SAND REPORT

SAND2002-1896
Unlimited Release
Printed June, 2002

Autonomous Dynamic Soaring Platform for Distributed Mobile Sensor Arrays

Mark B.E. Boslough

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2002-1896
Unlimited Release
Printed June 2002

Autonomous Dynamic Soaring Platform for Distributed Mobile Sensor Arrays

Mark B.E. Boslough
Computational Biology and Evolutionary Computing Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0318

ABSTRACT

This project makes use of "biomimetic behavioral engineering" in which adaptive strategies used by animals in the real world are applied to the development of autonomous robots. The key elements of the biomimetic approach are to observe and understand a survival behavior exhibited in nature, to create a mathematical model and simulation capability for that behavior, to modify and optimize the behavior for a desired robotics application, and to implement it. The application described in this report is dynamic soaring, a behavior that certain sea birds use to extract flight energy from laminar wind velocity gradients in the shallow atmospheric boundary layer directly above the ocean surface. Theoretical calculations, computational proof-of-principle demonstrations, and the first instrumented experimental flight test data for dynamic soaring are presented to address the feasibility of developing dynamic soaring flight control algorithms to sustain the flight of unmanned airborne vehicles (UAVs). Both hardware and software were developed for this application. Eight-foot custom foam sailplanes were built and flown in a steep shear gradient. A logging device was designed and constructed with custom software to record flight data during dynamic soaring maneuvers. A computational toolkit was developed to simulate dynamic soaring in special cases and with a full 6-degree of freedom flight dynamics model in a generalized time-dependent wind field. Several 3-dimensional visualization tools were built to replay the flight simulations. A realistic aerodynamics model of an eight-foot sailplane was developed using measured aerodynamic derivatives. Genetic programming methods were developed and linked to the simulations and visualization tools. These tools can now be generalized for other biomimetic behavior applications.

EXECUTIVE SUMMARY

This project makes use of “biomimetic behavioral engineering” in which adaptive strategies used by animals in the real world are applied to the development of autonomous robots. The key elements of the biomimetic approach are to observe and understand a survival behavior exhibited in nature, to create a mathematical model and simulation capability for that behavior, to modify and optimize the behavior for a desired robotics application, and to implement it. The application described in this report is dynamic soaring, a behavior that certain sea birds use to extract flight energy from laminar wind velocity gradients in the shallow atmospheric boundary layer directly above the ocean surface. Theoretical calculations, computational proof-of-principle demonstrations, and the first instrumented experimental flight test data for dynamic soaring are presented to address the feasibility of developing dynamic soaring flight control algorithms to sustain the flight of unmanned airborne vehicles (UAVs). Both hardware and software were developed for this application. Eight-foot custom foam sailplanes were built and flown in a steep shear gradient. A logging device was designed and constructed with custom software to record flight data during dynamic soaring maneuvers. A computational toolkit was developed to simulate dynamic soaring in special cases and with a full 6-degree of freedom flight dynamics model in a generalized time-dependent wind field. Several 3-dimensional visualization tools were built to replay the flight simulations. A realistic aerodynamics model of an eight-foot sailplane was developed using measured aerodynamic derivatives. Genetic programming methods were developed and linked to the simulations and visualization tools. These tools can now be generalized for other biomimetic behavior applications.

.

ACKNOWLEDGMENTS

This project benefited greatly from the contributions of and discussions with George Davidson, Rich Pryor, Pat Crossno, Dan Zimmerer, George Luger, Keith Wiley, Galen Shipman, Stephanie Forrest, Lisa Marron, Rush Robinett, Barry Spletzer, Joe Wurts, Pat Bowman, Dave Reese, Gordon Jennings, and Ferdinand Hendriks. Robert Carlton and Art Hale test flew the model Schweizer sailplane and provided constructive criticisms on its flight behavior. Rebecca Brannon reviewed an early draft of parts of Section II and corrected several errors. Funding was provided under the LDRD program, project 10380, task 01.

CONTENTS **Page**

I. INTRODUCTION

One of the major limitations in developing practical autonomous robots is the power requirement for their mobility. This is particularly true for unmanned airborne vehicles (UAVs) that are large enough to carry sensors to be used for surveillance or other data-gathering applications. UAVs must consume significant energy simply to stay aloft, even when they are not collecting information. This power constraint limits the potential feasibility of these aircraft for most applications unless they can harvest energy from the environment as they fly.

By applying biomimetic principles, one finds that nature has already solved this problem to some extent. A simple scaling argument demonstrates that larger birds require much more energy per unit mass to fly. Cruise velocity is proportional to wing loading (weight per unit wing area). Weight increases as the cube of the characteristic dimension, but wing area only increases as the square. Drag forces increase with the velocity squared, so large birds fly faster but suffer a significant power penalty owing to their size. Likewise, the power required to flap large wings for propulsion is much greater than that needed for flapping small wings. Consequently, large birds have learned to soar for extended periods of time without beating their wings.

For example, two of the world's largest birds are the California condor (*Gymnogyps californianus*) and the wandering albatross (*Diomedea exulans*, Figure 1). Both species have evolved flight behaviors that minimize wing flapping, but use two very different strategies that are adapted to their respective environments. Whereas land birds tend to take advantage of updrafts (static soaring), sea birds primarily make use of velocity gradients (dynamic soaring).

Land birds such as condors and vultures can sustain airspeeds of about 90 km/hour for hours without beating their wings. They soar in updrafts (thermal and orographic uplift) where the

Figure 1. Wandering albatross soars near Sydney, Australia (photo: Tony Palliser).

vertical component of wind velocity offsets the sink rate of their gliding flight. This is *static soaring*, and birds that depend on uplift often become grounded during still periods of temperature inversion and atmospheric stability.

Sea birds, such as albatrosses and petrels, can also fly for long periods while keeping their wings fixed. Weimerskirch and Robertson [1994] used satellite-tracking methods to observe a light-mantled sooty albatross (*Phoebetria palpebrata*) fly a distance of 6463 km in only 10.4 days. A recently publicized study by Wake Forest University used GPS measurements to track a Laysan albatross, which flew nearly 40,000 km in 90 days. Another GPS tracking study of wandering albatrosses revealed that they attained speeds of up to 135 km/hour, and typically cruise at speeds in excess of 85 km/hour [Weimerskirch *et al*. 2002]. Wandering albatrosses have been observed to follow ships for days at a stretch, with almost no need to flap their wings. These large sea birds make use of *dynamic soaring* in the near-surface atmospheric boundary layer by flying in a circuitous pattern of climbing upwind, turning, and diving downwind, then skimming the sea surface while turning back upwind. This allows vast areas of the sea to be searched and foraged with little expenditure of energy

The most dependable, sustained surface winds are over the open ocean in the wind belts known as the low latitude easterly trade winds, the mid latitude westerlies (known in the southern hemisphere as the "roaring forties"), and the high latitude polar easterlies. In addition, there are regional and seasonal monsoons, most notably the Asian monsoon--which is accompanied by sustained surface winds greater than 50 km/hour--in the strategically important vicinity of the northern Indian Ocean, Persian Gulf, and Arabian Sea. The development of an autonomous sailplane that is highly mobile and can remain aloft without power would provide a valuable platform for distributed sensor arrays in all of these parts of the world. Such UAVs could also be deployed over land under appropriate conditions. The purpose of this report is to outline the theory of dynamic soaring, present new computational and experimental proof-of-principle flight tests, and document the development of evolutionary biomimetic behavior algorithms.

II. DYNAMIC SOARING

A. Descriptive analysis

In a motionless atmosphere without wind velocity gradients, sustained gliding flight would be impossible. However, the real atmosphere is in near-constant motion, and many land-based soaring birds and experienced pilots of unpowered fixed-wing aircraft (sailplanes) have learned to remain aloft for many hours by seeking updrafts and avoiding downdrafts. When the upward velocity of an air mass exceeds the still-air descent rate of an aircraft, there is a net altitude gain without any local expenditure of energy. Updrafts can be caused by buoyant instabilities ("thermals"), orographic uplift ("slope lift"), and by atmospheric waves (Figure 2). The general term for this type of flight is "static soaring".

A second method of gliding flight is dynamic soaring, which is the focus of this project. This strategy was discovered and analyzed by Lord Rayleigh [1883], who developed a physical model to show how large sea birds can convert wind energy at a horizontal shear boundary into energy of flight without any physical work output. He presented a simple two-layer horizontal velocity model in which the bird soars in an inclined circle, crossing upward through the shear boundary when heading upwind, and crossing downward through it when heading downwind. His model

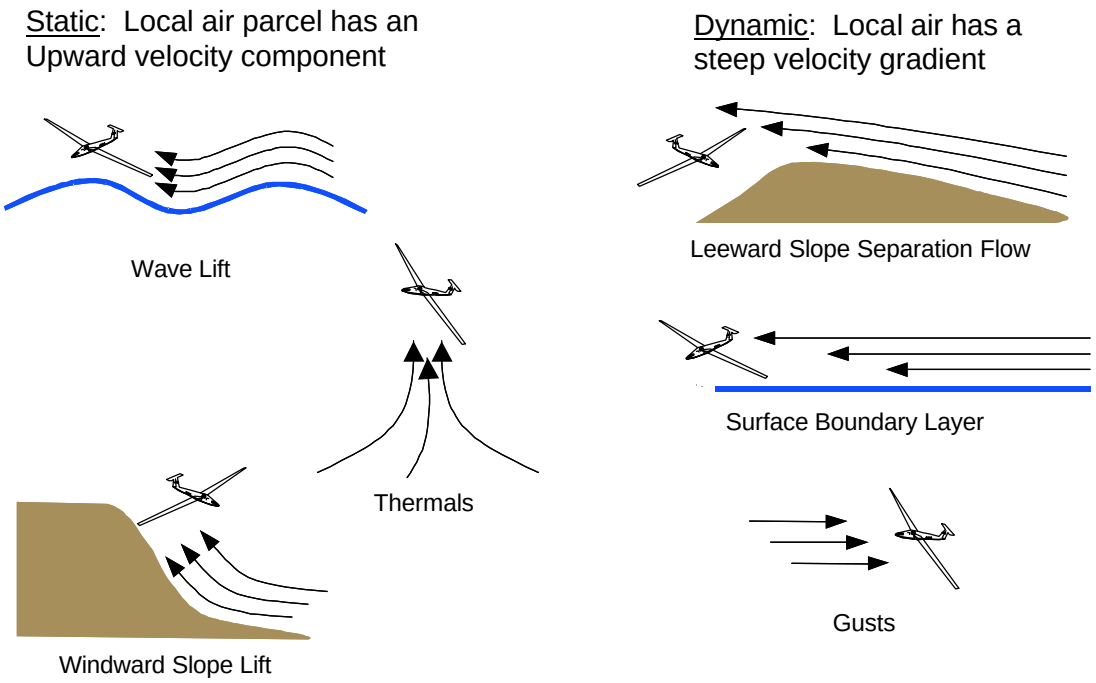

Figure 2. Various means of soaring.

explicitly assumed that the air parcels had no vertical velocity components, and therefore provided the first description of dynamic soaring, which he explained as follows:

> *…a bird without working his wings cannot, either in still air or in a uniform horizontal wind, maintain his level indefinitely. For a short time such maintenance is possible at the expense of an initial relative velocity, but this must soon be exhausted. Whenever therefore a bird pursues his course for some time without working his wings, we must conclude either*
>
> *(1) that the course is not horizontal,*
>
> *(2) that the wind is not horizontal, or*
>
> *(3) that the wind is not uniform.*
>
> *It is probable that the truth is usually represented by (1) or (2); but the question I wish to raise is whether the cause suggested by (3) may not sometimes come into operation.*

Indeed (1) is simple gliding flight, (2) is static soaring, and (3) is dynamic soaring. Subsequent work has demonstrated unambiguously that Rayleigh correctly described the flight of large sea birds, which fly in the relatively thin atmospheric boundary layer directly above the sea surface in places where the wind is strong and steady. Rayleigh concluded his paper with a paragraph dedicated to a potential application: human flight. Since he had shown that flight is possible by extracting energy directly from the wind, he made the following recommendation:

> *When a man can launch himself from an elevation and glide long distances before reaching the ground, an important step will have been gained, and until this is done, it is very improbable that any attempt to maintain the level by expenditure of work can be successful.*

This is precisely the approach taken by the Wright brothers two decades later at Kitty Hawk. While dynamic soaring for human flight was seen as an important possibility by many early aviation pioneers (*e.g.* Langley, 1893), interest quickly faded after powered flight became a reality. Moreover, thermal soaring was eventually discovered to be practical, and it became recognized that the marine shear boundary was too thin in comparison to the dimensions of a human-piloted glider. Dynamic soaring was all but forgotten by aviators.

On the other hand, workers in the relatively isolated field of "aeroecology" (the application of aerodynamic principles to the study of avian ecology) maintained their interest in dynamic soaring as a means by which sea birds extract flight energy from the wind. This research was reviewed and greatly extended by Cone (1964) who focused on flight in the marine shear boundary layer. According to Cone,

> *… the albatross has found a remarkable way to exploit the energy of horizontally-moving air flows. Unlike the steady and almost automatic soaring of the vulture, that of the albatross involves continuous voluntary maneuvering and control regulation. Yet, so perfectly adapted in structure and instinct is the albatross to its particular mode of flight that it performs its cycle with a geometrical precision of amazing exactness. … the albatross is able to accomplish such flight and to remain at sea for years at a time, covering untold thousands of miles, all without any significant expenditure of its own muscular energy for propulsion.*

While dynamic soaring has long been recognized as being an important component of the way that albatrosses transport themselves, it is not the only mechanism. Hargrave [1899] published his observations of sea bird flight and concluded that the animals obtain a significant fraction of their flight energy from static lift from flying on the windward side of ocean waves. He remarked that all the complications associated with real flight "…have been considered in the evolution of a sailing bird and must be reckoned with by the designer of a wave driven flying machine." Wilson [1975], developed a model for this type of soaring which he called "sweeping flight". By staying in the wave-generated updraft, the bird can increase its speed up to its polar maximum speed (V_2) for a given upcurrent, at which point its still-air sink rate exactly balances the upward velocity component of the wind and it ceases to accelerate. If the bird's mass is m and stall speed is V_1, then its excess kinetic energy is given by $E = m(V_2^2 - V_1^2)/2$, and it can convert this to potential energy by rapidly rising out of the wave to a maximum height of $h = (V_2^2 - V_1^2)/2g$. Once the bird has attained this height it can cruise back downward to the next wave in whatever direction it wishes to go. Wilson estimated that albatrosses should be able to reach an altitude of 20 m by this method, and suggested that by some complicated pattern of flight the birds could combine sweeping flight with dynamic soaring to maximize the conversion of wind energy to kinetic energy of flight. Wilson also felt obliged, like the others, to suggest a practical application and modeled the flight of a glider by this means. He also rejected this as unrealistic due to the large size of a human-piloted glider relative to sea waves.

There is still no consensus on the relative importance of dynamic soaring versus sweeping flight. In an authoritative review, Pennycuick [1982] suggests that albatrosses extract energy from the wind by both means, but that sweeping flight is relatively more important for upwind flight, while dynamic soaring is dominates for downwind flight. This conclusion is based on a review of the literature, combined with direct observations, and is presented as a tentative finding in need of more analysis and data. More recent observational research on sea bird flight [Spear

& Ainley, 1997] led to the conclusion that crosswind and tailwind flight of certain large petrels is different and more efficient than that of albatrosses. The authors suggest that this better- than-expected performance may have resulted from a different type of soaring, not previously described in the literature on sea birds. Clearly, there are still significant contributions to be made in the modeling of sea bird flight, especially in the identification of optimal flight strategies.

Nevertheless, the albatross is remarkable in its ability to extract energy from the wind for its own purposes. In the words of Cone,

> *Using the practically boundless energy of the sea shear layers, the albatross, on its efficient wings, traverses vast expanses of sea from dawn to dusk in endless search of the squid and shrimp which form its diet. With the exception of the breeding season, when it returns to the small isolated islands where it nests, the albatross is truly pelagic, remaining far at sea. There it ceaselessly orbits within the narrow wind shear layer above the water, alighting only briefly to claim its food from the sea.*

However, Cone was dubious that "any generally useful degree of dynamic soaring by man will be developed, at least in the foreseeable future, despite a number of theoretical possibilities". But the technical advances that Cone did not forsee--such as miniature computers and sensors--have now become a reality, and it is likely that Cone did not expect "dynamic soaring by man" to include "dynamic soaring by small robots".

B. Piecewise two-dimensional numerical analysis

As a first step towards applying evolutionary computing methods to the optimization of dynamically soaring "perpetual flight machines," sea bird flight maneuvers are analyzed in two- and three-dimensional space. A genetic algorithm is then applied to determine behavioral parameters that maximize upwind progress under a zero flight energy expenditure condition for idealized albatrosses. This analysis replicates previously published computer simulations of albatross flight [Wood, 1973], but reveals a mistake in the earlier analysis that invalidates one of its key results. Wood [1973] had incorrectly concluded that albatrosses in his model (using maneuvers similar to the type they have been observed to fly, but constrained to piecewise two-dimensional motion) can make direct upwind progress without power in a purely horizontal wind

Three flight cycles are considered, all involving motion in a laminar boundary layer in which wind speed depends only on the logarithm of the altitude. The equation of motion includes lift-dependent drag forces, and uses published values of drag coefficients for albatrosses, with a maximum wing load of three times the bird's weight. The first flight cycle is pure upwind flight in two dimensions. The second is three dimensional upwind fight with horizontal turns and a downwind dive. The third includes a generalized aerobatic figure--the Reverse Half Cuban Eight--by which the bird turns downwind using an inverted high-acceleration maneuver. The first two cycles are inadequate for upwind progress, but the third can be used to fly upwind at no energy cost, with parameters that can be optimized by genetic algorithm. Real albatrosses have not been observed to fly this way, and upwind progress in an idealized atmosphere is obviously not the only measure of success for analyzing the natural optimization of sea bird flight. Because survival and reproduction are the most important elements of Darwinian evolution of animal behavior, and low-altitude inverted aerobatics are extremely risky, the optimal flight

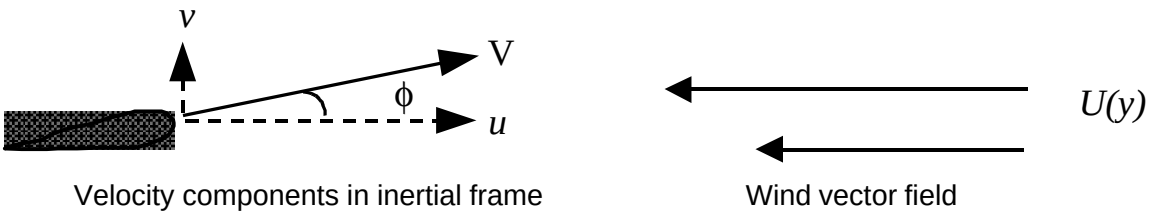

Figure 3. Bird and wind velocities in inertial reference frame.

maneuvers in nature are not necessarily the ones that maximize upwind progress. However, this measure of success can be used as a quatitative "fitness functon" for optimization of soaring robots, because the learning phase in which the unsuccessful individuals are killed computationally can be performed off-line (that is, as a simulation), with little relative cost.

