LEARNING TO SOAR

BY MICHAEL ALLEN

On a spring day at Edwards Air Force Base, someone pointed overhead to a flock of migrating white pelicans soaring gracefully in formation. I wondered if the Autonomous Soaring project I had just started would produce an unmanned aerial vehicle (UAV) that could soar as gracefully. Not likely, I thought, as the pelicans soared in perfect unison, optimizing their climb rate in an invisible column of rising air called a thermal or updraft.

Michael Allen stands with the SBXC glider.

The Autonomous Soaring project was initially funded in 2003 with director's discretionary funding. The result of the work done in 2003 was a simulation study using National Oceanic and Atmospheric Administration weather measurements to estimate updraft properties at Desert Rock, Nevada. The study showed that a small UAV could extend its two-hour nominal endurance to fourteen hours during the summer and up to eight hours in the winter. In 2005, with funding from the NASA Flight and System Demonstrations Project, I put together a small team to demonstrate that a small UAV could actually detect and stay within an updraft without human intervention.

While others had written papers and hypothesized about extending UAV flight time by using updrafts to reserve engine power, no one had tested the theory. If our experiment succeeded, we could influence the way UAVs are used for earth science, weather monitoring, and military surveillance. Our success could also increase the effectiveness of a Mars airplane, allowing for observation at levels between the rovers and the orbiters. Since the detection of dust devils on Mars indicates a movement of heat through the atmosphere, a small UAV could ride the same updrafts that form the dust devils and survey the planet from a new altitude.

The fact that hobbyists are able to soar with radio-controlled model gliders in updrafts for long periods of time using only visual cues indicated that our task was possible. The difficulty lay in trying to put soaring skill and good judgment into an algorithm that could run on a miniature autopilot aboard a UAV. The actual vehicle was unimportant. Soaring on updrafts didn't require a different fuselage shape or longer wing span, it required the correct algorithm for detecting thermals in the air and shutting off the engine. We knew immediately that our focus was not going to be on the hardware for our UAV; we had to get the programming right.

A great way to put human language rules into computer code is to use fuzzy logic, which uses a set of nonlinear functions and rules to capture the meaning of less-than-mathematically precise words in algorithms. Our rules for soaring came from the book *Cross-Country Soaring* by Helmut Reichmann, a well-known competition glider pilot. Those rules helped us create the controller that would switch the UAV to soaring mode when a thermal was detected. The new soaring controller worked the first time we tried it in our development simulation, but as soon as we included calculations to smooth the "noise" we expected from the flight sensors, the aircraft wandered along a path that looked like the petals of a daisy. A comparison of the smoothed value used by the controller to the original measured value showed that the problem was caused by a delay the smoothing calculations introduced. Finding better sensors to install on the aircraft would solve the problem, but purchasing and installing the new sensors was beyond the scope of our 1.5 full-time equivalent, one-year project. The solution had to come from a software change instead.

Our team included six engineers and two summer students. None of the engineers worked on the project full time, and the students did not stay for the entire project. This was a side experiment for all of us, and we worked on it because we were interested in the results. We had to be self-motivated for the project to continue and succeed. As a result of our part-time status, only two or three of us could get together at a time, so a rotating cast of people worked on our UAV at any moment. Because of this, software changes had to be easy to make and to test, and they had to be self-explanatory or easily shared through e-mail, since most of our communication was written.

We selected the Cloudcap Piccolo Plus autopilot for flight testing because it used Matlab Simulink software to describe the autopilot guidance and control algorithms. This meant that we could make changes in the software that could be tested using a Matlab-based simulation, or we could test the software using a hardware-in-the-loop (HIL) simulation. The HIL simulation fed aircraft parameters to the Piccolo hardware—the place where the calculations for flight control and soaring were made—and the Piccolo would send back new commands for the rudder, elevator, and aileron, for example, to better maneuver the airplane.

IF OUR EXPERIMENT SUCCEEDED, WE COULD INFLUENCE THE WAY UAVs ARE USED FOR EARTH SCIENCE, WEATHER MONITORING, AND MILITARY SURVEILLANCE.

Once we had the software that would allow us to make quick and easy changes, we had to fix the problem caused by sensor delay. We included an updraft position estimator in the algorithm to give the aircraft a quicker indication of when it was off course. Our subsequent controller worked very well in the Matlab simulation, but it caused erratic behavior in the HIL simulator. We quickly realized we had overtaxed the Piccolo by adding the fuzzy logic controller and updraft estimator in addition to its own autopilot calculations. The fuzzy logic controller was a major cause of the problem; it was taking more time to compute than the updraft estimator or the standard autopilot. Again the problem was quickly fixed in software by re-casting the fuzzy logic controller in a simpler form.

With the software and controller working to our satisfaction, we needed to select a plane. We chose a model glider called the SBXC, made by RnR Products, for flight testing because it had a large fuselage and a wide speed range (18–100 mph). An electric motor with gear reduction was added to allow the airplane to climb to test altitudes and cruise while searching for updrafts. A folding propeller reduced drag when the aircraft shut off its motor during soaring flight. Guidance and control were accomplished using the Piccolo autopilot, which is a self-contained flight computer and sensor package about the size of a brick that weighs only 7.5 oz. It includes GPS, aircraft accelerations and rotational rates, static and total pressure, and a radio modem.

Before flights could begin, we had to solve the problem of flight termination. If the flight had to be terminated, cutting power to the motor would not work because the excellent gliding performance of the motor-glider meant it could still fly long distances. We solved the problem by adding a mechanism to deflect the elevator to 60° and put the aircraft into a "deep stall" if we needed to bring it down. Fortunately, we never had to use this feature during flight tests.

Research flight testing began on August 5, 2005. The first time we turned on the soaring algorithms, the aircraft flew through several updrafts but never switched into soaring mode. One week later, after another software change, we saw our first autonomous detection and engagement of an updraft; the aircraft shut off its motor and climbed 300 feet. The aircraft switched into soaring mode many times during that flight, but it kept finding updrafts that were too weak for soaring. At the end of the flight, we let our pilot, Tony Frackowiak, soar the aircraft remotely so we could gather data to compare with the autonomous system.

We didn't immediately prepare for the next flight. Instead, the ground station operator, Victor Lin, and I did a quick replay of the flight data in Simulink to determine what was happening. We found that the software fix was simple and increased the updraft vertical velocity threshold used in the mode switching logic to determine if an updraft was strong enough to soar in. We demonstrated the flexibility of our process that day by making a flight software change in the field between flights. The total time needed to analyze the problem, fix the software, compile and load the software on the aircraft, and prepare the aircraft for flight was less than two hours. The change proved successful, and the aircraft found five more updrafts and successfully climbed in one of them to gain 1,000 feet of altitude.

I believe that the success of the project was due in a large part to the flexibility allowed by our unique flight-software change process. The development of analytical tools that could be used to look at both simulation results and flight data allowed flight tests to be another step in the development of the algorithms. The tools made possible a "fly-fix-fly" approach that sped the development of the soaring algorithms. Our UAV never did soar as gracefully as a white pelican, but it was amazing to see our design take flight and soar. ●

MICHAEL ALLEN graduated Embry-Riddle Aeronautical University in 1999 and since then has worked at NASA's Dryden Flight Research Center in the stability and controls branch. Michael has worked on the Autonomous Formation Flight Project, the Active Aeroelastic Wing Project, and the Autonomous Soaring Project. Michael is currently working on autonomous refueling and Autonomous Soaring with multiple UAVs.