THE TRANS-PACIFIC CROSSING: LONG RANGE ADAPTIVE PATH PLANNING FOR UAVS THROUGH VARIABLE WIND FIELDS

Juan Carlos Rubio[*], *University of Washington, Seattle, WA*
Sean Kragelund[†], *Naval Undersea Warfare Center, Keyport, WA*

Abstract

The application of an evolution-based path planner for a UAV in long range flight through realistic wind fields is presented. The planner makes use of wind information obtained from actual weather databases. Horizontal wind variability in both 3-D space and time is taken into account for the planning of the complete path. Two additional features in the planner include a spherical Earth geometry for path generation and a UAV performance and fuel consumption model. The objective of the planner is to reduce fuel requirements for the entire flight. The planner is capable of making in-flight modifications to further refine or adapt its path to updated wind information. Simulation results for a flight from Honolulu, HI to Long Beach, WA with an Aerosonde aircraft are presented.

Introduction

In August of 1998 the University of Washington and The Insitu Group achieved the first crossing of the Atlantic Ocean by a miniature robotic aircraft, the Aerosonde Laima [1]. It was launched from Newfoundland, Canada, and arrived in the Scottish Hebrides after flying a distance of 3270 km in less than 27 hours and consuming about 5.7 liters of gasoline. The flight route was preprogrammed at launch based on current weather conditions and forecast information, but once en-route, changes to the flight plan could no longer be made. Thus, luck played an important part in this successful flight.

For proactive success of future long range flights of small UAVs, en-route planning will be required. Even though forecast models have improved, weather and wind patterns continuously change and are still difficult to predict. Thus, the use of frequently updated weather information during the flight will allow adaptation and re-planning based on current and near-term forecasts.

This paper presents simulation results of an adaptive path planning scheme for the crossing of the Pacific Ocean from Honolulu, HI to Long Beach, WA with an Aerosonde UAV. The purpose of these simulations is to present a direct application of Evolutionary Algorithm (EA) techniques on nearly realistic scenarios. Actual weather data has been extracted from historical reanalysis archives for use in the planner. A spherical Earth model has been implemented for this long-range flight simulation. The airplane characteristics have also been taken into consideration through simplified tables of flight performance and fuel consumption characteristics.

Algorithms and Implementation

EA based techniques have shown promise for solving optimization problems where complex and variable environmental characteristics are an important factor [2], [3]. They are able to find near optimal solutions while also demonstrating adaptability to find solutions when situations change. In general, these algorithms operate by maximizing (minimizing) a fitness (cost) function based on the characteristics of the problem being solved. The path planner used for the simulations presented here, which is under development at the University of Washington, is based on these techniques [4].

The Path Planner

The basic idea behind the planner is to generate a set of paths (*population*) that are feasible solutions to the problem. After simulating a natural

---

[*]Research Assistant, Department of Aeronautics and Astronautics
[†]Electronics Engineer, National UUV Test & Evaluation Center

selection process, those solutions that 'survive' among their generation are retained. They (now *parents*) are then combined/modified to generate new solutions (*offspring*). These solutions now constitute a new population (next *generation*). The process is repeated (*evolution*) until a solution satisfies certain criteria. This is the core of the path planner, referred to as the *evolver*.

For the problem of this paper, a solution is a path. A path is defined as a route for the vehicle to follow. Once a path (or a part of one) has been selected for use by the vehicle, it is called a trajectory and no further evolution is performed on that part. Each path is constructed of segments. Each segment is discretized into subsegments for computation purposes, but each segment is maintained as a single entity (Figure 1). The evolution of a path consists of the modification of one or more of its segments to generate a new path.

The path planner consists of two general processes: initialization and evolution. Initially a population must be generated for it to evolve. A specified number of paths is generated from the starting point by connecting randomly created segments one after the other. The evolution process is divided into three main steps: fitness evaluation, selection, and mutation.

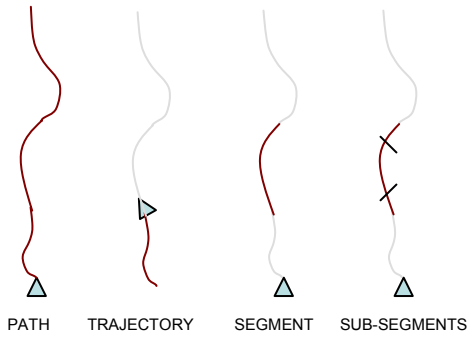

Figure 1. Basic Path Planner Route Concepts

Fitness evaluation. The performance of a path is measured using a cost function; the lesser the value, the better the 'fitness' of the solution. The function used in this path planner algorithm is a linear sum of parameters of interest. Different scaling factors are used to balance the importance of each parameter. Fitness evaluation is a subject of continuing research in the development of these adaptive path planners. However, for the purposes of this paper only three parameters were taken into consideration, and this very simple type of fitness function yielded good results.

The main optimization objective in these simulations is to find paths that minimize fuel consumption during a simulated flight. The fitness function also includes a term related to the distance between the endpoint of the path and the destination point. This second characteristic in the fitness function is weighted more heavily to ensure that the planner ends its paths where requested. At this time, no intermediate targets are inserted for the planner to consider. A third parameter, weighted less heavily, is the overall length of the path segments. This is to discourage very long segments compared to the average.