This work follows Wood's [1973] analysis of the kinematics of flight, using precisely the same parameters for the dynamic properties of flying albatrosses, and making the same assumptions about the gradient of wind shear near the ocean surface. This provides a conveniant means to validate the present simulations with his, and gave a departure point for new simulations. For two-dimensional flight constrained to a vertical plane parallel to the wind direction (Figure 3), the local state (with respect to the air) of the bird depends only on the climb angle (ϕ), and on the airspeed (V). If the wind speed depends on altitude (y), then its global state (with respect to the world) also depends on the altitude. The global state refers to the condition of the bird in an inertial reference frame, but it is the bird's kinetic and potential energy with respect to its immediate surroundings (the "wind-fixed" reference frame) that is useful for flight. For the present simulations, consider ϕ, V, and y to be the complete set of state variables. For a bird moving with an inertial horizontal velocity component u opposite to the altitude-dependent horizontal wind speed $U(y)$, and a vertical velocity component v, the kinematic equations are:

$$u = V\cos\phi - U(y) \qquad\qquad v = V\sin\phi \tag{1}$$

Dimensionless lift (l) is defined as an upward force per unit weight of the bird and normal to its direction of flight, drag (d) is a similarly normalized backward force, and parallel to the flight direction, and gravity (g) acts directly downward. The net force on the bird is the vector sum, and the two components are:

$$l\sin\phi + d\cos\phi = -\frac{1}{g}\left(\frac{\mathrm{d}u}{\mathrm{d}t}\right) \qquad\qquad l\cos\phi - d\sin\phi = l + \frac{1}{g}\left(\frac{\mathrm{d}v}{\mathrm{d}t}\right) \tag{2}$$

Differentiating equations (1) and substituting the acceleration terms into (2) yields equations for dimensionless lift and drag:

$$l = \frac{V}{g}\left(\frac{\mathrm{d}U}{\mathrm{d}y}\right)\sin^2\phi + \cos\phi + \frac{V}{g}\left(\frac{\mathrm{d}\phi}{\mathrm{d}t}\right) \tag{3}$$

$$d = \frac{V}{g}\left(\frac{\mathrm{d}U}{\mathrm{d}y}\right)\sin\phi\cos\phi - \sin\phi - \frac{1}{g}\left(\frac{\mathrm{d}V}{\mathrm{d}t}\right) \tag{4}$$

Wood [1973] used a standard aerodynamic drag equation which can be written in the form:

$$d = aV^2 + bl^2/V^2 \tag{5}$$

where the first term is due to skin resistance and the second is due to wake vortices which are strong functions of lift. The present calculations use the same values as Wood [1973]: $a = 0.96\text{x}10^{-4}$, and $b = 4.25$ (in m.k.s. units). These parameters were determined from measured values of albatross weight, wing area, and aspect ratio, and an estimated lift coefficient. Equations (3)-(5) can be combined into a nonlinear differential equation of motion, which can be solved numerically:

$$\frac{dV}{dt} = V\frac{dU}{dy}\sin\phi\cos\phi - g\sin\phi - gaV^2 - \frac{gb}{V^2}\left(\frac{V}{g}\frac{dU}{dy}\sin^2\phi + \cos\phi + \frac{V}{g}\frac{d\phi}{dt}\right)^2 \tag{6}$$

For non-horizontal flight, equation (6) requires an explicit numerical time integration to determine the trajectory of the bird, and the flight path can be defined by controlling the climb (or descent) angle ϕ within constraints imposed by the strength of the bird (assumed here to be three times its weight) and the stall speed below which the bird loses lift. For horizontal flight, ($\phi = 0$), equation (6) is easily seen to become linear and it can be integrated analytically. Some of the special cases require a preliminary generalization to three-dimensional flight, which needs

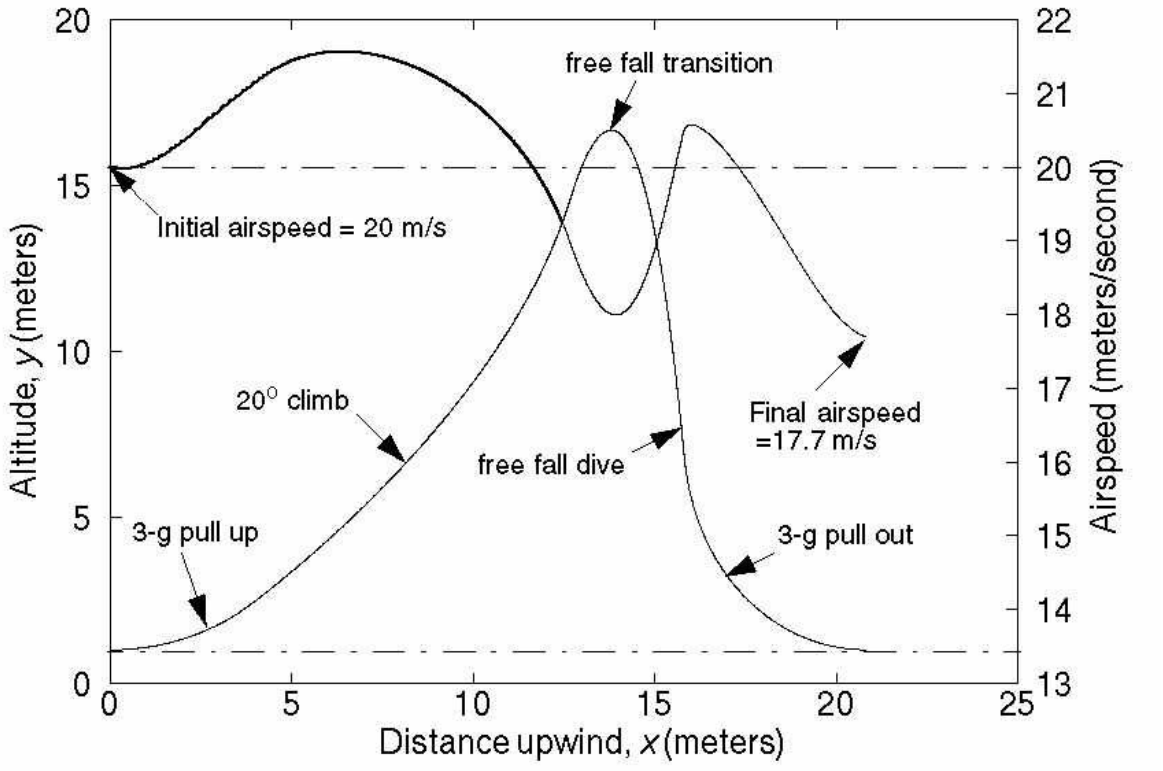

Figure 4. Replication of 2D upwind flight trajectory of Wood [1973] with identical results.

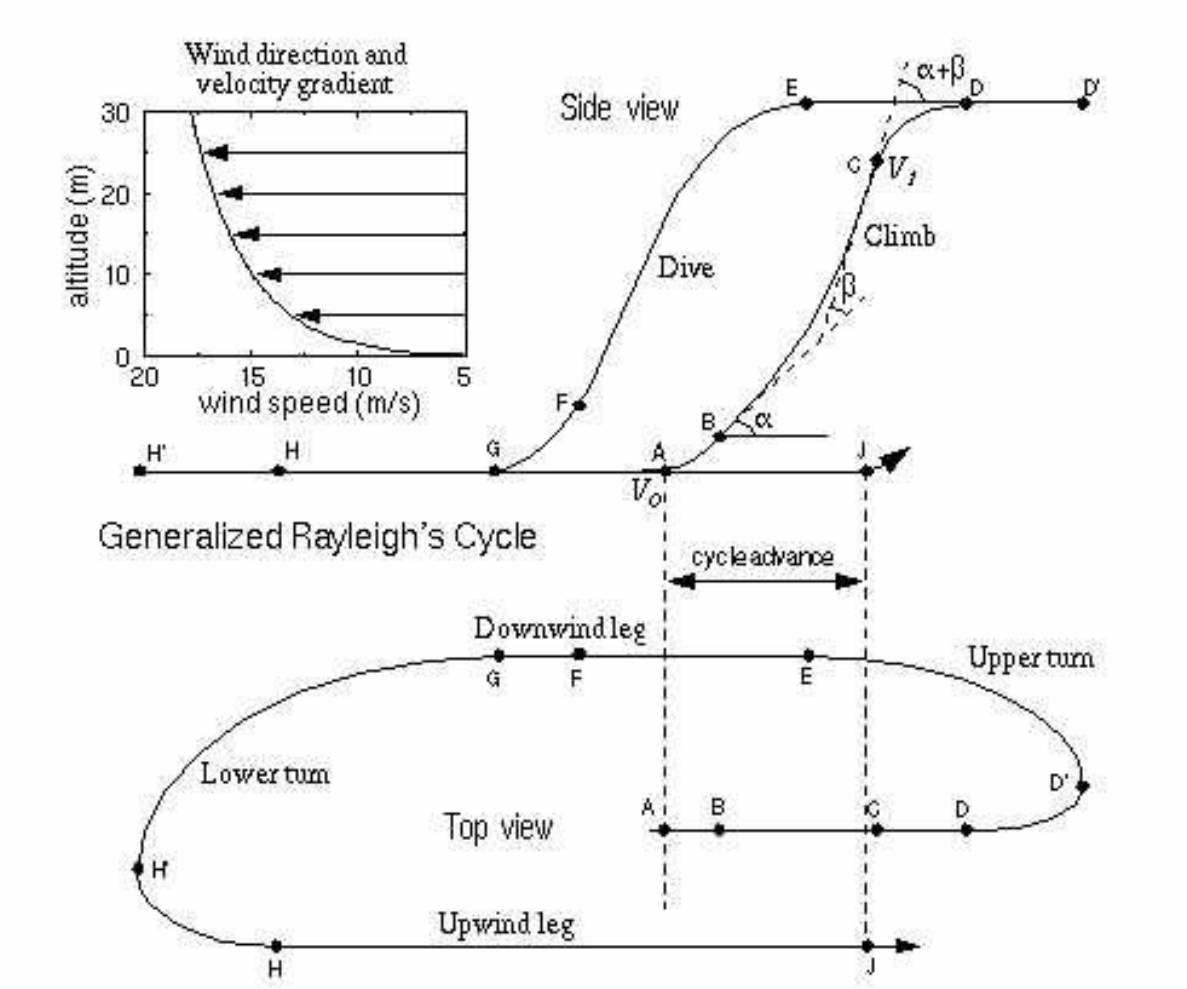

Figure 5. Generalized flight cycle originally suggested by Rayleigh [1883].

another state variable θ, the angle about a vertical axis with respect to the wind direction. All cases use the logarithmic wind speed of Wood [1973] and others:

$$\frac{U}{U_{10}} = 0.1769 \log(y / 0.03485) \qquad \frac{dU}{dy} = 0.1769 \frac{U_{10}}{y} \qquad (7)$$

where m.k.s. units are used, and where U_{10} is the reference velocity at 10 meters above the surface, which in all three simulations were set to 15 m/s (this is plotted in the inset in Figure 5).

Of the three different types of flight cycles calculated here, two can be represented as closed figures in a phase space (where one or more of the bird's state variables changes smoothly from an initial state, and then returns to its initial state). In general the bird's coordinates will have changed, and it is this change in coordinates that provides a fitness function to optimize the parameters that define the cycle. In the simple cases simulated here, only the *x* coordinate in the upwind direction determines the fitness, but in general the crosswind component could be combined to optimize progress in other directions. Obviously the change of coordinates must be in the desired direction for the bird to make progress.

The first cycle is purely two-dimensional flight with no turns, and completely in the upwind direction, as proposed by Walkden [1925]. This cycle was modeled by Wood [1973], who started the bird in the upwind direction, 1 meter from the surface of the water, with an inertial velocity of 20 m/s. In his simulation, the climb angle is increased at a rate which maximizes the allowable wing loading on the bird, until an angle of 20° is achieved. This climb angle is held constant until the bird's airspeed drops to 19 m/s, at which time the bird goes through a zero-lift transition (free fall) to a dive back toward horizontal flight at 1 meter from the surface, completing the cycle. This cycle is shown in Figure 4 by plotting altitude and airspeed as a function of distance. The present calculation yields results identical to those of Wood, which supports his conclusion that this cycle cannot be completed because the bird does not return to its original velocty. Figure 4 is in complete agreement with Wood's Figure 8 providing an independent confirmation for the present calculations. Note that the airspeed actually increases during the climb, which is the counter-intuitive basis for dynamic soaring and the mechanism by which energy is taken from the wind gradient. The purely upwind flight of Walkden's cycle results in a rapid loss of airspeed and energy during the dive phase, however, and this is why it does not work.

Table 1: Comparison to previous results

Point	Time t (s) [Wood, 1973]	Airspeed V (m/s) [Wood, 1973]	Time t (s) [present work]	Airspeed V (m/s) [present work]
A	0.00	20.00	0.00	20.00
B	0.36	20.13 (21.13?)	0.38	21.11
C	2.08	18.96	2.09	18.99
D	2.74	17.97	2.75	18.01
E	4.65	15.43	4.62	14.93
F	6.08	22.32	5.88	20.66
G	6.86	28.44	6.85	27.85
H	10.00	26.91	9.80	24.30
J	21.55	19.99	17.73	19.99

The second cycle is a generalized one that was also simulated by Wood, and has two turns and a downwind dive, as originally suggested by Rayleigh [1883]. In this cycle, Wood modeled the turns as being purely horizontal maneuvers. This allowed him to determine exactly the position, airspeed and time interval using an analytic solution of equation (6), which is linear for constant y. A generalized version of Rayleigh's cycle is shown in profile and plan view in Figure 5. Wood's version of Rayleigh's cycle, begins exactly like the purely upwind cycle. The bird initiates a maximum-load (3-g) pull up at 20 m/s at A, climbs at a constant angle of 20° between A and B, and transitions at zero load (free fall) to horizontal flight when the velocity drops to 19 m/s at D. The bird then makes a horizontal, maximum-load turn downwind, and initiates a zero-load dive upon completion of the turn at E, pulling out at maximum load at F to level out at the original altitude of 1 m, going downwind and initiating a 180° turn at G, which is complete at H. The bird then flies straight until its velocity drops back to the original 20 m/s, and the increase in upwind position is the "cycle advance" which is the measure of progress. The cycle can be generalized so that there are four adjustable parameters: V_0 is the initiation velocity that starts the cycle, α is the initial climb angle, β is the increase or decrease in climb angle, and V_1 is the initiation velocity for transition to level flight. The rest of the cycle is

completely constrained if these parameters are fixed (for a given wind profile and aerodynamic properties of the bird), and they can be optimized using a genetic algorithm to find the combination that maximizes the net cycle advance.

In this case, Wood's results were not reproducible. Table 1 compares the present output to Wood's and shows inexplicable discrepancies only for the turns, which Wood solved analytically (the airspeed at point B is probably a typographical error in Wood's paper because it is inconsistent with his Figure 8, and its difference from the present value is close to unity). This discrepancy was double-checked by solving the horizontal turn segments analytically, which reproduced the present numerical results to a high degree of accuracy (see Appendix A for this analytical solution).

Table 2: Comparison of analytical to numerical solutions for horizontal turns

Quantity	Numerical solution	Analytical solution
V_D, (velocity at point D, m/s)	18.01	18.01
V_E, (velocity at point E, m/s)	14.93	14.96
Δt_{DE}, (time to complete upper turn, s)	1.87	1.87
V_G, (velocity at point G, m/s)	27.85	27.85
V_H, (velocity at point H, m/s)	24.30	24.32
Δt_{GH}, (time to complete lower turn, s)	2.95	2.95

Table 2 presents a comparison between the present numerical solution and the analytical solution for the time interval and the initial and final velocities for both turns. It is likely that Wood used an incorrect integral formula for his analytical solution. It is reasonably straightforward to solve for the time and velocity loss, but the analytical solution for distance involves transcendental functions within functions and does not appear to be tractable. Whereas Wood calculated a net gain of 48.1 meters per cycle, there is actually a net loss of 13.3 meters. An albatross flying this way (with the parameters assumed by Wood) does not make progress directly against the wind, and when the generalized Rayleigh's loop is optimized, the net gain is almost negligible. Unfortunately, this mistake renders Wood's primary result incorrect, which was the final statement in his summary:

> *...the more usual flight cycle which involves a dive in the downwind direction is completed very easily without loss of flying speed. In fact, by diving steeply, sufficient excess airspeed can be gained to permit a very long final upwind glide at low level. This repasses the starting position and thus gives a slow but definite nett progress against the wind.*

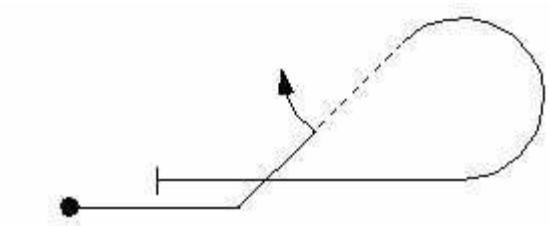

Figure 6. Schematic diagram of *reverse half Cuban eight*.

This conclusion has been used by Pennycuick [1982] and others as supporting evidence for the relative importance of dynamic soaring in sea birds. Even though the conlusion was due to a mistake by Woods, its still stands when Woods' contraints are relaxed.

A third cycle can be considered in an effort to capture the gain of the downwind dive while avoiding the huge loss associated with the downwind turn at high altitude where the wind speed is highest. The key element of this cycle is an aerobatic figure called a "reverse half Cuban eight," similar to the more familiar split-S, which is a reverse Immelman turn devised by pilots in World War I as a means of quickly reversing direction. Figure 6 depicts a reverse half Cuban eight using standard aerobatic notation, in which dot is the start, the arrow indicates a half roll, the dashed line refers to inverted flight, and the vertical bar is the end of the figure.

As with Rayleigh's cycle, this maneuver can be genralized in terms of the same parameters for optimization purposes. In the standard reverse half Cuban eight, the climb is executed at a 45^o angle, whereas the angle can vary in the generalized version, with the entry and exit angles being adjustable parameters.

C. Genetic algorithm optimization of simple cycle

The cycle parameters V_0 (initiation velocity for climb), V_1 (initiation velocity for end of climb), α (initial climb angle), and β (change in climb angle) can be varied to determine the combination that maximizes the net cycle advance. There are many algorithms that could be used for this optimization procedure, including random searches, hill climbers, and genetic algorithms. No optimization analysis was done by Wood [1973], who described his simulations as a few random cases and closed by saying that, "Lack of time has prevented any serious attempt to optimise the various cycle parameters..." As it turned out, he chose parameters that were close to the optimal case, even though he integrated incorrectly.

Genetic algorithms [*e.g.* Forrest, 1993] are well suited for optimization of parameters when there is no way to numerically invert a problem, and the forward problem must be simulated repeatedly. In the present case, a given set of parameters uniquely defines a cycle, and by running the simulation the net cycle advance can be determined and used to compare the fitness of different sets of parameters. The above-listed parameters are not the only possible set, but provide a starting point from which this work can be easily extended. Other possibilities are suggested below. Details specific to the genetic algorithm used here are given in Appendix B.

The same set of parameters was used to optimize both the generalized Rayleigh and Cuban eight cycles. Additional parameters could be used to characterize both flight maneuvers, perhaps leading to significant improvements in glide distance per cycle, but for this preliminary analysis it makes sense to chose the smallest number that would allow significant variation. In some cases, a bird would enter a turn at such a low velocity that its speed would drop to zero, and in other cases, the bird's altitude would be too low to recover from the Cuban eight maneuver.

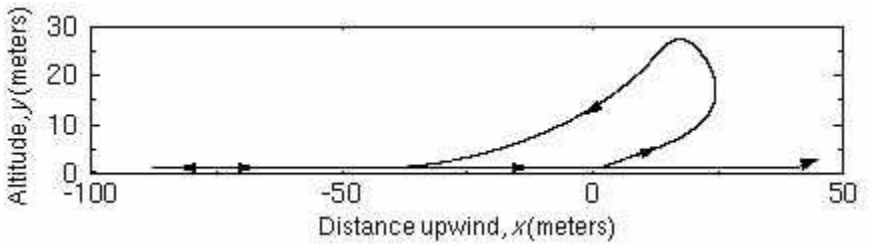

Figure 7. Actual trajectory of the near-optimum Cuban eight cycle in inertial frame.

When this happened, the function would set the individuals fitness to -100 for the Rayleigh, and to zero for the Cuban eight. These values were lower than any fitness achieved by a simulated albatross that actually made it around the cycle, so these individuals were strongly selected against. Population size was varied, with the largest run being 101 individuals for 450 generations. In general, 51 generations for about 50 cycles converged to within tens of centimeters of the best of much larger runs.

The best individual for the generalized Rayleigh's cycle began its climb at an airspeed of 19.45 m/s, with an initial angle of only 6.6°. The bird steepened its ascent into the wind up to 57.8°, and kept climbing until its airspeed dropped to 12.0 m/s. The rest of the cycle is prescribed: free fall to level flight, maximum acceleration horizontal turn downwind, free fall dive until maximum acceleration pullout is required, then maximum acceleration turn upwind and glide until airspeed is back to 19.45 m/s. At this point, the bird is only 2.8 meters beyond its starting point (compared to a loss of 13.3 meters using Wood's parameters). Wood's assumed starting speed of 20 m/s is remarkably close to the optimum, but there are two major differences between his climb parameters and the ones required to yield a net gain. The optimal albatross starts its climb much shallower, presumably to get the most gain out of the wind gradient while spending a larger fraction of its time flying against the lower-velocity headwind at low altitude. Once it is through the steep part of the shear boundary layer, it wants to have a steeper ascent to get to high altitude and dump its airspeed to as low a value as possible so it can turn around and dive before it gets blown too far backward by the higher winds at high altitude.

