movement, simple navigation, and game playing. In many of these cases, the problems to be solved were designed specifically for research purposes. While simple problems generally require a small number of behaviors, more complex real-world problems might require the coordination of multiple behaviors in order to achieve the goals of the problem. Very little of the ER work to date has been intended for use in real-life applications.

This paper presents an approach to evolving behavioral navigation controllers for fixed wing UAVs using multi-objective GP. Controllers should be able to locate a radar, navigate the UAV to the source efficiently using sensor measurements, and circle closely around the radar. Controllers were evolved for three different radar types. Despite success in evolving controllers directly on real robots [1], simulation is the only feasible way to evolve controllers for UAVs. A UAV cannot be operated continuously for long enough to evolve a sufficiently competent controller, the use of an unfit controller could result in damage to the aircraft, and flight tests are very expensive. For these reasons, simulation must be capable of evolving controllers which transfer well to real UAVs.

2 Unmanned Aerial Vehicle Simulation

The focus of this research was the development of navigation controllers for fixed wing UAVs. The objective is to autonomously locate, track, and then orbit around a radar site. There are three main goals for an evolved controller. First, it should move to the vicinity of the radar as quickly as possible. The sooner the UAV arrives in the vicinity of the radar, the sooner it can begin its primary mission, whether that is jamming the radar, surveillance, or another of the many applications of this type of controller. Second, once in the vicinity of the source, the UAV should circle as closely as possible around the source. Third, the flight path should be stable and efficient. The roll angle should change as infrequently as possible, and any change in roll angle should be small. Making frequent changes to the roll angle of the UAV could create dangerous flight dynamics and could reduce the flying time and range of the UAV.

The controller is evolved in simulation. The simulation environment is a square 100 nautical miles (nmi) on each side. The simulator gives the UAV a random initial position in the middle half of the southern edge of the environment with an initial heading of due north and the radar site a random position within the environment every time a simulation is run. In this research, the UAV has a constant altitude and a constant speed of 80 knots. The low level control of the UAV is done by an autopilot; the evolved controllers navigate the UAV.

The simulation can model a wide variety of radar types. For the research presented in this paper, three types of radars were modeled : 1) stationary, continuously emitting radars, 2) stationary, intermittently emitting radars with a period of 10 minutes and emitting duration of 5 minutes, and 3) mobile, continuously emitting radars. Only the sidelobes of the radar emissions are modeled. The sidelobes of a radar signal have a much lower power than the main beam, making them harder to detect. However, the sidelobes exist in all directions, not just the direction the radar is pointed. This model is intended to increase the robustness of the system, so that the controller doesn't need to rely on a signal from the main beam. Additionally, Gaussian noise is added to the amplitude of the radar signal. The receiving sensor can perceive only two pieces of information: the amplitude and the angle of arrival (AoA) of incoming radar signals. The AoA measures the angle between the heading of the UAV and the source of incoming electromagnetic energy. Real AoA sensors do not have perfect accuracy in detecting radar signals, so the simulation models an inaccurate sensor. In the experiments described in this research, the AoA is accurate to within $\pm 10°$ at each time step, a realistic value for this type of sensor. Each experimental run simulates four hours of flight time, where the UAV is allowed to update its desired roll angle once a second. The interval between these requests to the autopilot can also be adjusted in the simulation. Further details about the simulation environment can be found in [2].

3 Multi-objective Genetic Programming

UAV controllers were designed using multi-objective genetic programming which employs non-dominated sorting, crowding distance assignment to each solution, and elitism. The multi-objective genetic programming algorithm used in this research is very similar to the NSGA-II [3] multi-objective genetic algorithm. The function and terminal sets combine a set of very common functions used in GP experiments and some functions specific to this problem. The sensors available to GP measure the amplitude, AoA, and slope of the amplitude for incoming radar signals. When turning, there are six available actions. Turns may be hard or shallow, with hard turns making a 10° change in the roll angle and shallow turns a 2° change. The *WingsLevel* terminal sets the roll angle to 0, and the *NoChange* terminal keeps the roll angle the same. Multiple turning actions may be executed during one time step, since the roll angle is changed as a side effect of each terminal. The final roll angle after the navigation controller is finished executing is passed to the autopilot. The maximum roll angle is 45°.

Genetic programming was generational, with crossover and mutation similar to those outlined by Koza in [4]. A population of 500 individuals was evolved for 600 generations. The crossover rate was 0.9 and the mutation rate was 0.05. Tournament selection with a tournament size of 2 was used. Initial trees were randomly generated using ramped half-and-half initialization. The maximum initial depth was 5, and the maximum depth was 21.