Selection. This process compares each path in the population with q=population/2 randomly selected competitor paths. Points are assigned based on the results of pairwise comparisons, and selection is based on the number of points each path achieves. Also, the path with the best fitness value is always retained regardless of its competition results. The number of paths retained is predetermined before running the planner; all other paths are eliminated.

Mutation. Once the parent paths have been selected through the previous step, offspring are generated. Five randomly selected methods of mutation are used. The *crossover* method randomly takes some initial segments of one path and some ending segments of a second, and joins them with some necessary number of segments. The *1-point mutation* randomly selects a point in the path, keeps only the segments up to that point and re-propagates the path with new segments. The *2-point mutation* selects two points in the path, eliminates the segments in between, re-propagates from point 1 on and then joins the end of the propagation with point 2. The *shrink* and *expand* methods simply eliminate or add segments at the end of the path.

An additional feature implements an application of the common phrase 'two heads are better than one'. Instead of a single evolver, two independent evolvers perform the same vehicle path search. At scheduled intervals, just before the planner selects the next trajectory to traverse, the best path results of both evolvers are compared. The evolver whose population contains the overall best

path (based on fitness value) is selected to continue evolving the future path segments (the normal process for a single evolver). The second evolver is 'reset' and begins replanning from scratch for the remaining distance, with a handicap of a specified number of generations. This process provides more flexibility for reacting to new situations by having a 'fresh' mind available with a pool of different potential solutions. Currently no information sharing occurs between the evolvers which could eventually result in converging more rapidly to better solutions.

Spherical Geometry and Assumptions

For this long-range application, a flat Earth assumption is not sufficient. Instead, a spherical model has been implemented in the path planner. This approach takes several factors into account such as the distance of one degree of longitude varying with latitude; the shortest distance between two points in a sphere following a great circle; heading azimuth (the angle relative to north) varying while moving straight from one point to another; etc.

Two main types of geometries are used for segments: straight segments follow a great circle path, which is a 'projected' great circle path when changing altitude, and may include linear speed changes; curve segments follow a small circle path at constant altitude and speed (Figure 2).

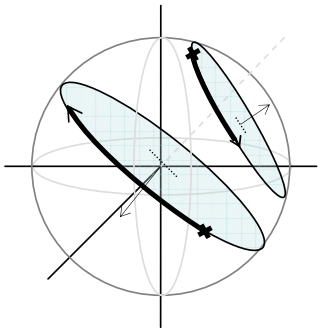

Figure 2. Great Circle and Small Circle Paths

Key parameters used to determine each segment are shape type (which defines specific parameters for each, i.e. radius of turn for a curve segment), initial and final state parameters (position, heading, speed), and length. The latter is either calculated based on initial and final states, or it is specified and used to determine the end of the segment based on an initial state.

Throughout the path planner, concepts such as longitude, latitude, altitude, heading azimuth and heading elevation (angle of path relative to the 'horizontal' plane) are used for segment construction. Required calculations for the spherical Earth model were developed using 3-D Cartesian vectors based on those five parameters. The original coordinate system is a right handed frame X-Y-Z with its origin at the Earth's center, the x axis passing through lat 0^o, long 0^o, and the z axis passing through the North Pole.

Two starting vectors are required for others to be derived through common vector operations (Figure 3). The first is the position vector $\underline{r}$, which is obtained through a well known relationship with latitude and longitude angles. The second is the azimuth vector $\underline{a}$. Its relationship to the latitude (θ), longitude (ϕ), and azimuth (ψ) angles is given by:

$$a_z = \cos\theta * \cos\psi$$

$$a_y = -\sin\phi * \cos\psi * \sin\theta + \cos\phi * \sin\psi$$

$$a_x = -\cos\phi * \cos\psi * \sin\theta - \sin\phi * \sin\psi$$

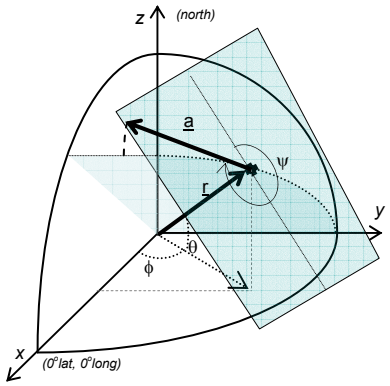

Figure 3. Geometry for Azimuth Vector Computation.

These two unit vectors $\underline{r}$ and $\underline{a}$ define the normal vector $\underline{n}$ to the great circle plane via the cross product $\underline{r}$ x $\underline{a}$. These three vectors form a right handed coordinate system R-A-N and provide a good basis for segment calculations.