Birds that flew the Cuban eight cycle did significantly better, with the best individual starting its ascent with an airspeed of 20.38 m/s, climbing at an initial angle of 12.6°, and steepening to 37.7° until its speed dropped to 15.0 m/s. In this case, the strategy is similar, but with one difference: the albatross only needs to gain enough altitude to safely do a reverse half Cuban eight without hitting the water. The optimum Cuban eight cycle is displayed in Figure 7 in the

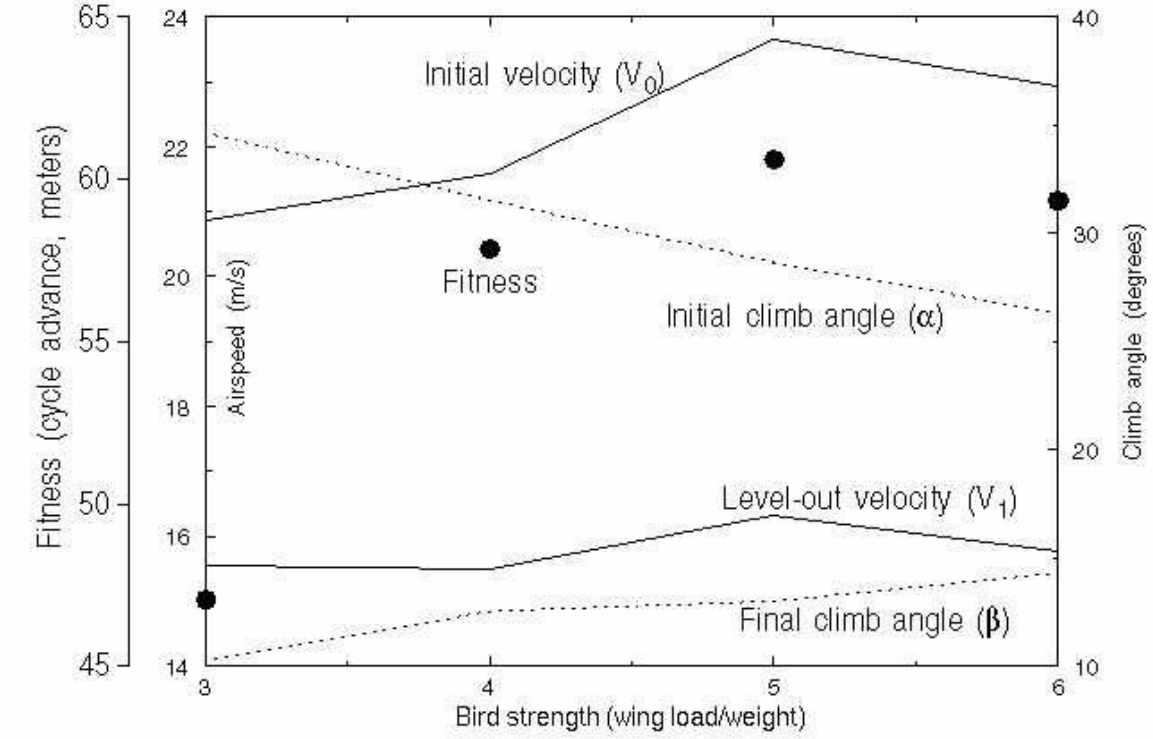

Figure 8. Determination of how fitness and optimization parameters vary with bird strength.

inertial reference frame (with respect to the ground). The "loop" is with respect to the air, and does not appear as a loop in interial coordinates because the bird is being blown downwind.

To explore the effect of wing strength on the best performance, the generalized Cuban eight cycle was optimized for birds that could survive wing loadings of 3, 4, 5, and 6 times the bird's weight. In all cases, the bird's acceleration was limited to 3 times gravity during the loop, but was allowed to increase to the maximum allowed during turns and climbs . The stronger birds do better (see Figure 8), primarily because that can turn upwind more quickly after their downwind dive. For this tightly-defined idealized problem, the results of the various flight strategies are compared graphically in Figure 9.

The calculations of this section were intended to provide insight, and there is little doubt that better cycles can be found with more generalization of the Cuban eight cycle, especially by allowing the turns to initiate while the bird is diving or climbing, and by letting the acceleration vary during the half loop. It makes much more sense to allow the genetic algorithm to link other generalized aerobatic maneuvers together to form cycles, rather than constraining the flight to a specific, predetermined maneuver and optimizing only on its parameters. This strategy is more consistent with the intent of genetic algorithms, and takes advantage of the building blocks concept of Holland [1975]. This might lead to the discovery of more complicated (but more efficient) flight patterns that would be required for crosswind flight, or for sweeping flight [Wilson 1975] under conditions that allow it, such as when the wind field includes contributions from wave upcurrents. This requires the use of a full, 6 degree-of-freedom flight dynamics model, which is described in Section III.

One obvious question deserves to be addressed: why don't albatrosses fly upside-down in the real world? This presumes that the fitness function chosen for the present optimization is the

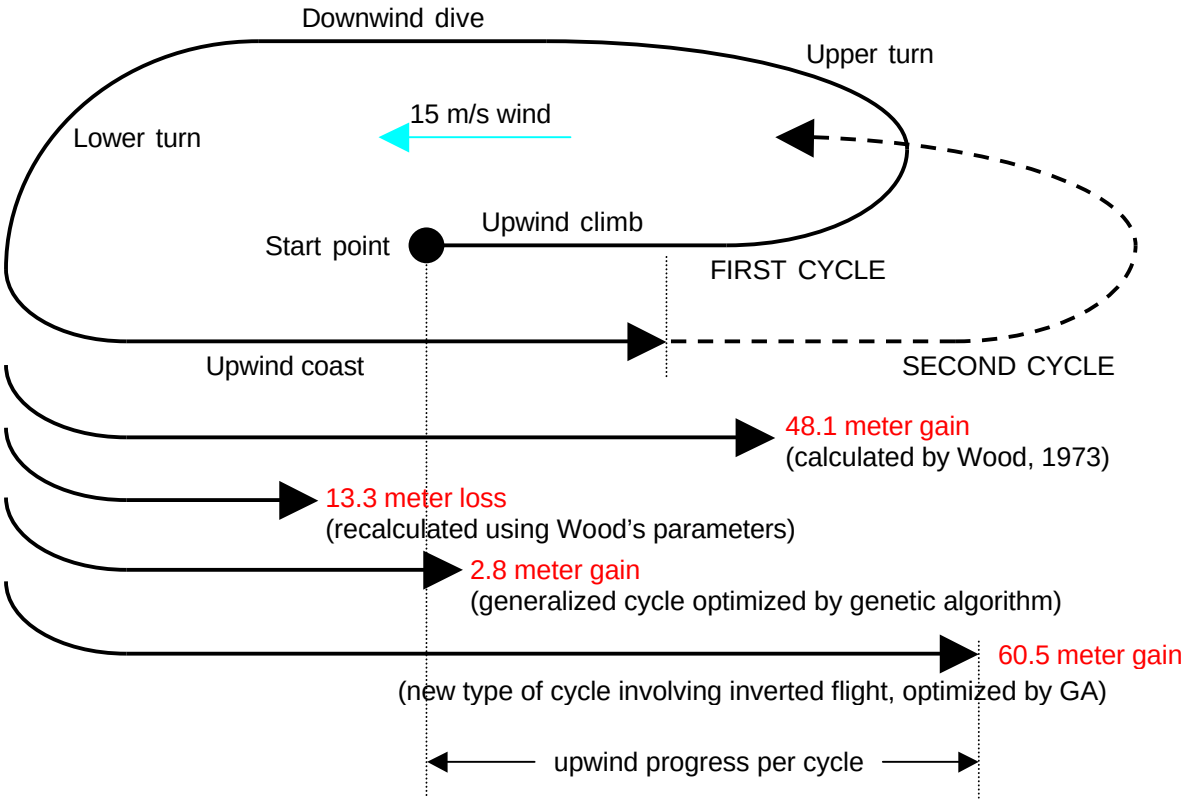

Figure 9. Comparison of fitness value for various flight cycles.

same as the one chosen by nature. Clearly there is more to the fitness of a real albatross than how fast it can fly upwind without flapping its wings. For an animal that is subject to the Darwinian evolution of the real world, the most important component of its fitness is the likelihood that it will survive long enough to reproduce (with viable offspring) and so unsafe behaviors are highly selected against. The fitness landscape of the Cuban eight cycle is shaped somewhat like a lookout point; it gradually rises as the parameters are optimized, but drops off precipitously on several sides. If the parameters are pushed too far (*e.g.* the bird starts its dive too low) it cannot pull out of its dive and collides with the water at high speed. In real biological systems, a bird will fly in a way that is unlikely to kill it, but new flight maneuvers must be developed "on-line" (*i.e.* in the real world). The Rayleigh cycle merely consists of climbs, horizontal turns, and dives, and can be modified without any real danger. On the other hand, there is no pattern that can be smoothly and safely evolved into a Cuban eight cycle, because entry into that maneuver requires an inverted dive. By contrast, a robotic albatross could be trained to fly "off-line" (within a simulation rather than in the real world) using genetic algorithms to optimize simulated flight maneuvers, and would not have to pay the ultimate price for pushing the envelope too far.

D. Three-dimensional analysis

The previous numerical analysis makes use a number of simplifying assumptions. The drag equation assumes that the aerodynamic shape of the bird remains fixed, and neglects other contributions to lift and drag for real birds and aircraft. All the aerodynamic characteristics of the bird are distilled into only five parameters. Moreover, to make the equations of motion easy to solve, the bird was constrained to move only in a horizontal or vertical plane.

Hendricks [1972] presents a more sophisticated analysis in considering the flight of a glider in a non-uniform wind field. He models the flight of a point-mass aircraft with realistic aerodynamic properties in two different kinds of idealized wind fields: 1) a uniform vertical wind shear gradient, and 2) isotropic turbulence. In addition to piecewise two-dimensional gliding in wind shear, three-dimensional motion is analyzed with perturbation methods. Hendricks shows that flight energy gain increases with wing loading, and that the optimum turning cycle frequency is identical to the phugoid oscillation frequency. He also addresses the possibility of "gust soaring" in which flight energy can be extracted from turbulence.

Hendricks describes soaring from the perspective of an observer riding in the plane, who experiences forces in addition to gravity. Drag is usually considered to be an instantaneous, time-dependent force. By describing it as an average force over a time scale that is long compared to the characteristic time of the aircraft motion, it can be balanced by a fictitious mass force due to the effect of a non-uniform atmosphere. For uncontrolled flight, the random effects of atmospheric turbulence have the opposite effect, adding to the effective time-averaged drag (the sink rate of a passive airplane is greater in turbulence than in smooth air). However, it has been argued that by applying appropriate control actions, flight energy can be extracted from eddies and gusts associated with turbulent flow. According to Hendricks, the attempts of pilots to soar using gusts have failed due to the sluggishness and inconsistency of human reactions. By contrast, birds may have developed the ability to properly react to gusts by drawing on an instinctive knowledge of turbulent velocity correlations acquired over millions of years of evolution.

For ease of mathematical analysis, Hendricks introduces the simplifying assumption that airflow around the glider is quasi steady, that the glider does not affect the flow field, and that

there are no torques acting on the glider (*i.e*. it is modeled as a point mass). By considering the case where the characteristic length scale of the wind gradients are much greater than those of the glider, the velocity of the glider relative to the local air mass can be defined. This relative wind velocity is simply the difference between the inertial wind velocity and the inertial glider velocity. It is this relative velocity on which the aerodynamic forces depend. Since flight in turbulence is considered, it seems questionable to assume quasi-steady flow over the aircraft. This simplification is justified by suggesting that flow fluctuations with length scales less than those of the glider are simply parameterized as the effective aerodynamic lift and drag coefficients C_L and C_D, respectively. The dimensionless lift and drag forces are

$$l = \frac{1}{2mg} C_L \rho V^2 S \qquad\qquad d = \frac{1}{2mg} C_D \rho V^2 S \tag{8}$$

where ρ is the air density and S is the wing area. Hendricks very reasonably treats turbulence as being homogeneous and isotropic, with length scales and atmospheric properties for which the 5/3 law of Kolmogorov [1941] applies to the energy spectrum. In addition, Hendricks' analysis depends on the assumption that the glider airspeed is sufficiently fast that the wind variation due to the turbulence is fixed in space.

For this three dimensional analysis, the inertial reference frame neglects earth rotation and other celestial motion, and ignores the curvature of the Earth by using a Cartesian coordinate

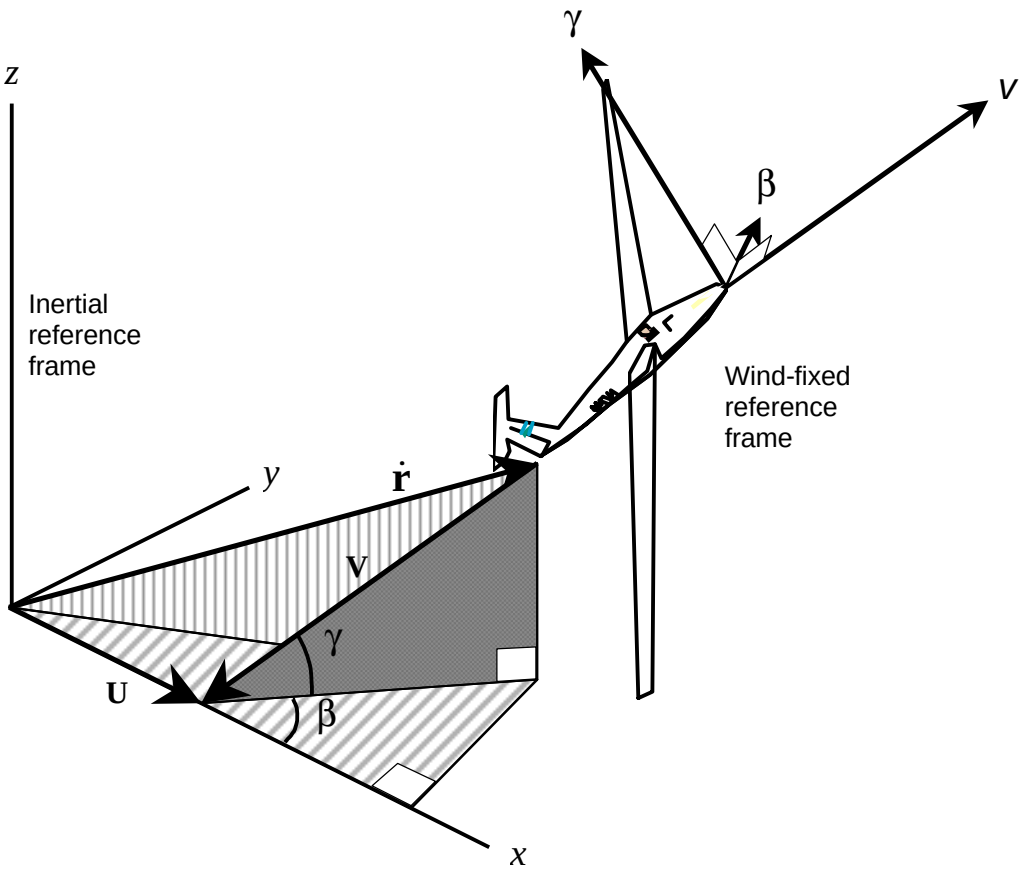

Figure 10. Coordinate systems used for three-dimensional analysis.

system with axes *x*, *y*, and *z*, (Figure 10) where the gravitational acceleration its directed along the negative z axis (note that this is a different convention than that used in the piecewise two-dimensional analysis). The wind field is approximated as time-independent, and can be written as a function of *x*, *y*, and *z* only. For a glider flying through this wind field, the wind as seen by an observer on the glider is $\mathbf{U}(\mathbf{r}(t))$ where $\mathbf{r}(t)$ is the glider's time-dependent inertial position vector. The velocity of the aircraft relative to the local air mass is therefore:

$$\mathbf{V}(t) = \mathbf{U}(\mathbf{r}(t)) - \dot{\mathbf{r}}(t) \tag{9}$$

To put these variables in aviation terminology, $\mathbf{V}(t)$, $\mathbf{U}(\mathbf{r}(t))$, and $\dot{\mathbf{r}}(t)$ are the time-dependent 3D vector generalizations of "true air speed", "winds aloft" at the location of the aircraft, and "ground speed" respectively. Dimensionless lift and drag are defined as before but can be written as time-dependent vectors: $\mathbf{l}(t)$ is the force per unit weight normal to the direction of flight, and $\mathbf{d}(t)$ is the normalized backward force in the direction of $\mathbf{V}(t)$. These force components together with the body force due to gravity are illustrated in Figure 11.

Since this analysis is primarily concerned with dynamic soaring in quasi-steady shear flows, it is useful to assume that the wind velocity only depends on altitude (*z*), and to define the *x* axis

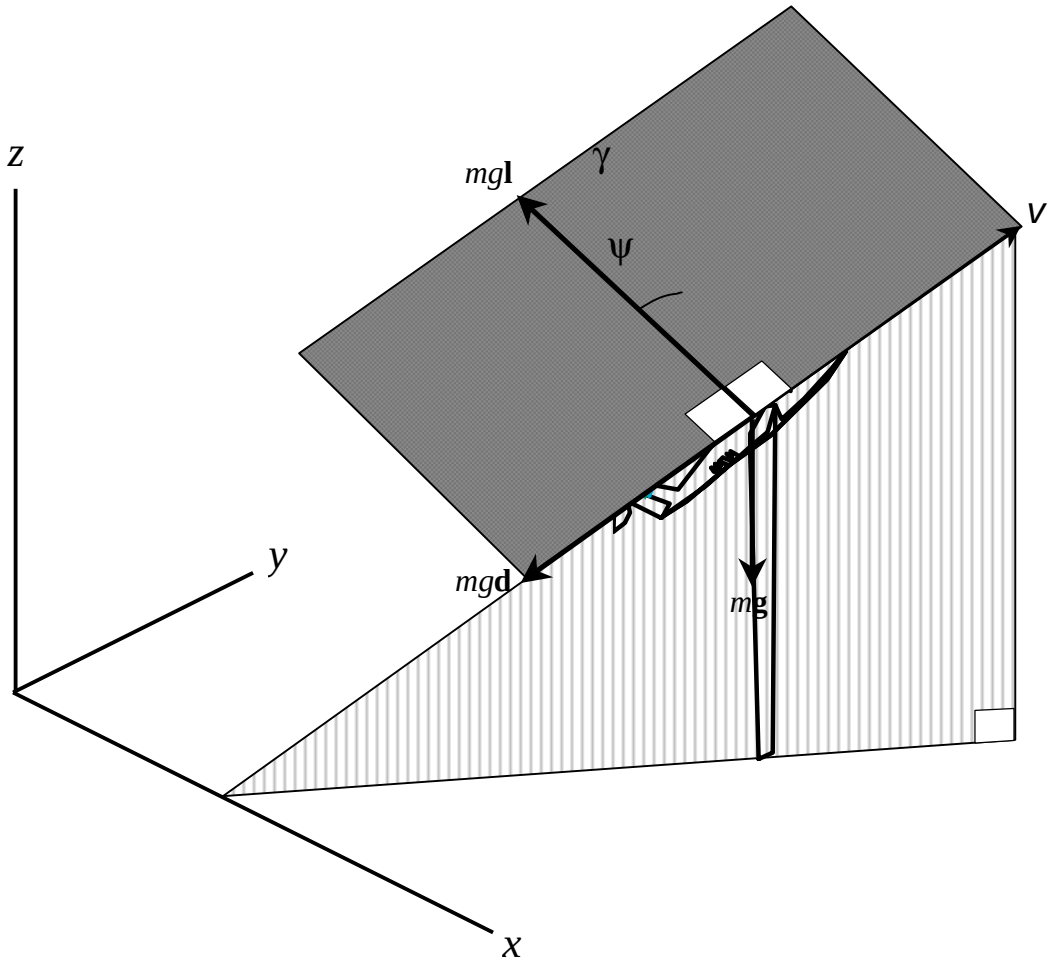

Figure 11. Components of force relative to inertial and wind-fixed frames.