4 Fitness Functions

Four fitness functions determine the success of individual UAV navigation controllers. The fitness of a controller was measured over 30 simulation trials, where the UAV and radar positions were different for every run. The four fitness measures were designed to satisfy the three goals of the evolved controller: moving toward the emitter, circling the emitter closely, and flying in an efficient way.

The primary goal of the UAV is to fly from its initial position to the radar site as quickly as possible. The controllers' ability to accomplish this task is measured by averaging the squared distance between the UAV and the radar over all time steps. This distance is normalized using the initial distance between the radar and the UAV in order to mitigate the effect of varying distances from the random placement of radar sites. The normalized distance fitness measure is given as $fitness_1 = \frac{1}{T}\sum_{i=1}^{T}\left[\frac{distance_i}{distance_0}\right]^2$, where T is the total number of time steps, $distance_0$ is the initial distance, and $distance_i$ is the distance at time i. The goal is to minimize $fitness_1$.

Once the UAV has flown in-range of the radar, the goal shifts from moving toward the source to circling around it. An arbitrary distance much larger than the desired circling radius is defined as the in-range distance. For this research, the in-range distance was set to be 10 nmi. The circling distance fitness metric measures the average distance between the UAV and the radar over the time the UAV is in-range. While the circling distance is also measured by $fitness_1$, that metric is dominated by distances far away from the goal and applies very little evolutionary pressure to circling behavior. The circling distance fitness measure is given as $fitness_2 = \frac{1}{N}\sum_{i=1}^{T} in\_range * (distance_i)^2$, where N is the amount of time the UAV spent within the in-range boundary of the radar and *in_range* is 1 when the UAV is in-range and 0 otherwise. The goal is to minimize $fitness_2$.

In addition to the primary goals of moving toward a radar site and circling it closely, it is also desirable for the UAV to fly efficiently in order to minimize flight time to get close to the goal and to prevent potentially dangerous flight dynamics, like frequent and drastic changes in the roll angle. The first fitness metric that measures the efficiency of the flight path is the amount of time the UAV spends with its wings level to the ground, which is the most stable flight position for a UAV. This fitness metric only applies when the UAV is outside the in-range distance, since once the UAV is within the in-range boundary, is should circle around the radar. The level time is given as $fitness_3 = \frac{1}{T-N}\sum_{i=1}^{T}(1 - in\_range) * level$, where *level* is 1 when the UAV has been level for two consecutive time steps and 0 otherwise. The goal is to maximize $fitness_3$.

The second fitness measure intended to produce an efficient and stable flight path is a measure of turn cost. While UAVs are capable of very quick, sharp turns, it is preferable to avoid them. The turn cost fitness measure is intended to penalize controllers that navigate using a large number of sharp, sudden turns because this may cause very unstable flight. The UAV can achieve a small turning radius without penalty by changing the roll angle gradually; this fitness metric only accounts for cases where the roll angle has changed by more than 10° since the last time step. The turn cost is given as $fitness_4 = \frac{1}{T}\sum_{i=1}^{T} h\_turn * |roll\_angle_i - roll\_angle_{i-1}|$, where *roll_angle* is the roll angle of the UAV and *h_turn* is 1 if the roll angle has changed by more than 10° since the last time step and 0 otherwise. The goal is to minimize $fitness_4$.

5 Results

Multi-objective GP produced controllers that satisfied the three goals of this problem. In order to statistically measure the performance of GP, 50 evolutionary runs were done for each type of radar. Each run lasted for 600 generations and produced 500 solutions. Since multi-objective optimization produces a Pareto front of solutions, rather than a single best solution, a method to gauge the performance of evolution was needed. To do this, values considered minimally successful for the four fitness metrics were selected. A minimally successful UAV controller is able to move quickly to the target radar site, circle at an average distance under 2 nmi, fly with the wings level to the ground for at least 1,000 seconds, and turn sharply less than 0.5% of the total flight time. If a controller had a normalized distance fitness value ($fitness_1$) of less than 0.15, a circling distance ($fitness_2$) of less than 4 (the circling distance fitness metric squares the distance), a level time ($fitness_3$) of greater than 1,000, and a turn cost ($fitness_4$) of less than 0.05, the evolution was considered successful. These baseline values were used only for analysis, not for the evolutionary process.

Controllers were evolved for 1) stationary, continuously emitting radars, 2) stationary, intermittently emitting, radars, and 3) mobile, continuously emitting radars. More complete results of these experiments can be found in [2] and [5].