Straight Segments (Climbs and Lines)

The general case is the geometry used for climbs (lines being the case when the elevation angle $\varepsilon = 0$). Figure 4 presents the geometrical

structure/method for these segments. It is a method to obtain the final state after traveling a distance S from the initial state. The assumption is that the path follows a circular type climb with the same radius as the great circle that would be flown with no altitude change, this given by the radius of Earth + initial altitude (R.E.A.). The final position vector $\underline{p}$, is obtained by

$\underline{p} = \underline{m} + \underline{f}$, where

$\underline{m} = \underline{cr} + \underline{r}$,

$\underline{f} = [\cos\beta, \sin\beta, 0]\,[\underline{cr}, \underline{e}, \underline{n}]^T$, and

$\underline{cr} = [-\cos\varepsilon, \sin\varepsilon, 0]\,[\underline{r}, \underline{a}, \underline{n}]^T + \underline{r}$

$\underline{e} = \underline{cr} \times \underline{n}$,

β = S / R.E.A., S being the desired flying distance,

$[\underline{cr}, \underline{e}, \underline{n}]^T =$

$[[cr_r, cr_a, cr_n],[e_r, e_a, e_n],[n_r, n_a, n_n]]\,[\underline{r}, \underline{a}, \underline{n}]^T$

$[\underline{r}, \underline{a}, \underline{n}]^T =$

$[[r_x, r_y, r_z],[a_x, a_y, a_z],[n_x, n_y, n_z]]\,[\underline{x}, \underline{y}, \underline{z}]^T$

Every vector above has been normalized by R.E.A.

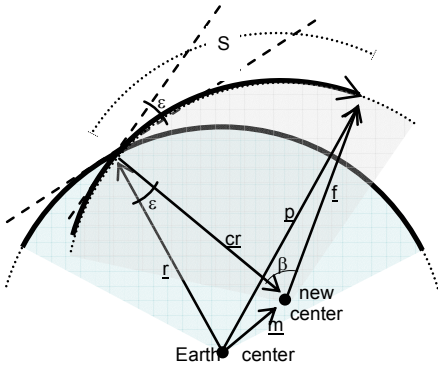

Figure 4. Great Circle Projection Method for Climb Segments (Normal View of Great Circle Plane)

The final elevation angle is given by the angle between vectors $\underline{f}$ and $\underline{p}$. The final azimuth angle (which is not the same as the azimuth of the initial state) is obtained by computing the azimuth angle that points from the end position to the initial position (i.e., look 'backwards' to where you started) through a method referred to as *getDirectionTo*, explained later in this section. This angle $+180^o$ gives the final state azimuth.

This circular geometry used for climbs restricts S not to exceed S_{max} = R.E.A.*(π + ε)/2 in order to avoid 'climbing back down to Earth'.

Turns (Curve Segments)

Defining the center of the small circle for the curve segment uses calculations as if creating a line segment. The method rotates the vector $\underline{a}$ 90^o to create a 'fake azimuth' vector $\underline{fa}$ which points towards where the center must be located (for constructing either a right–clockwise–turn or a left–counterclockwise–turn). The position vector $\underline{r}$, together with the vector $\underline{fa}$, form a set just as the one required to find the end position when moving over a great circle. The desired radius R for the curve determines where the center will be created. The position vector $\underline{c}$ (of the curve's center) is then found. Vector $\underline{rc}$ is the projection of $\underline{r}$ on $\underline{c}$, and $\underline{t} = \underline{r} - \underline{rc}$. The set $\underline{t}$, $\underline{n}$, $\underline{c}$ form a left-handed coordinate system T-N-C; the segment being constructed over the plane T-N. The final position vector $\underline{p}$ (see Figure 5), normalized by R.E.A., is obtained by

$\underline{p} = \underline{t}_f + \underline{c}$, where $\underline{c}$ is obtained through the line segment method, and

$\underline{t}_f = [\cos\gamma, \sin\gamma, 0]\,[\underline{t}, \underline{n}, \underline{c}]^T$, where

γ = S / $|\underline{t}|$, (S being the desired flying distance)

$[\underline{t}, \underline{n}, \underline{c}]^T =$

$[[t_x, t_y, t_z], [n_x, n_y, n_z], [c_x, c_y, c_z]]\,[\underline{x}, \underline{y}, \underline{z}]^T$

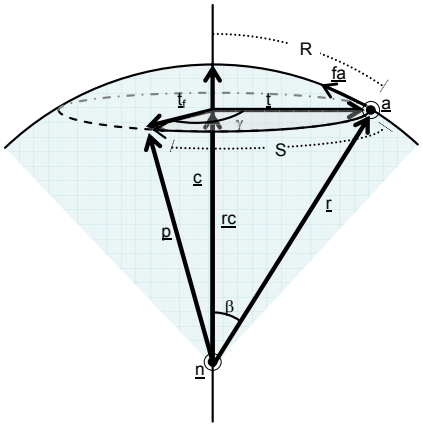

Figure 5. Curve Segment Geometry for a Right (or Clockwise) Turn

($\underline{n}$ being the normal vector generated through the line segment method when obtaining $\underline{c}$).

The final state azimuth is obtained through the *getDirectionTo* method (below) from the endpoint to the center point, and subtracting or adding 90^{o} if the turn is CW or CCW, respectively.

Other Geometric Calculations

Two other main computations are widely used. The *getDistanceTo* method allows the computation of the straight distance between two points. The constraint here is that the distance must be in accordance with the circular way straight segments (i.e. climbs) are defined. This distance (see Figure 6) is given by

$D = \beta \times R.E.A.$, where

$\beta = 2 * \text{asin}(L /(2*R.E.A.))$

and by the cosine rule:

$L = \text{sqrt}[\, p_1^2 + p_2^2 - 2*p_1*p_2*\cos\alpha]$

This method is restricted to cases where $L \leq 2*R.E.A$ due to the circular geometry used.