of the inertial coordinates to be the direction the wind is blowing. The vector equation of motion in the inertial frame becomes:

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} - \frac{\mathrm{d}U}{\mathrm{d}z}\hat{\mathbf{x}} = -g(\mathbf{l} + \mathbf{d} - \hat{\mathbf{z}}) \tag{10}$$

To solve the three-dimensional equations of motion for an aircraft in a non-uniform steady wind, it is convenient to define a non-inertial "wind-fixed" reference frame (Figure 10). The V coordinate is in the $-\mathbf{V}$ direction, the γ axis is in a vertical plane and normal to $\mathbf{V}$, and the β axis is horizontal and normal to the other two axes. Projection of the components of the vector equation (10) onto the new wind-fixed axes yields the three equations of motion in that coordinate system:

$$\dot{V} + V\frac{\mathrm{d}U}{\mathrm{d}z}\sin\gamma\cos\gamma\cos\beta = -g(d - \sin\gamma) \tag{11}$$

$$V\dot{\gamma} - V\frac{\mathrm{d}U}{\mathrm{d}z}\sin^2\gamma\cos\beta = g(l\cos\psi - \cos\gamma) \tag{12}$$

$$V\dot{\beta}\cos\gamma - V\frac{\mathrm{d}U}{\mathrm{d}z}\sin\gamma\sin\beta = gl\sin\psi \tag{13}$$

Hendriks [1972] expresses these as non-dimensional wind-fixed equations of motion

$$\dot{V} = -V\frac{\mathrm{d}U}{\mathrm{d}z}\sin\gamma\cos\gamma\cos\beta - C_D V^2 - \sin\gamma \tag{14}$$

$$V\dot{\gamma} = -V\frac{\mathrm{d}U}{\mathrm{d}z}\sin^2\gamma\cos\beta - C_L V^2\cos\psi - \cos\gamma \tag{15}$$

$$V\dot{\beta}\cos\gamma = V\frac{\mathrm{d}U}{\mathrm{d}z}\sin\gamma\sin\beta - C_L V^2\sin\psi \tag{16}$$

where variables are normalized with respect to characteristic length and time scales, $L_c = 2m/rS$ and $t_c = \sqrt{L_c/g}$, respectively. From the definition of the wind-fixed coordinate system (Figure 10), the time derivative of the inertial position can be written:

$$\dot{x} = V\sin\gamma\sin\beta + U \tag{17}$$

$$\dot{y} = V\cos\gamma\sin\beta \tag{18}$$

$$\dot{z} = V\sin\gamma \tag{19}$$

Substitution of (19) into equation (14) and multiplying through by V yields the total energy equation for dynamic soaring (DS) in steady laminar shear:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\tfrac{1}{2}V^2 + z\right) = -\tfrac{1}{2}V^2\frac{\mathrm{d}U}{\mathrm{d}z}\sin 2\gamma\cos\beta - C_D V^3 \qquad (20)$$

The left-hand side is the time rate of change of useful kinetic and potential energy. The first term on the right side is the "DS term", and is the rate of energy gain (or loss) due to the wind gradient, and the second term is the rate of dissipative loss due to aerodynamic drag. Equation (20) can be used to optimize a generalized three-dimensional analog of the flight cycle originally proposed by Lord Rayleigh.

Any cycle that pumps wind energy into flight energy requires that the integral of the right-hand side of equation (20) be greater than zero. To harvest energy at all times from a positive vertical wind gradient, the term $\sin 2\gamma\cos\beta$ must be less than zero. This condition requires that for downwind flight, $(-\pi/2 \le \beta \le \pi/2)$, the aircraft must descend $(-\pi/2 \le \gamma \le 0)$. Likewise, for upwind flight ($-\pi \le \beta \le -\pi/2$ or $\pi/2 \le \beta \le \pi$), the aircraft must ascend $(0 \le \gamma \le \pi/2)$. This is the rule that seabirds discovered through evolution, and that Lord Rayleigh first documented.

One of the simplest cycles to consider (and as we shall see in Section IV, to fly) is an inclined circle. To gather energy from the wind at all times, the circle must be tilted about the y-axis of the inertial frame, so that the maximum altitude coincides with the maximum downwind position and vice-versa. To estimate the value of the gradient required for continuous circular flight, the DS pumping and aerodynamic dissipation terms can be integrated over one cycle from $\beta = -\pi/2$ to $\beta = \pi/2$. By ignoring the modulation of V and by letting γ vary as $(\pi/2)\cos\beta$, a bounding condition can be calculated for an inclined circle in uniform laminar shear:

$$\frac{\mathrm{d}U}{\mathrm{d}z} > \frac{3\pi}{2}C_D V \qquad (20)$$

By re-dimensionalizing equation (20) and substituting in realistic values for wing-loading (mg/S= 250 N/m^2), lift coefficient (C_L=1.0), and drag coefficient (C_D=0.1), the minimum gradient is estimated to be about 0.5/s. This exceeds the gradient typically observed within several meters of the ocean surface [Cone, 1964].

A similar approximation was worked out by Hendriks [1972], who graphically solved a two-state flight cycle that only considers an upwind climbing segment where angles were fixed at $\beta = \pi$, and $\gamma = \pi/2$, and a downwind diving segment with $\beta = 0$ and $\gamma = -\pi/2$. Flight segments connecting these two were not considered, but domains of maneuverability in the $\dot{z}, \dot{V}$ phase plane were calculated assuming most of the cycle is spent on the two longitudinal legs. There are two steady aerodynamic states (one on each leg), and the aircraft "chatters" between them, with an intermediate mean state that represents a stable "chatter control" state. Hendricks determined a necessary condition for sustained flight is

$$\frac{\mathrm{d}U}{\mathrm{d}z} > 2C_D V \qquad (21)$$

The shear gradient required for the more realistic inclined-circle cycle with continuous variation of states is, not surprisingly, significantly greater.

To estimate the maximum velocity attainable for a dynamically soaring aircraft, the integrated DS term can be equated to the dissipation term over one cycle. Under ideal dynamic soaring conditions, the thickness scale of the shear boundary is insignificant compared to the scale of the aircraft, and there is an instantaneous jump in wind speed (ΔU) across the boundary. The equation can be written:

$$V_{\max} = K(\Delta U)(L/D) \tag{22}$$

where K is the reciprocal of the factors in the inequalities (20) and (21). These values provide approximate upper and lower bounds for a simple circular cycle ($0.2 < .K < 0.5$). L/D is the lift-to-drag ratio, which for a clean sailplane can exceed 50. By executing circular DS cycles across a 25 mph shear boundary, such a plane should be able to achieve airspeeds in excess of 250 mph.

Field observations of actual albatross flight reveal dynamic soaring patterns that are much more complex and sophisticated than those analyzed in this section. Clearly the real world is more complicated than our idealized equations of motion are able to capture. Nevertheless, the models considered thus far provide the insight and understanding required to develop the computational methods discussed in the next section.

III. COMPUTATIONAL MODELING

In reality, lift and drag are much more complicated functions of an airplane's angle of attack, sideslip angle, and other details of its state in three dimensional space, and the body is subject to torques about its three axes as well as three components of force. To realistically simulate the flight of a bird or airplane requires that its equations of motion be solved with six degrees of freedom ("6-DOF") using a more accurate nonlinear aerodynamic model. Fortunately there are various codes available that are appropriate for this problem. Special versions of these codes were created for this project, so that dynamic soaring could be more realistically modeled. A realistic UAV sailplane model was developed based on NASA flight tests of a full-scale Schweizer SGS 1-36 sailplane and scaling it to a model with an eight-foot wingspan. The various codes are summarized in this section, as well as results showing the feasibility of dynamic soaring for a realistic UAV model.

A. LaRCsim

Early in this project, LaRCsim was identified as the code of choice for the full 6-DOF dynamic soaring simulations and optimization because it was a stable code and the source was available from NASA and could be modified or linked to user-defined control routines. LaRCsim was developed at NASA Langley Research Center as a computational flight dynamics model for debugging aircraft flight control laws, and is a direct descendent of BASIC, a set of FORTRAN simulation routines written at NASA Ames [McFarland, 1975]. These were further developed by Jackson [1995] for the Naval Air Warfare Center, and then rewritten in ANSI C to take advantage of the RISC architecture of workstation-class computers of the 1995. It solves a full set of equations of motion for rigid-body flight, with all six degrees of freedom [*e.g.* Stevens

& Lewis, 1992]. It includes full earth geodesy, gravity, and atmospheric models, and can be applied to low-earth orbital flight as well as atmospheric flight.

LaRCsim has a number of features that made it attractive for investigating and developing dynamic soaring strategies with realistic aircraft and atmosphere models. It models moments as well as forces, so vehicle properties must include a moment of inertia tensor and aerodynamic moment parameters in addition to mass and aerodynamic force parameters. It also models moments and forces due to engine thrust, and moments and forces due to landing-gear drag. Unlike the simple Cartesian three-dimensional numerical analysis developed in the previous section, LaRCsim models flight using a rotating oblate spheroidal earth using both geocentric and geodetic coordinate systems, allowing cross-country flight to be simulated in a way that keeps track of the actual curvature of the Earth, and accounts for the Coriolis effect. An atmospheric model is built into the code, using a cubic spline function (in a lookup table) as a smooth interpolative fit to the 1962 standard atmosphere that tabulates air density up to 75,000 feet. Higher order spherical harmonic gravity terms are included to account for the nonspherical figure of the Earth. For the angular orientation, quaternions are used to avoid the pole singularity when the pitch angle is vertical.

One feature that made LaRCsim particularly useful for the present application is its ability to allow user-defined local air mass velocities that include both steady and transient (gust) components. This allows shear gradients to be modified at will for the analysis of dynamic soaring in steady laminar flow, or of gust soaring in turbulence, or of any combination of these and other types of wind fields. A flight session can be initialized to start anywhere in the world, and the aircraft can be piloted using keyboard and mouse interfaces. A very important feature is that flight data can be output to a file for plotting and post-flight analysis.

Unfortunately, LaRCsim suffers from the drawback of lacking visualization capabilities, using only a text-based user interface (using *curses*). Another shortcoming is the fact that certain flight parameters are hardwired and the code must be recompiled whenever they are changed. Most problematic is its dependence on Silicon Graphics GL workstation debugger symbol tables to access static and global variables for display and recording. These important features did not survive porting to other workstations. While LaRCsim enabled the first full 6-DOF dynamic soaring simulations, its lack of platform independence and other limitations made it necessary to choose between rewriting LaRCsim or finding another computational tool. Fortunately, the world of open-source code development allowed the second choice to be viable.

B. FlightGear

In late 1997, an open source flight simulation project called FlightGear incorporated LaRCsim as its default computational flight dynamics model (FDM). The FlightGear Flight Simulator (FGFS) developers ported LaRCsim to a variety of other platforms and created a 3D graphics interface using OpenGL for the purpose of maintaining a system-independent simulator. Because FGFS source code is available under the GNU General Public License, it has attracted a variety of professionals from research and academic environments to its development team, which has greatly enhanced its quality, features, and usefulness over the past few years. By keeping the code open source, it is much more useful than commercial simulators which are proprietary and lack extensibility necessary to be used for research. Moreover, the open-source licensing allows inter-organizational and international collaborations to emerge. Over 50 code developers, engineers, scientists, and hobbyists from around the world have contributed to the project. One of the greatest advantages of using FGFS for research is that it will not become

"frozen" when the project is over. As new software libraries, improved hardware, and more data become available, there will be a developer somewhere who will incorporate them into FGFS.

For a research project such as this, the platform independence of FGFS is important. The current release (7.10) runs on all distributions of Linux, Windows NT/2000/XP, Windows 95/98/ME, BSD UNIX, SGI IRIX, Sun-OS, and Macintosh. It consists of more than 150,000 lines of C/C++ code. By using OpenGL, it produces high quality 2D and 3D graphics with frame rates of over 70/sec on a Dell 610 workstation with a high-end 3D graphics card. By using a CVS repository and modern software engineering techniques, the individual researcher can maintain version control and merge her local changes and additions with those in the current development version. FlightGear is managed using the SourceForge collaborative software development platform.

Much of the development of FGFS is devoted to the "bells and whistles" that would be out of reach of a developer in a ordinary research environment. For example, terrain rendering is based on more than two gigabytes of U.S. Geological Survey data (based on satellite imagery) that can be downloaded from the internet for any location on Earth. This capability can be seen in Figure 12, which shows a frame from a special version of FGFS (developed for this project) that flies a hardwired lemniscate pattern over Albuquerque. To emphasize the main advantage of using FGFS, if the USGS data format is changed or goes to a higher resolution, in all likelihood this change will be nearly transparent to the researcher because one of the many developers will adapt the code, which is a "living" entity. Other features include a "sky dome" in which the sun, moon, planets, and stars are in their proper positions, texture-based scenery which includes user-

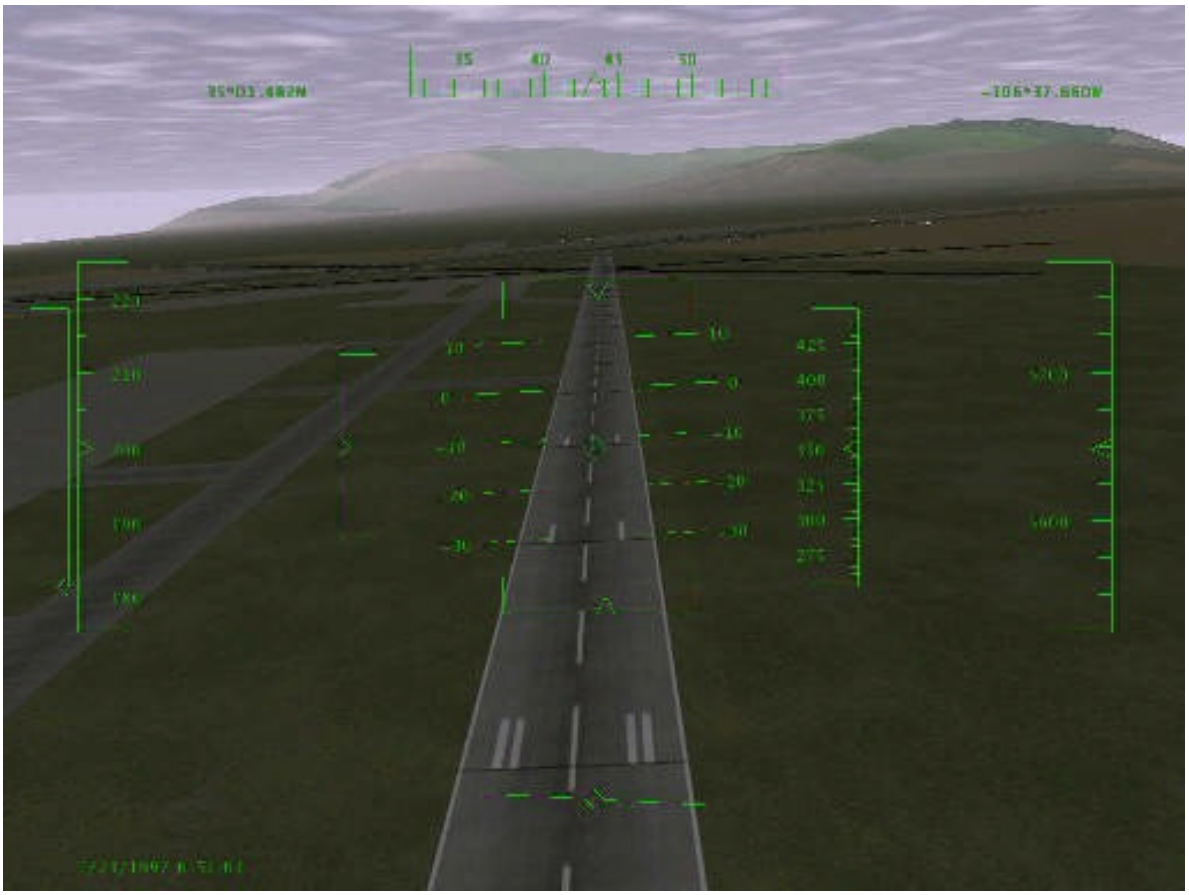

Figure 12. Frame capture of simulation over Albuquerque, flying toward Sandia Mountains.

defined haze density and cloud layers, a heads-up display (HUD) that provides graphical display of flight data, a rudimentary autopilot that includes heading and altitude hold modules, and audio support, which allows users to customize sound feedback based on flight parameters such as altitude, airspeed, or total energy. While these features are not required for running simulations, they are very useful for "pilot in the loop" tests, for developing intuition, for replaying flights, and for demonstrations.

Over the three-year course of this project, the rate of improvement to FGFS has been spectacular, and other researchers are also making use of it and aiding its continued development. This network of researchers gives FGFS even more value to the individual worker, because it provides a built-in technical support group and field of potential collaborators For example, a University of Illinois at Urbana-Champaign (UIUC) aeronautical engineering group is using FGFS to develop an autonomous "smart icing system" that will use a neural network to improve flight safety in atmospheric icing conditions by sensing changes in an aircraft's performance characteristics and assist the pilot in responding properly [Sehgal, 2002]. The U. Illinois team has added 15 new aircraft models to the FGFS database, including a reconfigurable aircraft model that includes aircraft icing effects. This capability of incorporating a dynamically changing aerodynamics model into a flight simulation can certainly be applied to simulations of soaring birds as well. This would allow unprecedented simulation fidelity of birds (or dynamically changing UAVs), which until now have been modeled as static rigid bodies. Moreover, adaptive reconfiguration strategies could be investigated for application to smart autonomous UAVs.

Other research groups are also using FGFS to aid the development of autonomous UAVs. A group at University of Wales, Aberystwyth, is building a virtual environment to simulate the flight of a lighter-than-air intelligent robot ("aerobot") to be used in planetary exploration. FGFS is being coupled to a computational fluid dynamics (CFD) code to generate realistic meteorological conditions, and the digital terrain model is derived for Mars from the Mars Global Surveyor orbiter's laser altimeter. FGFS is also being used by the Simon Fraser University Aerial Robotics Group to develop another autonomous UAV, the FireMite.

The high degree of modularity of the FGFS code encourages researchers develop their own versions while minimizing conflicts with other parts of the code. Because the modules can be compiled independently, the entire code does not need to be re-built whenever a change is made to a particular module; it only needs to be re-linked. The default flight dynamics model for FGFS has been a version of LaRCSim, but various groups have contributed their own FDMs that can be chosen at run time. For example, the UIUC group has written and integrated its own UIUC-Aearomodel code as a wrapper interface around LaRCSim, and a newer FDM, called JSBSim has been written that incorporates the flight dynamics and other functions of LaRCSim, but by taking advantage of the object-oriented capabilities of C++. JSBSim has emerged as the new default FDM for FGFS, and for that and other reasons was chosen as the FDM for the current research project.

C. JSBSim

In addition to its other disadvantages, LaRCsim was unable to read flight characteristics and aerodynamic properties of a specific aircraft from a data file. To model a different kind of aircraft, the code had to be modified and recompiled. For some applications, this is acceptable, but it makes LaRCSim very difficult to use as a research tool. The UIUC approach was to write a wrapper interface that allowed other aircraft models to be implemented, but this method still

has its limitations. Over the past few years, a new open-source FDM was written in C++ as an object-oriented code with emphasis on configurability. JSBSim also takes advantage of the C++ features of polymorphism, inheritance, and encapsulation. Because of its flexibility and use of configuration files and scripting, it has just recently replaced LaRCsim as the default FDM for FlightGear. A standalone version of JSBSim was developed simultaneously with the FlightGear version, which allows it separated from the computationally intensive components of FlightGear for use in fitness evaluation and flight behavior development. JSBSim, together with FGFS, is ideally suited for this research project.