The first experiment evolved controllers on a stationary, continuously emitting radar. Of the 50 evolutionary runs, 45 runs were acceptable under the baseline values. The number of acceptable controllers evolved during an individual run ranged from 1 to 170. Overall, 3,149 acceptable controllers were evolved, for an average of 62.98 successful controllers per evolutionary run. The best evolved controllers fly to the target very efficiently, staying level a majority of the time. Almost all turns are shallow. Once in range of the target, the roll angle is gradually increased. Once the roll angle reaches its maximum value to minimize the circling radius, no change to the roll angle is made for the remainder of the simulation. Populations tended to evolve to favor turning left or right.

The second experiment evolved controllers for a stationary, intermittently emitting radar. The radar was set to emit for 5 minutes and then turned off for 5 minutes, giving a period of 10 minutes and a 50% duty cycle. This experiment was far more difficult for evolution than the first experiment, because the radar only emits half of the time. A new set of 50 evolutionary runs was done, and 25 of the runs produced at least one acceptable solution. The number of controllers in an evolutionary run that met the baseline values ranged from 1 to 156, 1,891 successful controllers were evolved, and the average number of acceptable controllers evolved during each run was 37.82. The flight paths for these controllers were similar to those for the continuously emitting radars. Despite the increased complexity from the first experiment, GP was able to evolve many successful controllers.

The third experiment evolved controllers for a mobile, continuously emitting radar. The mobility was modeled as a finite state machine with the following states: *move*, *setup*, *deployed*, and *tear down*. When the radar moves, the new location is random anywhere in the simulation area. The finite state machine is repeated for the duration of simulation. The radar site only emits when it is in the *deployed* state; while the radar is moving, the UAV receives no sensory information. The time in each state is probabilistic, and the minimum amount of time spent in the deployed state is an hour. Of the 50 evolutionary runs, 36 were acceptable under the baseline values. The number of acceptable controllers evolved in each run ranged from 1 to 206, and 2,266 successful controllers were evolved for an average of 45.32 acceptable controllers per evolutionary run.

To test the effectiveness of each of the four fitness measures, evolutions were done with various subsets of the fitness metrics. These tests were done using the stationary, continuously emitting radar, the simplest of the three radar types presented above. Based on these tests, it was determined that all four fitness functions were necessary to evolve successful controllers. In a comparison, controllers evolved using only the normalized distance fitness function exhibited slightly better performance than a human-designed, rule-based controller.

Flying a physical UAV with an evolved controller is planned as a demonstration of the research, so transference was taken into consideration from the beginning. Several aspects of the controller evolution were designed specifically to aid in this process. First, the navigation control was abstracted from the flight of the UAV. Rather than attempting to evolve direct control, only the navigation was evolved. This allows the same controller to be used for different airframes. Second, the simulation parameters were designed to be tuned for equivalence to real aircraft. For example, the simulated UAV is allowed to update the desired roll angle once per second reflecting the update rate of the real autopilot of a UAV being considered for flight demonstrations of the evolved controller. Third, noise was added to the simulation, both in the radar emissions and in sensor accuracy. A noisy simulation environment encourages the evolution of robust controllers that are more applicable to real UAVs.

6 Conclusions

Genetic programming with multi-objective optimization was used to evolve navigation controllers for UAVs capable of flying to a target radar, circling the radar site, and maintaining an efficient flight path, all while using inaccurate sensors in a noisy environment. Controllers were evolved for three different radar types. The four fitness functions used for this research were sufficient to produce the desired behaviors, and all four measures were necessary for all three cases. Methods were used to aid in the transference of the evolved controllers to real UAVs. In the next stage of this research, controllers evolved in this research will be tested on physical UAVs.

7 Acknowledgments

This work was supervised by Dr. Edward Grant (NCSU) and Dr. Choong Oh (NRL), financial support was provided by the Office of Naval Research, and computational resources were provided by the Naval Research Laboratory.

References

1. Nolfi, S., Floreano, D.: Evolutionary Robotics. MIT Press (2000)
2. Barlow, G.J.: Design of autonomous navigation controllers for unmanned aerial vehicles using multi-objective genetic programming. Master's thesis, North Carolina State University, Raleigh, NC (2004)
3. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6** (2002) 182–197
4. Koza, J.: Genetic Programming. MIT Press (1992)
5. Oh, C.K., Barlow, G.J.: Autonomous controller design for unmanned aerial vehicles using multi-objective genetic programming. In: Proceedings of the Congress on Evolutionary Computation, Portland, OR (2004)