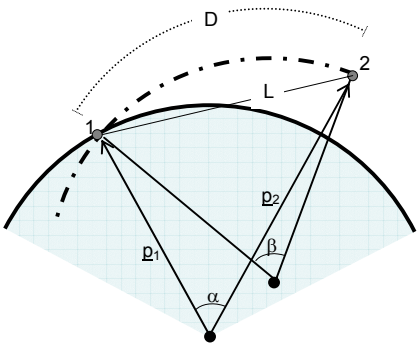

Figure 6. Computed Distance Between Two Points.

The *getDirectionTo* method computes the azimuth at point 1 with which following a great circle path would reach point 2 (not requiring altitude change). This angle is given by

$\text{azimuth} = \text{asin}(\cos\Omega / \cos\theta_1)$, where

$\cos\Omega = n_z / |\underline{n}|$

where $\underline{n} = \underline{p}_1 \times \underline{p}_2$ is the vector normal to the plane defined by the position vectors, and n_z the z component of this normal vector (Ω is the angle between $\underline{n}$ and the vertical z axis).

Other Assumptions

Continuity between segments is maintained through end-start position and end-start speed. Heading continuity is maintained whenever a segment is added to a series of previous segments. When joins between two segments are required, heading continuity is currently not enforced for long-range simulations. This should not have considerable effect as segments (~40km, variable) are much larger than the minimum achievable turn radius of the vehicle (~100m). This reduces computation time as calculations become simpler. Similarly, the elevation angle change between a line or curve followed by a climb segment is assumed to be instantaneous.

Joins between two existing segments are first attempted using five segments of similar length. On the first fifth of the distance, a climbing segment tries to match speed and altitude (based on the vehicle's capabilities). If the attempt succeeds, the remaining distance is joined using four straight segments. If the attempt fails, the speed and altitude match is attempted on one fourth of the distance; if it fails again it tries on one third of the distance and so on. If neither of these are possible, attempts are made to join to a different segment. This avoids the rigidity of joining only with a single segment for future mutations.

Feeding the Path Planner with Realistic Wind Information

Dealing with air vehicles requires consideration of winds and airspeeds. In real life winds are far from being invariant in space and time, and thus any path planning process must be adaptable to these changes. And as the ultimate goal is to have these methods applied to actual UAV flights, realistic weather information must be provided to the planner. GRIB (GRIdded Binary) format is a World Meteorological Organization (WMO) standard for the exchange and storage of weather data. Therefore the path planner has been designed to extract atmospheric information necessary for the planning process from GRIB data files.

Extracting GRIB Formatted Weather Data

GRIB records consist of six different sections which identify and define the data contained[1].

IS – indicator section

PDS – product definition section

GDS – grid description section

BMS – bit map section

BDS – binary data section

End section

Each section's information is classified and contained in specific octets (bytes). The first section identifies the file and its length. Except for the first and last sections (which are invariant in length), each section's first three octets indicate its length, and distinguish each section from another. For the purposes of the planner, three of these sections are the main source of information: PDS, GDS and BDS.

The PDS section contains, among other things, information regarding the initial time of the data (UTC), the time intervals of data and the total time range included, the weather parameters in the report, grid type used, and whether a GDS section is included. The GDS section may provide a detailed description of the 2-D grid type used. The BDS section contains the parameters and data values themselves, which are given at different pressure altitude levels. This defines a complete 4-D array of data.

A vast amount of weather information is stored in GRIB formatted files. Since the parameters contained may vary from source to source, a pre-processing step has been developed in order to standardize the input data used by the path planner. The path planner requires one input file per weather parameter (i.e. temperature) for a time frame which contains the expected complete flight time. This separation process was developed with the aid of a freely distributed utility for GRIB data extraction called wgrib[2]. As both forecast data (from the GFS forecast model) and observed data (NCEP-NCAR Reanalysis data) are stored and distributed in GRIB format, only slight differences between them (such as the time intervals of the data) must be taken into account during preprocessing. Once the data goes through this pre-processing step, the information is ready to be read by the path planner.

For the purpose of the simulations presented here, the only information used by the planner was horizontal wind velocity at seven different pressure levels (UGRD and VGRD parameters). A standard atmosphere is assumed for the geometric-altitude to pressure-altitude relationship.

Planning with Variable Wind Data

UTC 'launch time' is provided to the path planner to determine the applicable weather forecast files to use for the flight. Once these files are identified, the planner stores all the 2-D spatial array data for each pressure level and measurement (or forecast) time into a separate 4-D array for each weather parameter to be used.

At the evolver initialization stage, the initial state of the wind field is determined using launch time and vehicle position to interpolate horizontal wind vector components from the 4-D arrays created from the weather data files. Once the initial wind field state and time have been determined, the path planner starts generating the first population of paths (as described in the initialization process). Every time that a new segment is created, the planner estimates the time it will take to traverse it and stores the end clock time as part of the segment's information. Estimations are computed at a sub-segment level based on length, airspeed at the sub-segment's start point (determined by the final airspeed of the previous sub-segment for continuity purposes), a randomly defined end airspeed (or pre-determined if connecting to an existing segment), and the wind field at the initial state of the sub-segment to estimate the groundspeed when traversing the sub-segment's length. This estimation (which certainly depends on the number of sub-segment discretization levels and the rate of change of the wind field) provides the approximate time at the end of the sub-segment, which is then used to determine the wind field at that state (latitude, longitude, altitude) for the next sub-segment. In addition, after a mutation process (which can modify a path at any point along its length), the planner re-computes the estimated end times for all

[1]NCEP, Office note 388
http://www.nco.ncep.noaa.gov/pmb/docs/on388/
[2]http://wesley.wwb.noaa.gov/wgrib.html

follow-on segments in order to determine new wind fields corresponding to their new 4-D locations.