The stated goals of JSBSim are: 1) to allow different aircraft to be modeled without writing new code, 2) to be open source and run on any platform, 3) to be useful as a tool in studying flight dynamics and as a showcase for some of the equations, and 4) to be executable in both standalone mode and integrated with FlightGear (or any other flight simulator). The last goal also provides a strong motivation to use JSBSim, not only because it allows the flight dynamics to be decoupled from the computationally intensive graphics rendering, but because it can be integrated with other tools. For future work involving robotics applications, it will become important to port the knowledge and tools developed to Sandia's existing robotics simulation tools such as Umbra [Gottlieb *et al*. 2001]. Since both JSBSim and Umbra were designed from the start to be modular and easily integrated, it makes sense to work toward integrating them with one another.

One of the most important attributes of JSBSim is the capability to accept the specifications for a flight control system of any level of complexity by creating a user-defined configuration file. These files are in XML format and contain a table of aerodynamic stability derivatives, as well as mass properties (moment of inertia tensor and center of gravity), landing gear, propulsion system, and control surfaces (ailerons, elevators, rudder, flaps, and spoilers). Aerodynamic coefficients can be expressed as constants or as 1D or 2D tables to account for nonlinearities and dependences, allowing much more realistic behavior than the numerical models described in Section II of this report can provide.

Another feature of JSBSim that is useful for this project is that it can simulate a complex flight control system using information provided but the user in the aircraft configuration file. The simplest control system would be a linear scaling factor that converts the movement of an input device (*e.g.* mouse or joystick) to a control surface deflection command. JSBSim allows a much more sophisticated control system that can be assembled using elements such as digital filters, scheduled gains, and sensor inputs. This can be extended to calls to user-defined genetic programs or other evolutionary methods. The flight control system is built up of sequential commands, starting with the pilot control input (typically by using a 3-axis joystick for rudder, aileron, and elevator control). In the simplest case, the only thing between the joystick and the control surface would be the linear scaling function. For trim, stability, and autopilot control, various means of feedback can be employed. In a more advanced control system, the joystick output can be combined with values derived from sensor outputs such as indicated airspeed, altitude, pitch angle, heading, GPS coordinate, magnetic declination, or acceleration. As part of the chain of control, the sensor outputs can be combined with one another to yield physically meaningful state values such as total energy. In this case, temperature, altitude, and barometric pressure can be used to convert indicated airspeed to true airspeed, which can then be squared, halved and added to altitude to give instantaneous total energy relative to the air mass. These derived values can be used to provide situational awareness to the control system, or can be compared to threshold values to switch the control system from one mode to another (as in the case of policy tables to be described in Section V).

To make JSBSim into a useful tool for investigating dynamic soaring and developing flight strategies, it was necessary to modify it to simulate various three-dimensional vector wind fields, such as the laminar boundary flow analyzed in Section II. This turned out to be straightforward, as is the implementation of a turbulence model for the investigation of extraction of energy from gusts, which will be the subject of future work. Another necessary modification was to output data to a file in a format that could be read and plotted by a graphing package such as Microsoft Excel, and to be able to import the data from the log file into FlightGear so that a given flight could be replayed--even if it was originally generated by JSBSim in standalone mode. This allows JSBSim to be used to do a fitness calculation as part of a genetic program without graphics, with the best flights to be saved and played back or plotted for analysis. Using JSBSim for developing flight behaviors for a UAV also required the development of a realistic flight model for a small glider, described in the next sub-section.

D. UAV model characteristics

For the full six-degree of freedom flight simulations, it was necessary to develop a more realistic aerodynamic model than the one that was used to model albatross flight in the previous section. Fortunately, extensive wind tunnel and flight data are published for a modified

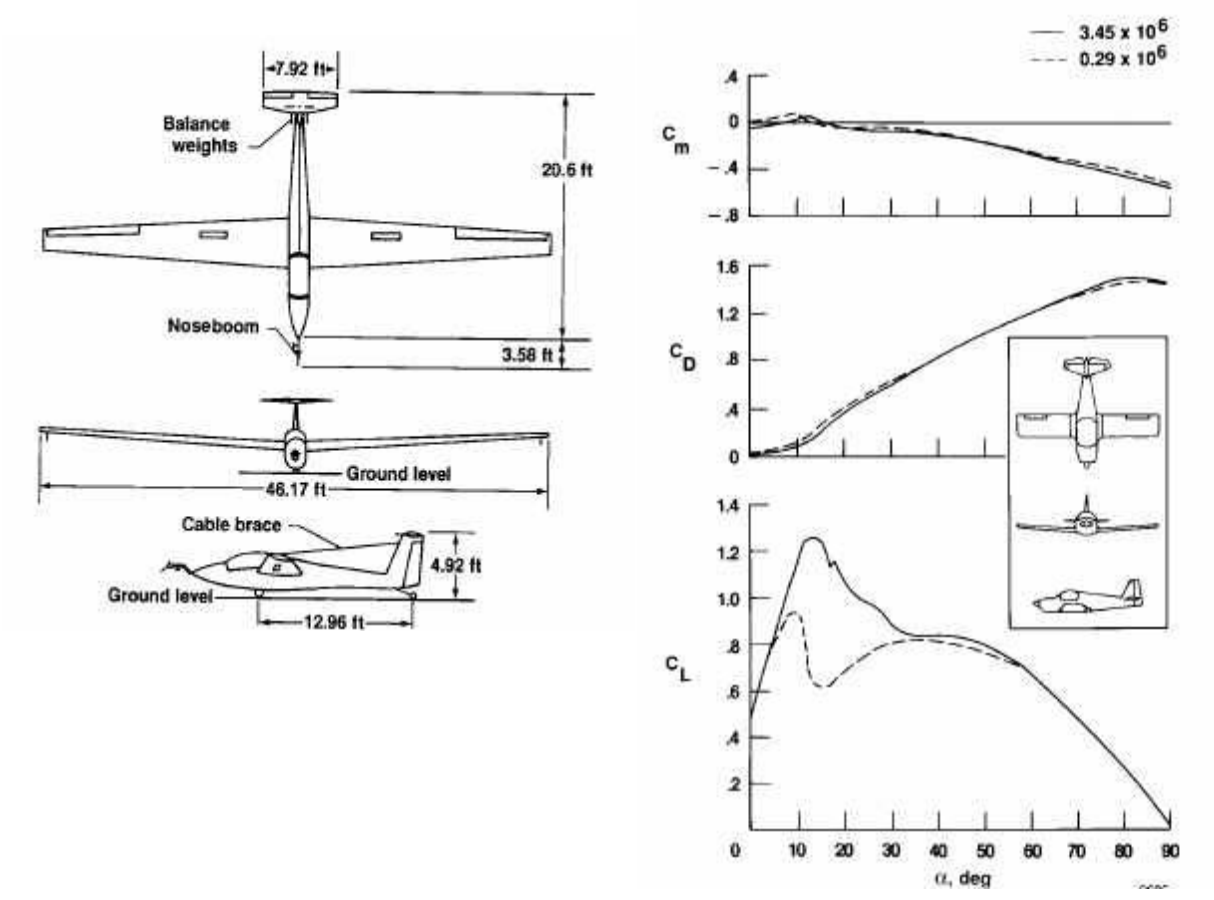

Figure 13. Modified SGS 1-36 sailplane, and pitching moment, drag and lift coeffiecients as a function of angle of attack for two Reynolds number conditions [from Sim, 1990].

.

Schweizer SGS 1-36 Sailplane [Sim, 1990]. This is a single-seat, T-tail unpowered aircraft that is used commercially as an advanced trainer and has many characteristics that are typical of full scale and model sailplanes. Force and moment data were collected using wind-tunnel tests at NASA Langley, and flight data were obtained from flight tests at NASA's Dryden Flight Research Facility. Pitching moment, drag, and lift coefficients from this report are plotted as a function of the angle of attack in Figure 13. The instrumentation for the flight tests consisted of 26 channels of data that were used for the derivation of aerodynamic stability derivatives. The flight data included three-axis accelerometers and rate gyroscopes, a pitch and roll attitude gyroscope, a pitot-static port for static and dynamic pressure (for altitude and airspeed), and control surface position sensors. The moment of inertia tensor was determined with a physical measurement on the ground. It is notable that the modified glider used for these measurements had a significantly lower aerodynamic performance as measured by the lift-to-drag ratio, because the test plane was extensively modified. Thus, using this as a model in JSBSim tends to underestimate the performance of a real glider.

Table 3: Characteristics of Schweizer SGS 1-36 Sailplane

Quantity	Value (rad^{-1})
Lift due to elevator deflection	0.342
Drag at zero lift	0.0007
Side force due to sideslip angle	-0.285
Side force due to aileron	-0.0456
Side force due to rudder	0.188
Roll moment due to sideslip angle	-0.0513
Roll moment due to roll rate (roll damping)	-0.47
Roll moment due to yaw rate	0.15
Roll moment due to aileron	0.252
Roll moment due to rudder	0.0046
Pitch moment due to angle of attack	-0.573
Pitch moment due to pitch rate	-9.0
Pitch moment due to angle of attack rate	-5.2
Pitch moment at zero angle of attack	0.0
Pitch moment due to elevator deflection	-0.1261
Yaw moment due to sideslip angle	0.017
Yaw moment due to roll rate	-0.18
Yaw moment due to yaw rate	-0.025
Pitch moment due to aileron	0.0115
Yaw moment due to rudder	-0.074

.

The aerodynamic properties are defined by the partial derivatives of various coefficients as a function of flight state variables. Coefficients include axial, normal, and side forces, rolling, pitching, and yawing moments, lift, and drag. Flight state variables include roll, pitch, and yaw rates, velocity, angles of attack and sideslip, pitch and bank angles, and control surface positions. The significant derivatives were extracted from the NASA report by reading them off the graphs (interpolating and extrapolating when necessary). These were input into a JSBSim configuration

XML file that is read at runtime, and are listed in Table 3. The full-scale model was "test-flown" using FlightGear's "pilot-in-the loop" capability by two experienced sailplane pilots. They used a joystick for control inputs and viewed the 3D graphics using a large wall-projection system. The aerodynamic stability derivatives were tuned somewhat based upon their comments. The major criticism by the pilots was not about the accuracy of the flight performance, but the about the lack of certain sensory cues experienced when flying real gliders. The lack of a side view made it very difficult for one pilot to maintain situational awareness and control the model's attitude. The other pilot was more concerned with the lack accurate changes in wind sound, especially the increased volume of wind noise during high-sideslip-angle flight. To create a realistic 8-foot UAV model, the full-scale SGS parameters were appropriately scaled. The values in Table 3 are dimensionless, so they remained the same. The lifting surface spans were linearly scaled; areas, masses, and moment constants were scaled by powers of 2, 3, and 5, respectively. Flight data from this scaled model were collected during dynamic soaring conditions, as summarized in the next sub-section.

E. Dynamic Soaring simulations

A special version of FlightGear was created for this project, allowing flight in arbitrary wind fields to be modeled. Rudimentary models of thermals, slope lift, and turbulence were tested, but the main focus was laminar shear for developing dynamic soaring strategies. A logarithmic boundary layer was implemented in an attempt to determine the difficulty of dynamic soaring in typical low-altitude marine atmospheric conditions. The most useful wind field was a simple velocity step function, which simulates the leeward hill slope separation flow in which radio-controlled dynamic soaring has actually been successfully flown. Flight under such conditions requires a significant degree of practice and pilot skill (the experienced test pilots were not immediately able to do it). The most successful strategy is to attempt to fly in an inclined circle,

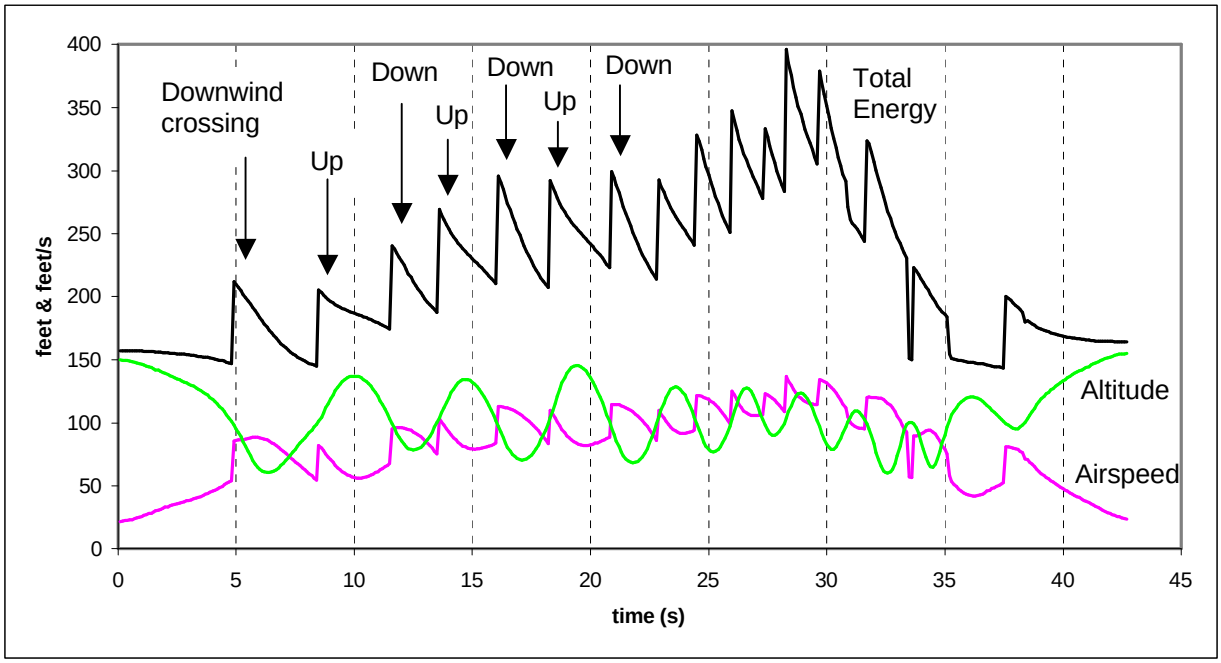

Figure 14. Dynamic soaring simulation using 8-foot scaled Schweizer model with modified version of JSBsim.

similar to the one numerically analyzed in section II(d), keeping the plane of the orbit as close as possible to the plane of the shear layer. Tight turns are required necessitating very steep bank angles. This is consistent with the perturbation analysis of Hendriks [1972] from which an optimal bank angle of 55^{o} from the horizontal was derived, and is also in agreement with the observations by Idrac [1931] of albatross soaring in the shear boundary layer at sea. Data were logged for flights under various dynamic soaring conditions and can be plotted or played back using the replay option of the modified version of FlightGear. Figure 14 is a plot of airspeed and altitude logged from a flight in a shear layer with a velocity jump of 33.8 feet/sec for the scaled 8-foot Schweizer sailplane. The shear boundary is horizontal at an altitude of 100 feet, and the airspeed increases by a value close to the velocity difference for the optimal upwind and downwind shear crossings. Total energy is calculated and put into altitude units, and is seen to be increasing with time. This data is quite similar to that collected from an actual flight test (see Figure 20).

Further refinements in JSBSim, however, will be required to model the detailed interactions between steep wind velocity gradients and an aircraft under these conditions. In this case the wind gradient is so steep that when the glider passes through the shear layer, different parts of the plane experience different airspeeds. In reality, this would result in a torque on the aircraft, but in the current simulation the wind is only defined at a single reference point, and the glider effectively experiences a uniform air velocity at any instant in time.

IV. FLIGHT TESTS

Fortuitously, the first radio-controlled dynamic soaring flights were successfully flown at about the same time that this project was initiated. The pioneer of this mode of flight was Joe Wurts, who—in addition to being an aeronautical engineer at Lockheed-Martin Skunk Works and specializing in micro-UAV development—is a National and World Champion radio-controlled sailplane pilot. The explanation for his "backside" flights was initially met with considerable skepticism both within and outside of the recreational radio-controlled flying community, but over the past few years more r/c pilots have learned how to do it. "DS" has since become somewhat of a fad because of the very high speeds that can be attained (one of Wurts' flights was unofficially clocked at 156 mph using a radar gun), greatly exceeding those normally achieved in sustained unpowered flight. Based on his achievements and technical expertise, Joe Wurts was asked to be a collaborator on the flight test portion of this project.

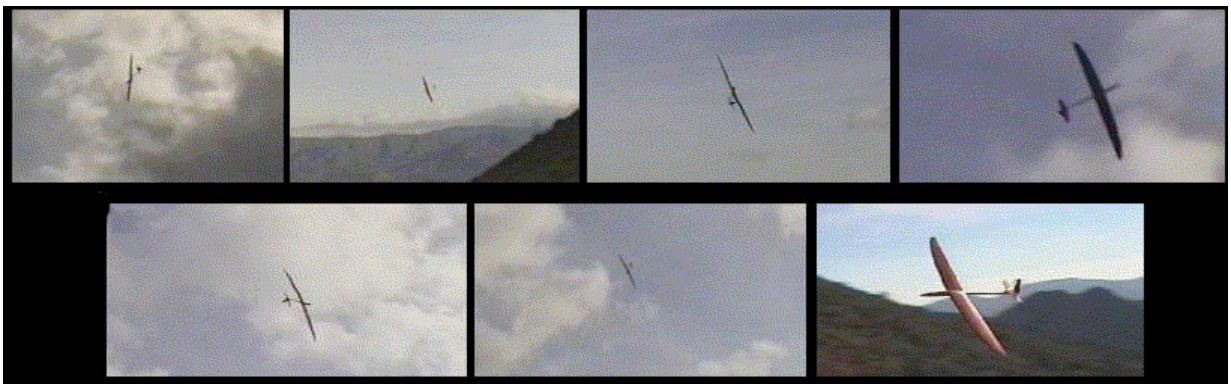

Figure 15. Dynamic soaring flight at Parker Mountain, Calif. (photo: Paul Naton).

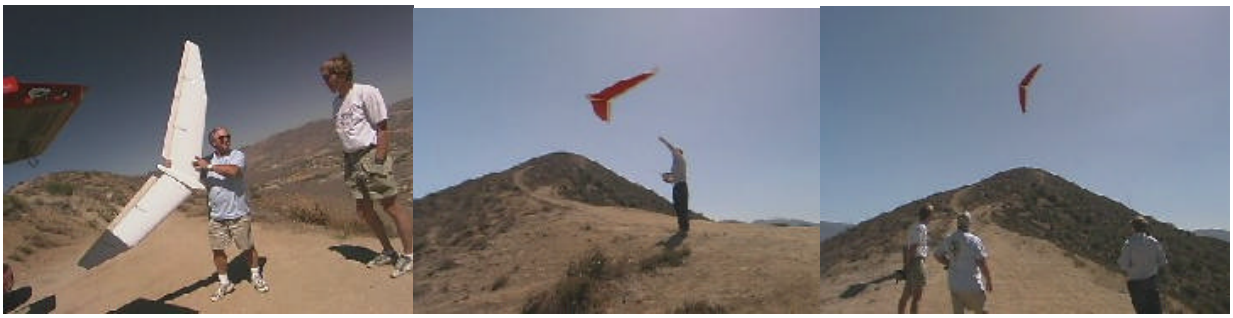

Figure 16. Sandia test glider, the "DS Beast" is used to collect the first airborne data during a dynamic soaring flight at Parker Mountain, California on Nov. 16, 1999. *Left*: Plane Builder Pat Bowman holding glider with P.I. *Center*: Pilot Joe Wurts launching into wind. *Right*: Glider executes upwind dynamic soaring run on leeward side of ridge (Photo: Dave Reese).

A. Dynamic Soaring in a Shear Boundary

Hendriks [1972] was also the first researcher to recognize the possibility of dynamic soaring with a radio-controlled glider, but his attempts to demonstrate it in wind shear over flat terrain were unsuccessful. Wurts'success was due to his discovery was that the wind velocity field on the leeward side of some recreational slope-soaring hills was ideal for dynamic soaring of model-scale sailplanes. This is due to a separation of the flow fields when a prevailing upslope wind reaches the top of a hill or ridge and does not mix with the stagnant air on the leeward side. Under certain conditions the flow fields remain separated for a long distance down wind, and a stable, very steep velocity gradient exists over a large enough area for radio-controlled sailplanes to maneuver. Parker Mountain is a popular southern California flying location where such conditions are common. It is actually a saddle ridge that is oriented perpendicular to the prevailing onshore wind. The topography rises at a constriction at the downwind end of a large valley, so wind velocities tend to be high and consistent. Moreover, the orientation of the ridge lends itself to solar heating of the uplifted air parcel, causing it to be locally unstable on the windward side. As air passes over the ridge the temperature profile becomes inverted because the stagnant downwind air mass is colder. Under ideal conditions, this temperature inversion tends to damp out any shear instabilities that form in at the flow boundary, and it remains extremely sharp and suitable for dynamic soaring. Figure 15 shows a sequence of images of a dynamic soaring flight by Joe Wurts at Parker Mountain.

B. EPP foam sailplanes

Another fortuitous development of the late 1990s was the pioneering of the use of expanded polypropylene (EPP) in the manufacture of cheap, very durable, model aircraft. Expanded polystyrene (*i.e.* Styrofoam) had been a popular model aircraft material for a number of years because of extremely low cost and manufacturing ease, but suffers several drawbacks. Most significant is its relatively low elastic limit, which allows it to break upon impact or by significant point loading. Another problem is that its preserves surface deformations such as

dents and scars, so even without a catastrophic crash Styrofoam airplanes age poorly. The lack of durability of this material greatly offsets the savings due to its low cost.

Expanded polypropylene, on the other hand, has mechanical properties that make it far superior in terms of durability, which ultimately keep the cost much lower because replacement expenses are minimized. It's lower bending and torsional stiffness allow it to recover from severe deformations and impacts, and it tends to bounce rather than break. One of the pioneers of EPP foam for r/c aviation applications, Pat Bowman, was given the contract to build three test planes ("the DS Beast", see Figure 17) for this project, which was outfitted with instrumentation and a data logging capability. It was with this plane that we collected the first in-flight dynamic soaring data.

C. Data Logging

A set of data loggers (Figure 17) and control software was designed and built for this project by Jennings Engineering, Inc, with the following specifications:

Enclosure Size

Width: 3.70" (94mm)
Length: 4.75" (120mm)
Height: 1.34" (34mm)

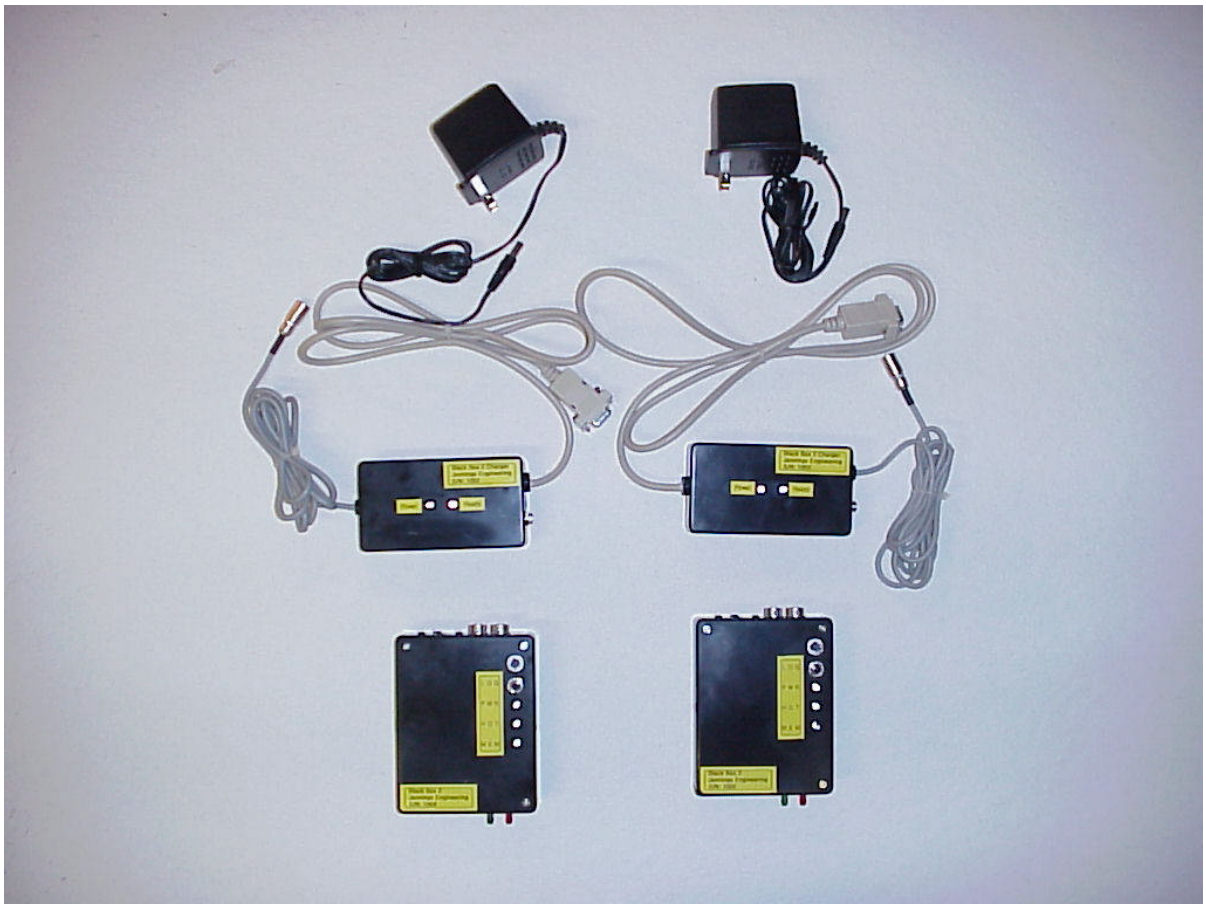

Figure 17. Jennings custom data loggers with power supplies and battery chargers.

Weight: 8.4 oz. (240 grams)

Battery Capacity

Operating Life: Approximately 2 hours (5 hours with charger connected)
Recharge time: Approximately 4 hours (data logger off)

Storage Capacity

Memory: 4 MB battery backed up static ram (3.90 MB available for data storage)
Recording time: Approximately 24.5 minutes total (at the default data recording rates)

Recorded Parameters and Rates

Acceleration: 3 axis, +/- 50 g full scale, 100 samples/second
Angular rate: 3 axis, +/- 480 deg/second full scale, 100 samples/second

Magnetic fields: 3 axis, +/- 1.5 gauss full scale, 20 samples/second
Dynamic pressure: 0.7 psi full scale (approximately 200 mph), 20 samples/second
Static pressure: 15 psi full scale, 20 samples/second
Servo commands: 8 channels, 20 samples/second

GPS: Latitude/Longitude/Altitude/Course/Ground Speed, 1 sample/second

D. Flight Data

The data logger hardware and software, designed and built under contract, includes 3-axis accelerometers, 3-axis gyros, magnetometers, dynamic and static pressure sensors, GPS, and control surface data commands, with 4 Mbytes of static ram for collection of about 25 minutes of flight test data. Logged data can be converted to airspeed, altitude, and total energy, and 6 degree-of-freedom position and attitude histories can be extracted. We have collected data for three test flights in dynamic soaring conditions, and have analyzed the data to show unambiguously that dynamic soaring can be used to sustain flight in a fundamentally different way than static soaring

Figures 18-21 show four selected one-minute intervals of the full 22-minute record of a DS Beast II test flight on Nov. 16, 1999 at Parker Mountain, California. Data from the onboard data logger are plotted with notations that describe events and flight patterns. GPS data are excluded from the plot. Total energy is plotted in units of altitude, derived from true airspeed (corrected for density altitude) and altitude from pitot-static system. In the upper graphs the vector sum of the magnetic field is plotted after calibration and removal of the B field induced by the circuitry of the logger. Three-axis accelerometer data are shown in blue. For the lower graphs, control inputs are derived by unmixing the recorded pulse-code modulated signal to servos on four control surfaces. No independent yaw control was used in this test flight. Three-axis gyro data are plotted with vertical offset. In order to help interpret the recorded data, notes were also logged during the 22-minute test flight (Appendix C).

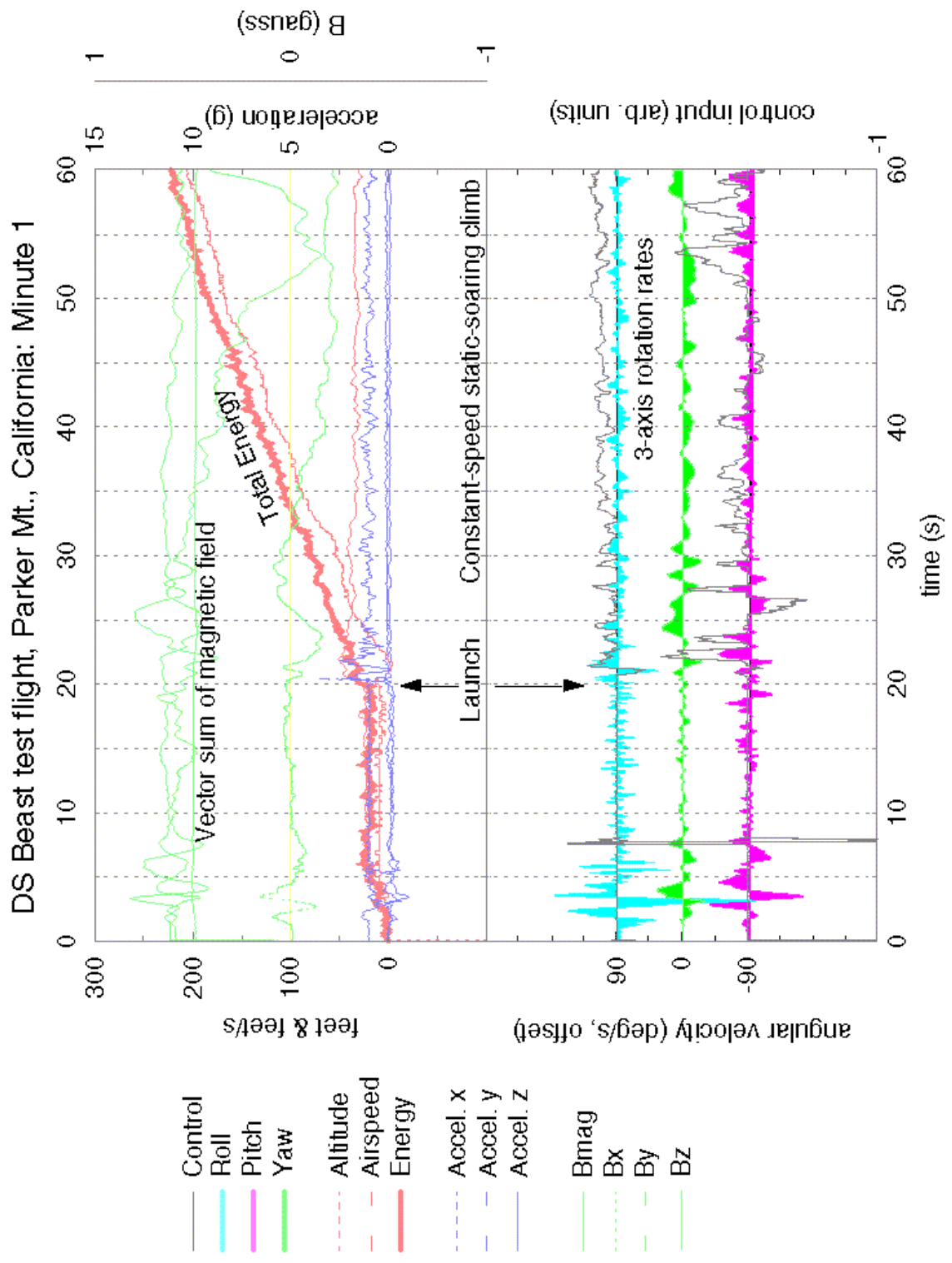

Figure 18. First minute of flight, showing data for launch and climb-out.

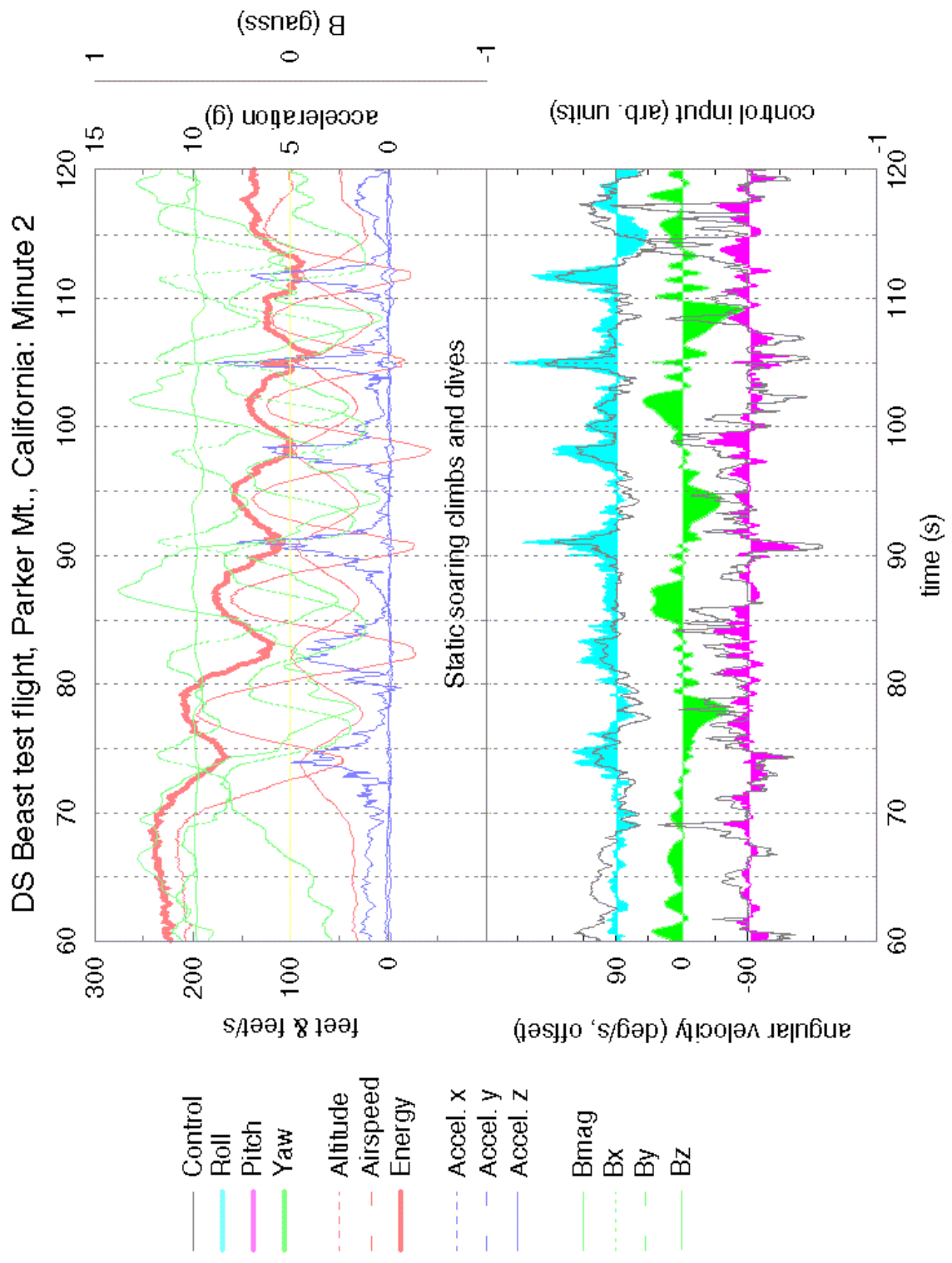

Figure 19. Second minute of flight, showing data for *static* soaring climbs and dives.

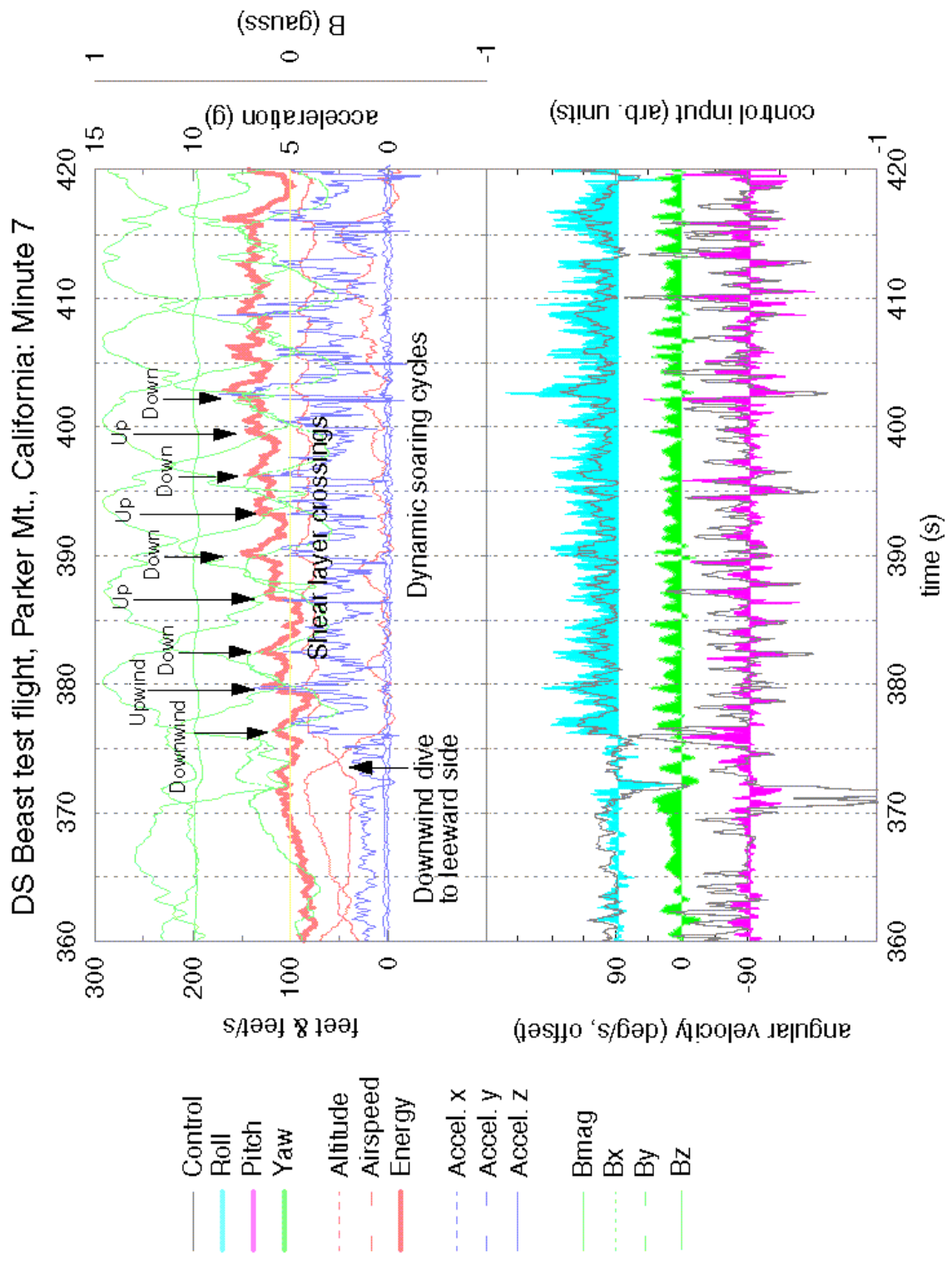

Figure 20. Seventh minute of flight, showing data for *dynamic* soaring shear crossings.