This time-variable consideration allows the planner to compute fuel consumption by taking into account the wind fields that the aircraft will encounter at the moment it estimates it will arrive at each location. Thus, when going through the fitness evaluation and selection process, it is able to generate better solutions for the complete flight.

Dealing with Wind Uncertainty.

In actual flight, the wind fields for future segments of the flight path cannot be precisely known; rather they can only be estimated from available forecast data. Each forecast contains a complete set of horizontal wind field components at fixed grid locations and pressure levels for future times (forecast intervals). As mentioned previously, the planner linearly interpolates this information to provide itself with wind field estimates at any given location and time. Moreover, the planner may be injected with updated forecast data (as it typically occurs every six hours) and incorporate new information into its continuous in-flight planning process. In this way, updated (and presumably more accurate) forecasts can begin to influence "future" segments of the flight path—those segments not yet converted into trajectories. This feature allows the planner to adapt to possible changes in the expected environment.

To exploit this feature and allow for path refinement even if no important changes occur to the expected environment, planning is not confined to a pre-launch exercise. Rather, continuous planning and re-planning is performed during the entire flight. At a predetermined time interval a "present" path section is selected as the next actual trajectory to be flown, but the remaining "future" path segments continually evolve until the next time interval. This system allows modifications to the planned path if better solutions are found along the way.

The Airplane's Flight Performance and Fuel Consumption Characteristics

In order to assure that the airplane is capable of traversing a path, every segment is constrained by the vehicle's performance characteristics (i.e. maximum rate of climb, climb angle, speed, altitude, etc.). The planner does not deal with vehicle dynamics, and it assumes the vehicle maintains course.

Because mutations in any segment may affect the total length of a path, the weight of the vehicle (and thus its capabilities) will vary while traversing subsequent unmodified segments. For this reason, the performance characteristics used to constrain every segment are determined by the initial weight of the vehicle. However, weight changes during the flight are considered in the planner's fuel consumption computations (the UAV consumes less fuel as it gets lighter). Therefore, whenever a path is modified at any segment (through the mutation process), the vehicle's weight is recalculated for the remaining segments along that path (similarly to the time re-computation step mentioned earlier). The fuel consumption model uses linear interpolations from a table of fuel use rate (kg/s). Rates for climbs (and descents) are determined by altitude, weight and slope under the assumption that the UAV uses full throttle to climb (idle for descent). Rates for level flight are based on altitude, weight and speed. Mutations of segments are not restricted only to their geometry, but may include the speed at which to fly them, thus taking advantage of different fuel consumption rates.

The performance table used was constructed based on a very detailed performance model of the Aerosonde. The use of accurate aircraft performance data in this planning system adds another layer of realism to the close-to-real case simulations demonstrated in the following section.

Honolulu, HI to Long Beach, WA Simulations through Real Wind Fields

Several simulation cases are presented here with the intention of showing different characteristics of the behavior of the planning system. The first cases demonstrate situations in which the planner has been provided with wind field information for every place and time and has generated a good overall plan. These simulations are followed by a case in which new wind information is provided to the planner during flight, which would occur as updated forecast information became available.

The simulation results presented in Figures 7 through 11 utilized historical (reanalysis) wind data covering a mission with launch time on March 23, 1999, 00:00 UTC. The airplane started with a full fuel tank, and its flight is bounded between 300m and 4500m altitude. Initial altitude is set at 300m; final altitude at the goal is not constrained to a specific value. Initial planning consisted of 100 generations; replanning during the flight was set to run for 100 generations every three hours (thus, the aircraft repeatedly selects three-hour trajectories to fly while replanning the rest of the path). The handicap for the losing evolver was also set to 100 generations each time. Parent and population parameters were set to 8 and 16, respectively. Only the best path of each evolver is shown. The wind fields shown correspond to the altitude at which the aircraft is currently positioned. Each figure presents the planner's estimated fuel requirement at the given state.

Figure 7 shows the initial path plan before launch. Figure 8, 9 and 10 depict three snapshots during the flight. Figure 11 presents the completed path. Note from the figures how the actual flight followed a path that resembles one of the initially planned paths. This shows how the planner was able to consider future winds that it estimated it would encounter during the flight. Furthermore, en-route re-planning continued with both evolvers achieving refinements to the planned path. It is clear from the figures that the aircraft is well-positioned based on the winds at each time.

While the available wind information does not contain vertical wind field components, the variations in the horizontal (dominant) wind field at different altitudes are taken into account by the planner. Figure 12 presents the altitude history of this flight showing that the planner is able to exploit all three dimensions.

To compare this solution against a shorter flight (in distance), the simulation was repeated with the aircraft forced to fly along a straight path, though still allowed to alter its altitude and speed. Figure 13 shows that the total fuel consumption for this path was higher than the path freely generated by the planner of Figure 11. Figure 14 shows that the planner decided once again to fly at various altitudes for this second case even when laterally restricted to fly straight to the goal.