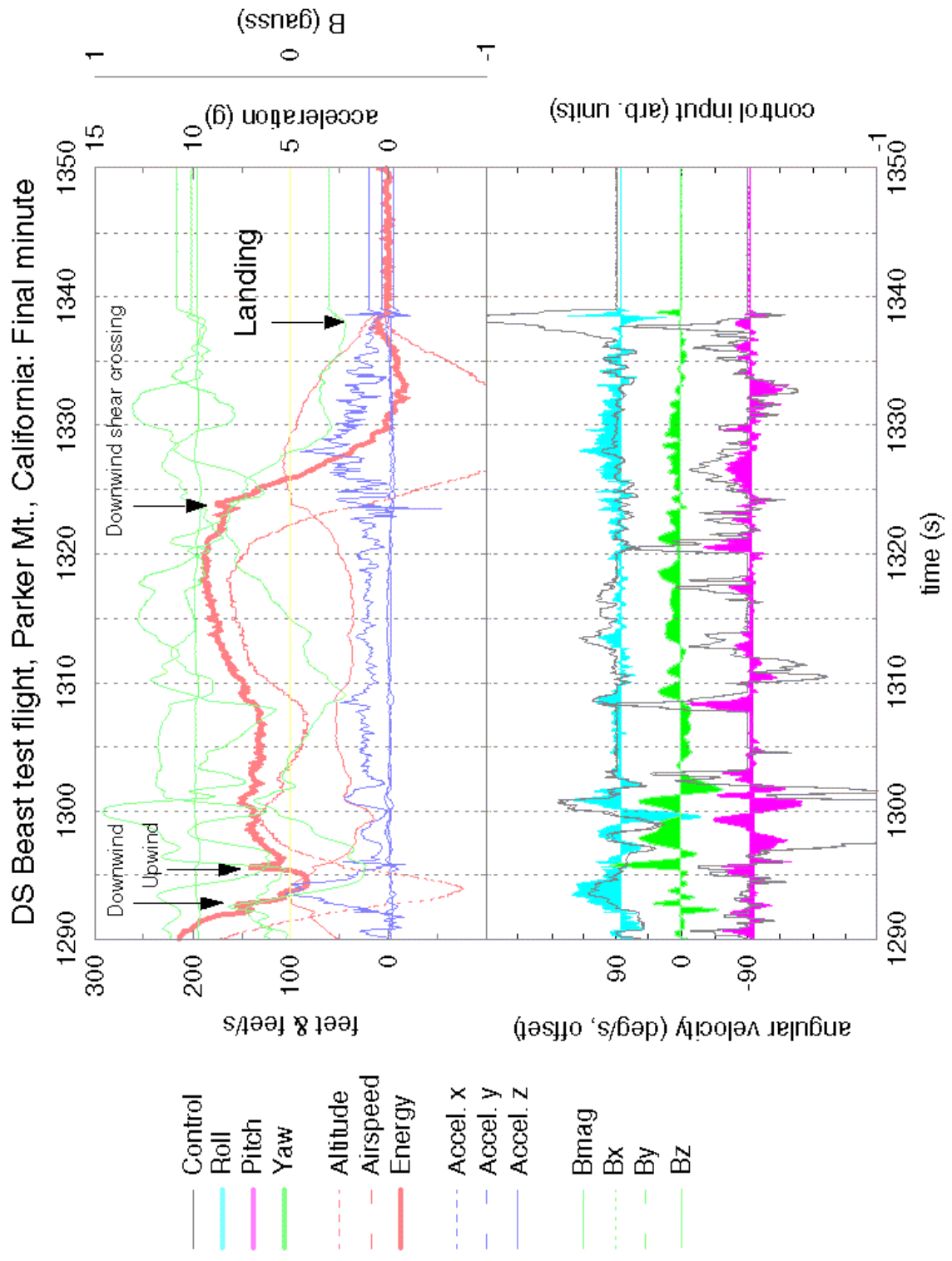

Figure 21. Final minute of flight, showing shear crossings and landing.

For about 20 seconds prior to launch, the aircraft was held steady and pointed into the wind above the shear layer to record the wind speed as an estimate of ΔU, about 20 feet/sec (Figure 18). For the first minute the aircraft was held at constant airspeed and was flown in slope lift on the windward side of the ridge, resulting in a steady climb. This was followed by a series of climbs and dives on the windward side to record the flight characteristics of the sailplane (Figure 19). The variation in altitude during these 8-second oscillations was between 100 and 200 feet, and the airspeed varied between about 30 and 90 feet/sec. Several times during the high-speed, low-altitude phase of the cycle, the normal (z) acceleration exceeded 7-8 g. These high velocities and accelerations led a high rate of dissipative energy loss due to drag, and this rate of loss exceeded the rate of replenishment by the slope lift, resulting in a net decrease in total energy with time.

The dynamic soaring phase of the flight exhibits very different behavior (Figure 20). The period of the cycles was about the same, but the airspeed remained high (> 80 feet/sec) throughout the series, so the dissipation rate was much greater. Nevertheless, there was a net increase in total energy due to the two shear crossings for every cycle (the velocity increase during the shear crossings averaged around 20 feet/sec, consistent with the previous estimate of ΔU). Assuming that the K value for this cycle is about 0.2, the L/D ratio for this plane can be estimated from equation (23) to be about 22.

V. EVOLUTIONARY BEHAVIORAL ENGINEERING

This part of the project extends the work of Pryor [1998] and Barnette *et al*. [2000] who applied evolutionary methods to the development of robotic behavior. Pryor [1998] originally developed a genetic programming model to solve a suite of high-level robotics problems. The motion of Pryor's "robugs" was idealized and highly constrained so that the focus of his evolutionary model could be on high-level navigational behaviors and goals. Low-level maneuvering issue--such as locomotion, steering, and braking control--were not initially addressed. Instead, the simulated robugs were constrained to move on a discrete two-dimensional square grid with instructions such as "turn" or "move ahead".

Behaviors were automatically generated using a genetic program to solve a series of problems in which robots are randomly distributed on a grid with obstacles and rewarded for finding a source that is emitting a signal. The required high-level navigational behavior can be encapsulated as a computer program, which can be engineered by a variety of methods. In the simplest case—when effective rules are easy to conceive and implement—the behavior can be coded by hand. For trivial goals, common sense is all that is required for inventing rules. More difficult challenges require more sophisticated solutions, which can be generated by a variety of optimization methods including thermodynamic analogy models, conventional guidance theory, or reinforcement learning.

It is important to recognize that—whatever method is employed—behavioral engineering is not a true optimization problem. The types of problems that are likely to be of most use are analytically intractable and involve behavior that operates in a noisy, changing, real-world environment. Any optimization method that is applied to designing for such an environment is going to yield a solution that depends on the simplifying assumptions, and on the figure of merit, chosen by the behavioral engineer. The global optimum may strongly depend on seemingly subtle differences in problem representation. It is therefore important to focus on behaviors that are "good enough" and avoid thinking in terms of "the best," because the best solution

determined by some quantitative test will always be an artifact of the problem definition. Because representation is the most important part of the problem, and will always be based on the discretion of the researcher, behavioral engineering is as much of an art as it is a science.

A. Genetic Programming

Pryor [1998], and Barnette *et al*. [2000] outline the details of the implementation, and the specifics are not repeated here. The representation allows a great deal of flexibility, and can be adapted to many types of problems. The behavior programs execute by traversing a tree that is made up of building blocks called nodes, which can either be a function or a terminal. Functions perform operations and contain pointers to other nodes. Terminals return values that result in an instruction to the robot. The trees themselves are generated by a genetic programming model originally developed by Pryor [1998] and based on methods described by Koza [1992], within a general framework presented by Holland [1975]. Genetic programming is a type of genetic algorithm, an evolutionary computing method that is based on the principles of biological evolution.

Evolution takes place over many discrete steps called generations. Generations in nature are not synchronous because lifetimes and breeding times vary in length, but evolution has been in operation for billions of years on Earth. In the model they are synchronized for simplicity, and the number of generations is limited by practical considerations to hundreds. Each generation consists of a population of individuals. In nature these are organisms and the population size can vary and can reach numbers of millions or billions. In the model they are computer programs and the populations are held fixed for a given problem, with typical sizes on the order of thousands. The Darwinian principle of "survival of the fittest" is applied. In nature, any individual that survives long enough to breed and generate offspring is fit by definition. In the model, each individual behavior program is assigned a numeric score based on a "fitness function" chosen by the researcher to represent the problem, and survival depends on rank. Finally, the individuals must reproduce. In nature, the genotype is carried by DNA, which is mixed between individuals by sexual reproduction, and random processes occasionally introduce mutations. The human genome contains between 30,000 and 100,000 genes representing many millions of base pairs. In the model, the "genotypes" of two individuals are mixed with a crossover operator, and random mutations are introduced to allow exploration of other regions of the "fitness landscape" (see Koza [1992] for more a more detailed descriptions of these concepts). These artificial genomes have much lower information content than DNA in nature, containing hundreds of genes represented by thousands of bits of information.

It is clear that artificial evolution does not come close to the fidelity of evolution in the natural world in terms of numbers of generations, numbers of individuals, or information content of the genome. However, the largest obstacle seems to be the fact that evolutionary computing methods are simulations, whereas natural evolution operates in the real world. Unless the "off-line" simulation environment captures the important characteristics of the real world, successful individuals (behaviors) may not survive in reality. In biology, on the other hand, the fitness of individuals is tested "on-line" under actual survival conditions.

This project makes use of the most pragmatic approach and rejects the purist strategy that "evolution should be allowed to operate on its own and eventually it will find the best solution". For this application, the genetic programming methods are intended to extend (but not replace) the creativity and intelligence of the designer. Unlike the natural world, we do not have billions

of years, the high information density of DNA, or large populations that can generate behaviors from scratch such as (for example) the dynamic soaring flight of the albatross.

B. 2D Grid-based albatross simulations

Pryor [2002] defined the grid-based UAV problem that became a benchmark for this project. The problem is similar to the robug problem in that the UAV is constrained to move on a two dimensional Cartesian grid. The aircraft requires flight behavior logic to search for uplift regions and investigate the surrounding area without crashing. This problem is more complicated than the robug problem because of the different procedures that must be done sequentially. The agent must search and map the uplift region before the surrounding area can be explored, and conflicting goals of exploration versus exploitation (of the lift zone) must be balanced. This conflict was too difficult for a conventional genetic program, and more advanced methods had to be added.

The benchmark problem is illustrated in Figure 22. A rectangular "lift zone" is generated with its short dimension fixed at 20 units and its long dimension randomly chosen from a range of between 85 and 105 units. The long-axis is randomly oriented along the *x*, *-x*, *y*, or *-y* axis of the Cartesian coordinate system. The rectangle is placed such that the origin is 5 units from one

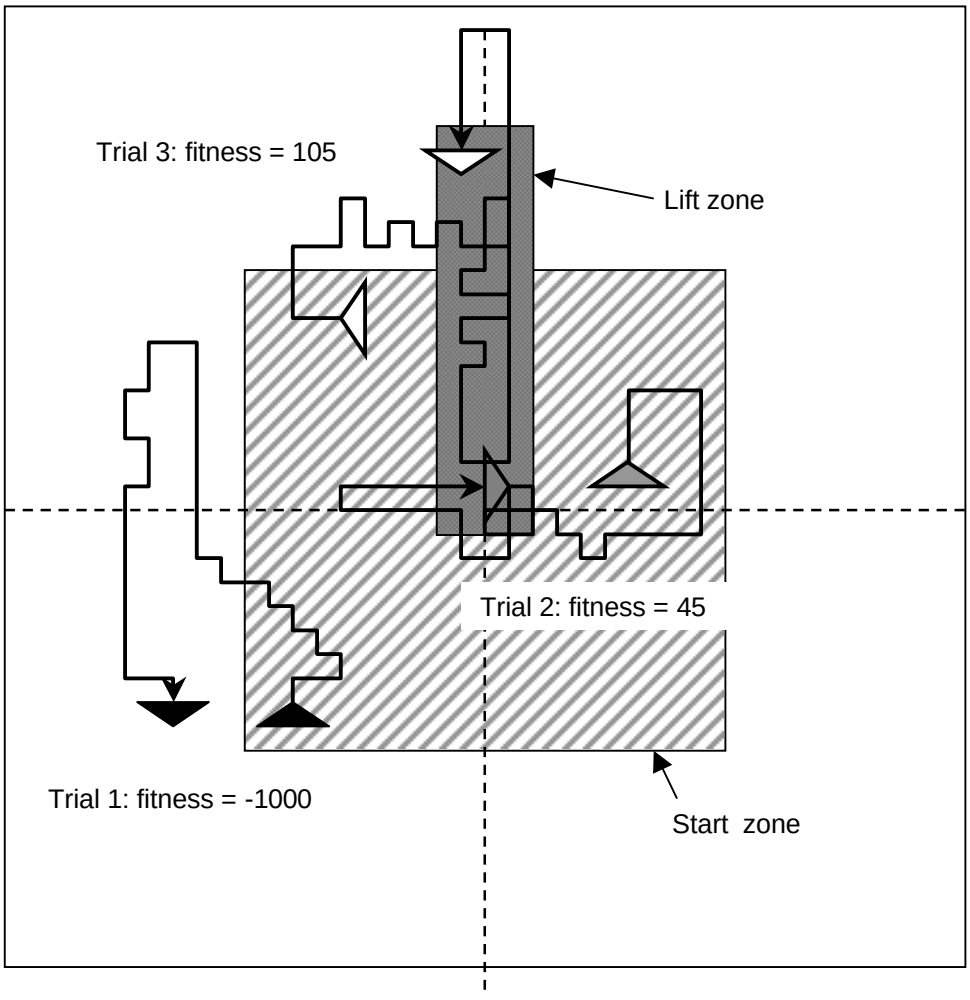

Figure 22. Benchmark problem for developing GP methods.

end, and centered on its width. This leaves between 80 and 100 units extending on one of the four directions. A bird is placed with its orientation chosen randomly from one of the four directions at a random location within a 100×100 square box centered on the origin, at an altitude of $|x| + |y| + 20$ units. When it is outside the lift zone, it loses on unit of elevation for every time step. Inside the lift zone it gains one unit, up to a maximum of 250. It moves one positive or negative unit along either the *x* or *y*-axis, every time step. It can go straight, turn left, or turn right (a recent modification to the benchmark allows for U-turns). The fitness is defined by the maximum distance away from the origin that a bird reaches before returning to the lift zone. The actual fitness is associated with the evolved behavior program, not with a given instantiation of the bird, so the fitnesses of individual birds running the same program--but with a range of initializations—are averaged. If its altitude goes to zero, a bird crashes and dies, and its contribution to the fitness function is assigned a large negative value.

This would appear to be an extremely simple problem compared to actual flight, but it involves staged and somewhat contradictory goals that force a balance between exploration and exploitation. The conservative bird will stay in the lift zone to preserve its life, but will get a low score. The bold bird will fly away from the lift zone and increase its risk of crashing. The highly ranked bird will require a delicately tuned algorithm. The benchmark also involves a set of goals that must be accomplished in sequence: 1) the bird must find the lift zone, 2) the bird must exploit the lift zone, 3) the bird must explore away from the lift zone, and 4) the bird must return the lift zone.

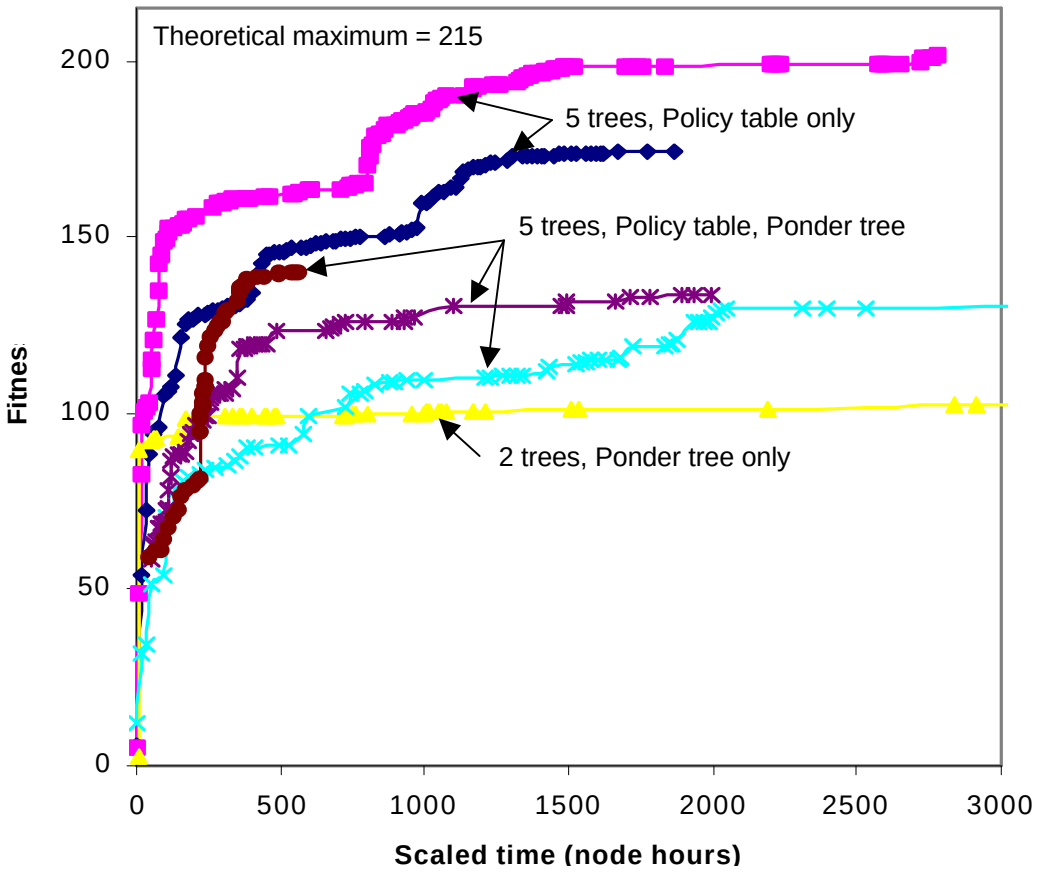

Figure 23. Benchmark GP problem performance on CPlant.

C. YGP generalized genetic program

Many iterations of the genetic programming tool were implemented by Pryor [2002] to tackle this problem, with varying degrees of success. Versions of the code were written with single trees, multiple trees, and various trees executed in sequence. This work formed the basis of the evolutionary model employed for this project The details of Pryor's work will be the subject of a future report.

YGP was developed for this project as a generalized version of Pryor's GP, with the added goals of 1) platform independence, 2) modularity, and 3) integrated visualization tools. In YGP, the data structure of the tree that contains the flight control algorithm consists of multiple arrays that keep track of state variables (for flight simulation), calculated variables (for high-level situational awareness and decisions), calculated integers (for policy table flags), registers (genetically calculated values) and pointers (for multiple decision trees that depend on policy flags). Array dimensions are determined at compile time, and depend on the evaluation function, the number of degrees of freedom of the flight vehicle, and whether the flight space is discrete or continuous. By generalizing the genetic programming, behaviors for any environmental rules can be developed with the same code, from the most basic (4 degrees of freedom, discrete space, simple flight rules, fixed lift zone) to the most advanced (6 degrees of freedom, continuous space, full aerodynamics, turbulent boundary layer). We used the most basic example as a benchmark because it runs much faster than a simulation that must solve realistic equations of motion at every time step.

The only version of YGP that demonstrated fast convergence and approached the theoretical best solution for the benchmark problem is the one that implemented a policy table. This makes sense because of the sequential and competing goals described earlier. It is because a bird in one situation, such as needing to find the lift zone, needs to be running a very different program than one in a situation where long-distance exploration is paramount. For that reason, several trees were evolved independently, and each were called based on the a policy table value that represented its current situation, such as 1) the bird has not discovered the lift zone, 2) the bird is inside the lift zone, 3) the bird has reached and altitude of 250, or 4) time is running out. When such a policy table was implemented, the ability of the code to find good solutions increased

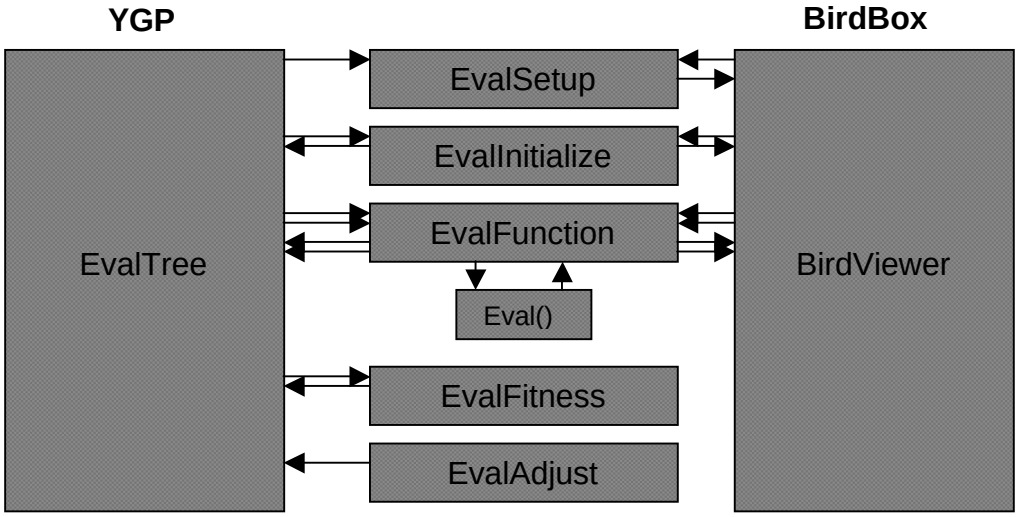

Figure 24. Block diagram for YGP integrated with BirdBox.

markedly (Figure 23).