In order to compare what the planner could achieve while restricting the straight plan to a constant altitude (and only being able to vary its airspeed to affect fuel rates), Figure 15 is presented. In this case, the planner was restricted to fly straight at 1160m, the average altitude flown in the simulation of Figure 13. The estimated fuel consumption for this case is clearly higher.

These simulations demonstrate how the planner can plan for a complete flight based on future wind information. A different more realistic case will be demonstrated next. In this case, even when the planner has initial pre-flight forecast information on future wind fields, updated forecasts will become available during flight.

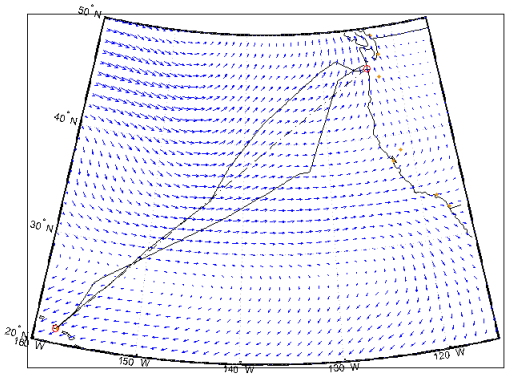

Figure 7. Initial Plan (Mar 23, 1999, 00:00UTC)

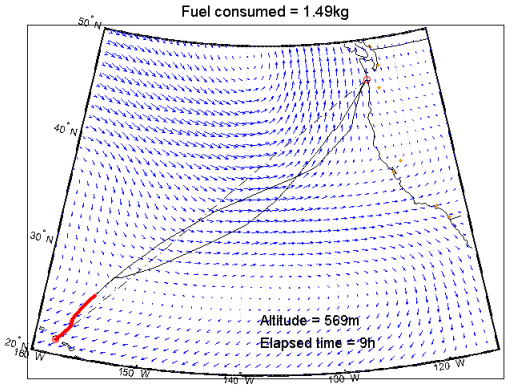

Figure 8. State at +9hrs

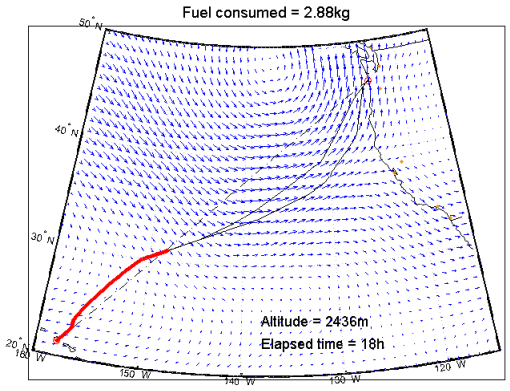

Figure 9. State at +18hrs

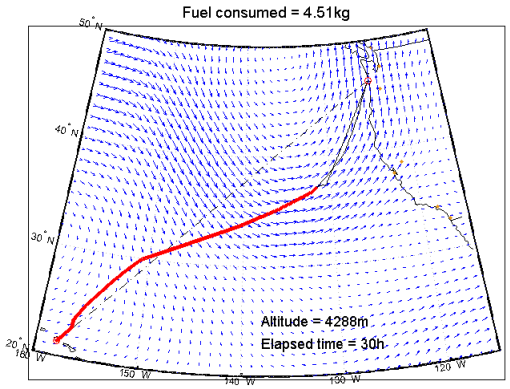

Figure 10. State at +30hrs

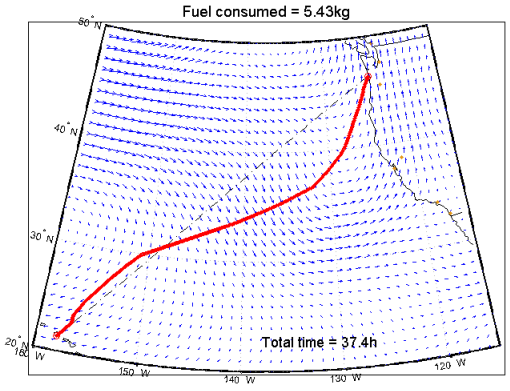

Figure 11. Flight Completion (+37.4hrs)

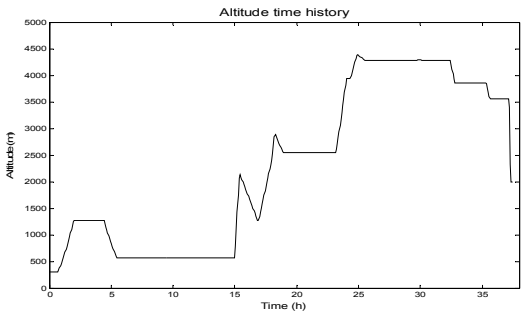

Figure 12. Altitude history of the flight.

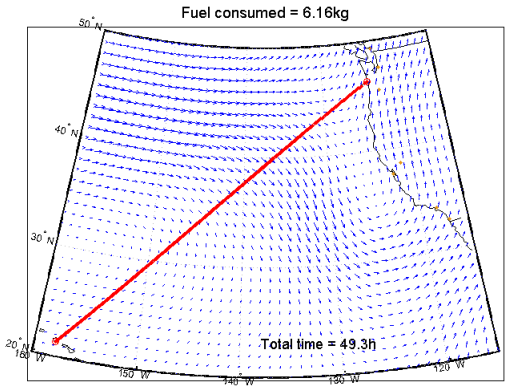

Figure 13. Straight Flight Comparison Case

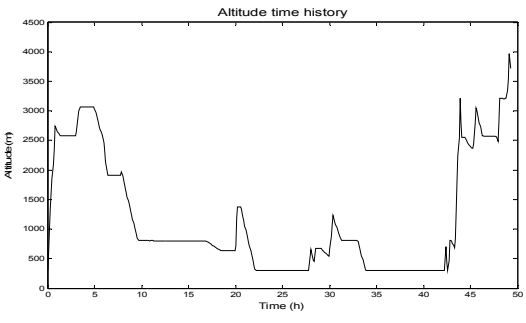

Figure 14. Altitude history of the flight.

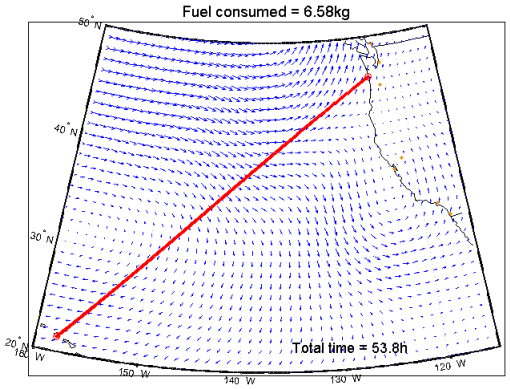

Figure 15. Straight Flight Case at 1160m

For the following simulation, historical weather data was again used as the source of wind information. Using this data would give the planner access to wind field information at any given location and time, failing to present any 'sudden surprises' to the planner during flight. In the interest of demonstrating a more realistic situation, the following results were obtained by simulating 'new wind field forecasts' at predetermined intervals. This was accomplished by advancing the airplane's clock at each interval so that it considered wind data corresponding to future times. In effect, this forced the planner to readapt the path to a new 'forecast' different from one it had previously considered. This is similar to what would occur if updated forecast information were provided the planner.

Figures 16 through 22 present several snapshots of the simulated flight sequence. Initial conditions were set to February 5, 1997 at 00:00 UTC; the initial altitude was now set at 4500m. Updated forecast data was simulated by advancing the weather data by twelve hours at every trajectory selection interval, which was set to occur every six hours. For this case, 200 generations were run in each step. Two independent evolvers were used.

These figures show how the initial plan suggested a nearly straight path to the goal. It is in general maintained for the first 18 hours of flight, even though wind time had already been advanced twice in 12hr intervals (thus suggesting that even the 'new forecasts' were not affecting the plan much). However, Figure 18 shows one of the evolvers suggesting a plan that deviates from the originally selected path due to more intense forecasted wind changes. The planner decides to take this alternate route as shown in Figure 19. The remainder of the path did not require a second major change even when the wind time was again advanced twice. Once more, note from the figures how the aircraft is well-positioned based on the wind fields present at each time. Figure 22 shows the complete route. Figure 23 presents the altitude history of this flight.

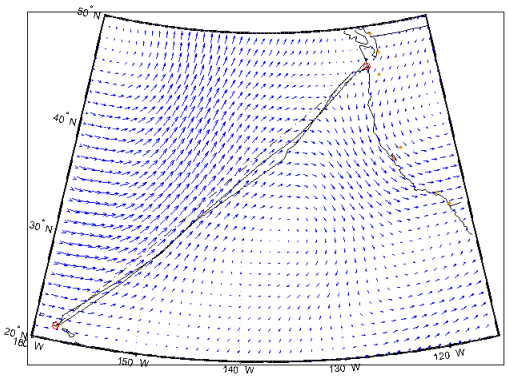

Figure 16. Initial Plan (Feb 5, 1997, 00:00UTC)