Platform independence of YGP was achieved by implementing a text-based description of the program trees in place of the original binary representation. These text trees are compact, with about a 10 Kbytes storage requirement per behavior. The code was reorganized so that the evaluation routines were broken into modules that could be called either from YGP or another (e.g. visualization) program (Figure 24). By keeping the modules independent, the code could be applied to other problems simply by inserting a different evaluation function.

D. BirdBox visualization tool

Finally, a cross-platform visualization tool called BirdBox (Figure 25) was written using OpenGL, GLUT, and GLUI libraries. BirdBox allows a researcher to visually observe simulated gliders exhibiting flight behaviors that have been evolved with YGP. The user can modify the camera rotation, camera translation, number of birds, size of birds, texture mapping, water animation, and screen update rate. GUI buttons provide the underlying functionality by allowing the user to load a new behavior file, and switch to other environments. A data window displays time and instantaneous information about the bird state, such as distance, altitude, and current register. Birds are rendered to appear as animated paper airplanes, each with its own color and shadow on the ground to locate the bird in the *x*-*y* plane. The same evaluation function subroutines that are used in YGP are used in BirdBox, with the same data structure and traversal function for the decision trees that are evolved with YGP. BirdBox was a useful tool to assist in developing YGP for the simple benchmark test, but for the more sophisticated flight analysis its function was replaced by FlightGear, and more recently by Umbra.

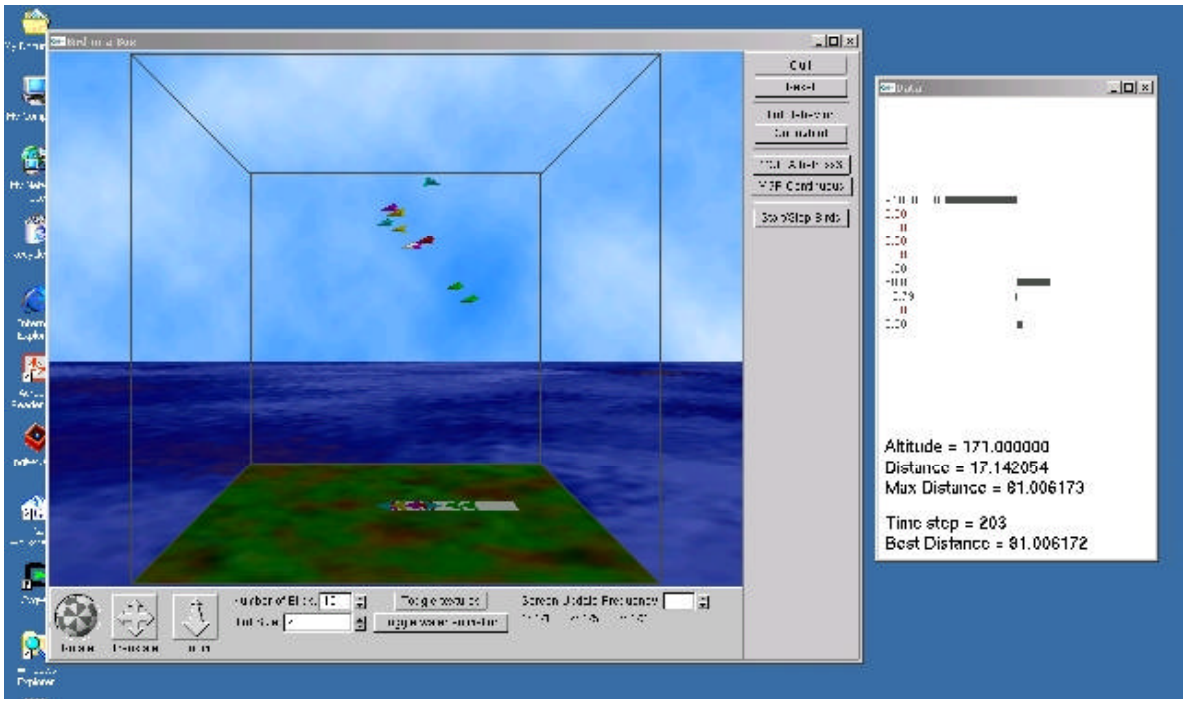

Figure 25. Screen capture of BirdBox visualization tool.

VI. APPLICATIONS AND FUTURE WORK

Since the time of Lord Rayleigh, researchers have speculated on the possibility of imitating the soaring behavior of albatrosses to extract energy from the wind to power human flight. However, the more that was understood about the aerodynamics of flight, the more it was realized that the practical aspects of the problem were insurmountable with 20th-century technology. Fortunately, in the final years of the century, advances on several fronts converged to make this dream possible if the definition of human flight is extended to include flight of all human-designed aircraft. Technology has now reached a state where autonomous dynamic soaring UAVs, and perhaps swarms of these vehicles, are within reach. These advances include inexpensive consumer electronics for radio-controlled flight, durable foams for the construction of low-cost small aircraft, powerful but cheap programmable palm computers to provide sufficient intelligence for control, lightweight GPS receivers for navigation, small solid-state sensors for avionics, and wireless internet communication for exchanging information between aircraft. Simultaneously, the world of radio-controlled recreational flight has just discovered and learned to exploit dynamic soaring for entertainment and competition, providing a new source of experience and information about the subject.

The ultimate goal of this research is to develop UAVs that extract their locomotion energy from the local environment, have a high degree of free mobility and endurance, adaptively modify their flight cycle to optimize energy harvesting, and have simple behavioral rules that can be used to build collective intelligence. This project has made progress toward these goals by 1) developing an understanding of dynamic soaring by numerically modeling and optimizing simple cycles, 2) developing a variety of software tools that can be used to analyze and visualize dynamic soaring, 3) developing software tools that can automatically generate flight behaviors that can be programmed into autonomous UAVs, and 4) designing and building prototype UAVs for collecting and analyzing the first data from dynamic soaring flight in laminar shear. Such a small autonomous glider could cover large distances with very little energy consumption. This device could have practical applications for aiding search missions, reconnaissance, or measurements of sea-surface or meteorological properties (*e.g.* surface and air temperature, barometric pressure, wind speed and direction, wave height, or even oil-slick characterization).

These tools can also be used to assess other strategies for wind-powered or wind-assisted flight. Extraction of energy from gusts is theoretically possible; it is well known that the flight performance of passive “dumb” gliders decreases in turbulent conditions, whereas vultures and other soaring birds exhibit better performance. One key to extracting energy from the motion of the air may be the invocation of collective distributed intelligence of a swarm of aircraft that communicate their state and local conditions to one another. Even under atmospheric conditions that prohibit the extraction of 100% of flight energy from the wind, it may be possible to radically increase the efficiency of powered flight by a judicious choice of flight behavior that can be discovered using the tools developed in this project.

VII. REFERENCES

Barnette, D.W., Pryor, R.J., and Feddema, J.T. (2000), "Development and application of genetic algorithms for Sandia's RATLER robotic vehicles," SAND2000—2846, Sandia National Laboratories, Albuquerque, New Mexico, November 2000.

Capper, D.M. (1994), *Introducing C++ for scientists, engineers, and mathematicians*. Springer-Verlag, London.

Cone, C.D., Jr. (1964), "A mathematical analysis of the dynamic soaring flight of the albatross with ecological interpretations," Virginia Institute of Marine Science Special Scientific Report. No. 50, 104 pp.

Forrest, S. (1993), "Genetic algorithms: principles of natural selection applied to computation," *Science,* **261**. 872-878.

Gottlieb, E., Harrigan, R., McDonald, M., Oppel, F., and Xavier, P., (2001), "The Umbra simulation framework," Sandia Report SAND2001-1533, Sandia National Laboratories.

Hargrave, L. (1899), "Sailing birds are dependent on wave power," *Journal of the Royal Society of New South Wales,* 125-128..

Hendricks, F. (1972), *Dynamic Soaring*. Ph.D. Thesis, University of California, Los Angeles.

Holland, J.H. (1975), *Adaptation in natural and artificial systems.* Univ. of Michigan Press, Ann Arbor.

Idrac, P., (1931) "Études Expérimentales sur le Vol à Voile," *Vivien*, Paris.

Jackson, E.B. (1995). "Manual for a workstation-based generic flight simulation probram (LaRCsim) version 1.4," NASA Technical Memorandum 110164.

Kolmogorov, A.N. (1941), "Dissipation of energy in a locally isotropic turbulence," *Doklady Akad. Nauk SSSR*, **32**, 141.

Koza, J.R.(1992), *Genetic Programming*. The MIT Press.

Langley, S.P. (1893), "The Internal Work of the Wind," *Smithsonian Contributions to Knowledge*, No. 884.

McFarland, Richard E., (1975), "A standard kinematic model for flight simulation at NASA-Ames," NASA CR-2497.

Pennycuick, C.J. (1982), "The flight of petrels and albatrosses (procellariiformes), observed in South Georgia and its vicinity," *Phil Trans. R. Soc. Lond.*, **B300**, 75-106.

Pryor, R.J. (1998), "Developing robotic behavior using genetic a genetic programming model," SAND98—0074, Sandia National Laboratories, Albuquerque, New Mexico, January 1998.

Pryor, R.J. (2001), report in preparation.

Sehgal, B., Deters, R.W, and Selig, M.S., (2002), "Icing Encounter Flight Simulator," AIAA 2002-0817, 40th AIAA Aerospace Sciences Meeting and Exibit, Reno, Nevada.

Sim, A.G. (1990), “Flight Characteristics of a Modified Schweizer SGS 1-36 Sailplane at Low and Very High Angles of Attack,” *NASA Technical Paper 3022*, **B300**, 44 pp.

Stevens, B.L. and Lewis, F.L. (1992), *Aircraft Control and Simulation*. John Wiley & Sons, New York, 617 pp.

Lord Rayleigh (1883), “The soaring of birds,” *Nature*, **27**, 534-535.

Spear, LB, and Ainley, D.G. (1997), “Flight speed of seabirds in relation to wind-speed and direction,” *Ibis.*, **139**, 234-251.

Walkden, S.L. (1925), “Experimental studies of the soaring of albatrosses,” *Nature*, **116**, 132-134.

Weimerskirch, H. and Robertson, G. (1994), “Satellite tracking of light-mantled sooty albatrosses,” *Polar Biol*, **14**, 123-126.

Weimerskirch, H., Bonadonna, F., Bailleul, F., Mabille, G., Dell’Omo, G., and Lipp, H-P., (2002), “GPS tracking of foraging albatrosses,” *Science*, **295**, 1259.

Wilson, J.A. (1975), “Sweeping flight and soaring by albatrosses,” *Nature*, **257**, 307-308.

Wood, C.J. (1973), “The flight of albatrosses (a computer simulation),” *Ibis*, **115**, 244-256.

Appendix A
Analytical solution for horizontal turns

For level flight in a horizontal turn, the lift is equal to wing loading per unit mass, normalized to gravitational acceleration, and the drag is equal to the (negative) change in velocity:

$$l = (c^2+1)^{1/2}, \qquad d = -(dV/dt)/g \tag{A1}$$

where c is the centripetal acceleration divided by g. Equations (A1) and (5) become:

$$dV/dt = -gaV^2 - gb(c+1)/V^2 \tag{A2}$$

If the bird enters the turn with velocity V_0, the time interval for the velocity to drop to V_1 is:

$$\Delta t = -\frac{1}{g}\int_{V_O}^{V_1}\frac{dV}{V^2+b(c+1)^2V^2} = -\frac{1}{ga}\int_{V_O}^{V_1}\frac{V^2dV}{V^4+k^4} \tag{A3}$$

where $k^4=b(c+1)/a$. The definite integral can be solved exactly to yield:

$$\Delta t = -\frac{1}{2\sqrt{2}gak}\left[\frac{1}{2}\log\frac{V^2-\sqrt{2}kV+k^2}{V^2+\sqrt{2}kV+k^2}+\tan^{-1}\frac{\sqrt{2}kV}{k^2-V^2}\right]_{V_0}^{V_1} \tag{A4}$$

The initial velocity V_0 is given, but V_1 is unknown independently of Δt. We need a second integral that relates the velocity to angle, and solve for a 180° turn. We use the definition of centripetal acceleration, $cg = Vd\theta/dt$, to determine the differential change in direction angle (θ):

$$\Delta\theta = \frac{cg}{V}dt = \frac{cg}{V}\frac{dt}{dV}dV \tag{A5}$$

Substituting the drag-induced negative acceleration, with constant c, equation (A2), yields:

$$\Delta\theta = -c\int_{V_O}^{V_1}\frac{V\ dV}{aV^4+b(c+1)^2} = -\frac{c}{a}\int_{V_O}^{V_1}\frac{V\ dV}{V^4+k^4} \tag{A6}$$

This can be solved analytically for V_1 in terms of V_0 and $\Delta\theta$:

$$V_1 = k\sqrt{\tan\left(\tan^{-1}\frac{V_0^2}{k^2}-\frac{2ak\Delta\theta}{c}\right)} \tag{A7}$$

By substituting (A7) back into (A4), the time required for the maneuver is determined. The analytical values for two turns in Rayleigh's cycle are compared to the numerical solutions in Table 2.

Appendix B
Genetic Algorithm

The genetic algorithm code used in the two-dimensional piecewise optimization (Section II) is written in C++ and uses a modified bit-string class of Capper [1994] so that it has functions for mutate, crossover, and various fitness evaluations. An "individual" class carries the phenotype, fitness value, parents, and last crossover point. The bit-string has functionality, such as Gray-Encode and Gray-Decode (to convert integers directly from and to Gray). Binary-Gray and Gray-Binary conversion functions were added, mainly for testing purposes. For the optimization, the real number range is discretized into the appropriate number of intervals with sizes that depend on the desired numerical resolution. This gives the bit- string individual enough flexibility to handle a real number optimization, without the overkill one might get if all reals had to be represented by type double, for example.

The individual class contains the set of variables "phenotype" which can be thought of as a vector consisting of the four parameters to be optimized. This phenotype is output for every individual of every generation along with fitness determined by the fitness function. Fitness functions can be customized to handle the various flight cycles (such as the Cuban 8 cycle).

Representation

Phenotypes can be determined by converting Gray-coded bit-string fields to real numbers. The following functions return the various phenotypes:

```
   double bit_array::Pheno_Function0(void) {  //v0
   return 5*Gray_Decode(0,8)/255 + 18;  // range 18 to 23
   }
   double bit_array::Pheno_Function1(void) {  //v1/v0
     return  0.3*Gray_Decode(8,8)/255 + 0.5;  // range 0.5 to 0.8
(fraction of V0)
   }
   double bit_array::Pheno_Function2(void) {  //alpha
     return 10*Gray_Decode(16,8)/255 + 5;  // range 5 to 15 (degrees)
   }
   double bit_array::Pheno_Function3(void) {  //beta(60-alpha)
     return  Gray_Decode(24,8)/255;  //range 0 to 1
   double bit_array::Gray_Decode(void) {  //decode Gray representation
     double accum=0;
     int previous=0, current=0, powerof2;
     powerof2 = pow(2,max_bits-1);
     for (int i=0; i<max_bits; i++) {
       current = (previous+operator[](i))%2;
       accum += powerof2*current;
       previous = current;
       powerof2 /= 2;
```

```
  }
  return accum;
}
```

Selection

The GA was run with scaled Roulette Wheel selection, using elitism with an odd-numbered population, and with a copy of the elite individual in the odd position that never crosses over.

Crossover

Crossing over within the 8-bit real fields is not indicated by the building blocks hypothesis, so a crossover function was implemented so that the fields would be treated as the building blocks:

```
jcross = Uniform(1,max_bits-1);
  // Original version of single point crossover
jcross = Uniform(1,3);
  // This is the version is used for 4 8-bit reals
jcross = jcross*8;
```

Mutation

Likewise, mutation should be treated differently when the bit strings are representing real values, so a mutation operator preferentially mutates the rightmost bits of each field, corresponding to smaller changes in the real value. Here are the corresponding functions:

```
for (int i=0; i<max_bits; i++) {
    // Original for uniform mutation
    double d = drand48();
    if (d <=  pmutation) {
      nmutation++;
      bitflip(i);
    }
for (int j=0; j<4; j++) {
    for (int i=0; i<8; i++) {
      double d = drand48()*2*(8-i)/8;  //backwards
      if (d <=  pmutation) {
        nmutation++;
        bitflip(i+j*8);
        cout<<"mutating i = "<<i<<" and j = "<<j<<endl;
      }
    }
 }
```

Appendix C
Flight log for test flight #3, Nov. 16, 1999.

00:00	Logger on.
00:10	Hold in launch position to record wind speed.
00:20	Launch, followed by steady climb to record total energy gain rate on windward side.
01:15	Series of dives: steep dive to north; dive to south; "mild dive" to north; flip switch; loop to fly through gradient the wrong way.
3:00	Low altitude passes; Joe trying to keep turns to within 5 feet in altitude.
5:00	Still on front side, back and forth slope soaring. visually bisecting airframe with horizon.
6:15	Downwind dive to leeward; Clockwise (plan view) cycles, constant altitude 4th one very good; Then went to inclined circles with consistent delta altitude.
6:53	Decreased altitude.
7:50	Went from circle to oblong race track; so consistent crossings of high shear layer.
8:20	Went front to reverse to CCW cycles, a little more delta altitude.
9:30	Some shear buffeting, flying almost parallel to shear.
9:50	Back to CW plan view cycles, now flying classic DS cycles tried to time some cycles: 8.5 sec, 7.9 sec.
10:40	8.4 second cycle period.
10:50	6.2 second cycle period.
11:06	Switched to oblique shear crossing, had to grab some slope lift; some big cycles to get more than a few GPS pt/cycle.
12:10	Higher crossings.
12:40	Big high cycles, a big roll-yaw wiggle in shear.
13:50	Mixing it up, and will have to sort out from data.
14:15	Flipped switch and went up front.
14:30	Dive parallel to hill, then climb parallel to hill.
15:00	Steeper dive, steep upward crossing almost perpendicular to shear interface.
15:30	~75 degree crossing angle…tail slide?
16:10	Vertical crossing down and up; top of plane (+z) was pointed down wind; no energy gain.
17:00	Circling back.
17:30	Dive with top (+z) upwind.
18:00	Crossing toward south perpendicular to wind direction no energy addition.
18:30	“Gassing up” on front side, flying south.
19:15	Same going north.
20:00	Headed south through boundary; N. through boundary. no energy gain, stalled.
20:40	DS to get energy.
21:40	Cross below and then above interface going south. Big yaw like a weathervane when went through shear.
22:11	Set down. Low battery light blinking.

DISTRIBUTION

MS 0310 A. Backer, 9212
MS 0318 M.B.E. Boslough, 9212 (20)
MS 0318 K. Boyack, 9212
MS 0196 C. Churchwell, 9212
MS 0318 G. S. Davidson, 9212
MS 9951 J-L. Faulon, 9212
MS 0316 L. Frink, 9212
MS 0196 S. Martin, 9212
MS 0196 E. May, 9212
MS 1109 R. Pryor, 9212
MS 0316 S. Rempe, 9212
MS 0316 M.D. Rintoul, 9212
MS 0321 W. Camp, 9200
MS 0612 Review and Approval Desk, 9612
MS 0899 Technical Library, 9616 (2)
MS 9018 Central Technical Files, 8945-1