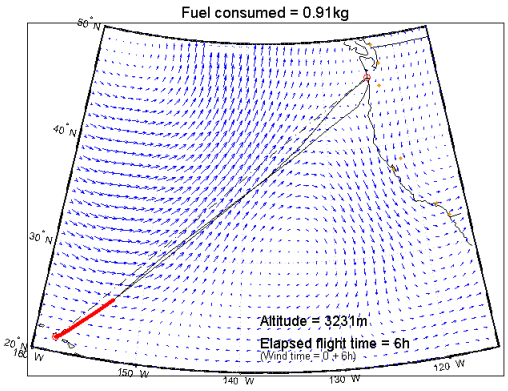

Figure 17. State at +6hrs

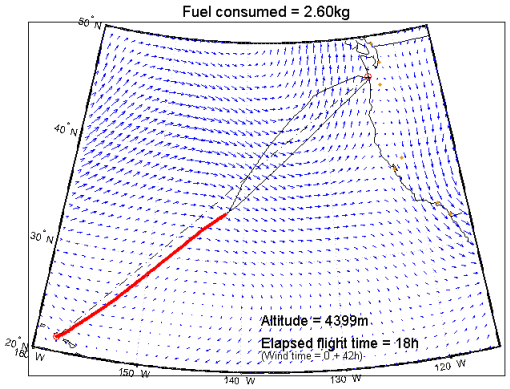

Figure 18. State at +18hrs

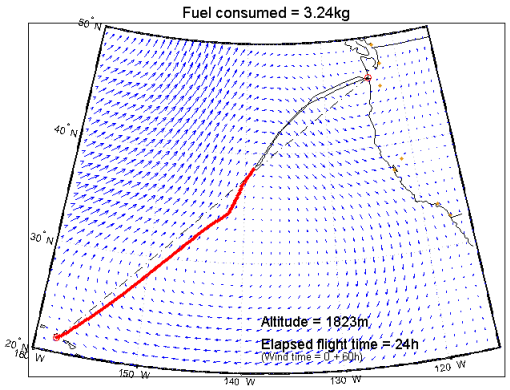

Figure 19. State at +24hrs

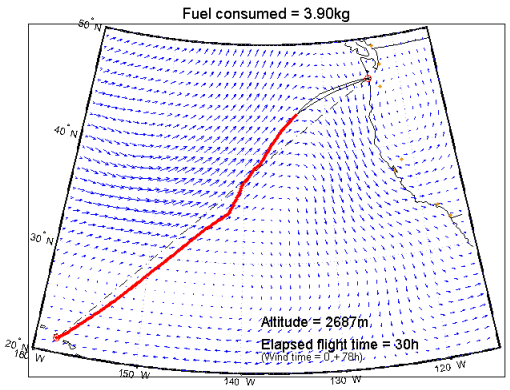

Figure 20. State at +30hrs

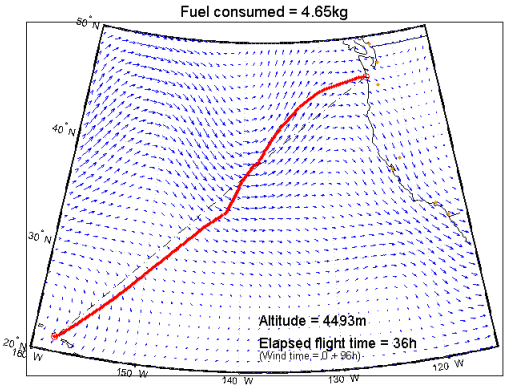

Figure 21. State at +36hrs

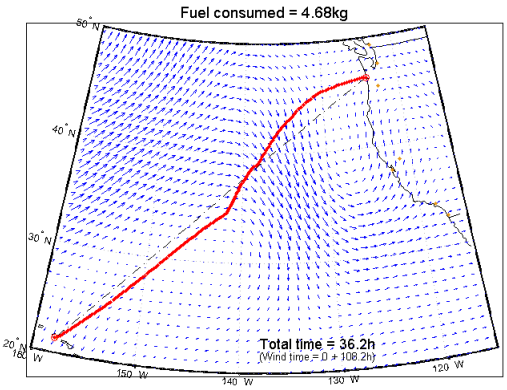

Figure 22. Flight Completion (+36.2hrs)

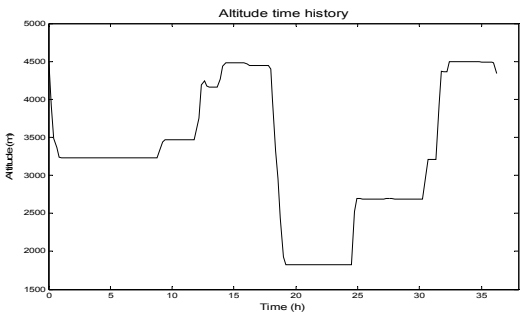

Figure 23. Altitude history of the flight.

One final comparison case is presented. Figure 24 shows a solution for a situation similar to the one presented in Figures 7 through 11 (launch on March 23, 1999 at 00:00 UTC). For this case, however, the path planning process uses flat Earth geometries as presented in [4]. The total time and estimated fuel requirement (by comparison with those in Figure 11) are clear indicators that this long range application must consider more than a simple flat earth model.

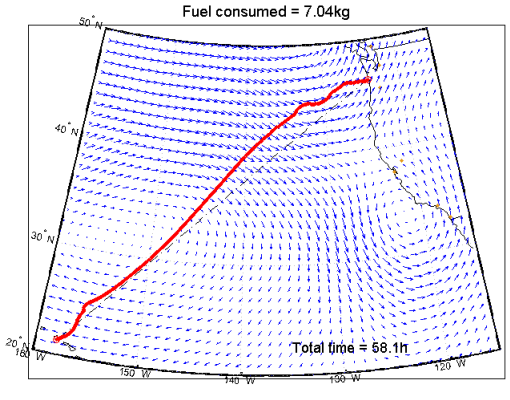

Figure 24. Simulation with a Flat Earth Model (to compare with Figure 11)

It is important to comment that although the aircraft used in these simulations can only hold 5kgs of fuel, simulation results that estimate larger fuel requirements have been presented. It was of interest for this paper to present results that could clearly show the behavior of the planner in different situations, regardless of the specific feasibility of the simulated flights. However, simulated Honolulu to Long Beach flights using actual wind data from different dates have been performed in which the estimated fuel requirements are within the vehicle's capacity.

Summary

An evolution-based path planner has been used to simulate the planning of a UAV long range flight through variable wind fields. The use of actual wind information available in GRIB format is implemented for planning purposes. The planner combines UAV fuel burn characteristics together with favorable tail winds to obtain path solutions that reduce fuel consumption over the entire flight. Forecasted wind fields are taken into consideration to plan for the complete flight. En-route constant replanning and the use of two evolvers allow path refinement and adaptation to updated forecast information. The planner uses a spherical Earth model for long range applications.

Acknowledgments

The research presented in this paper is funded by the NOAA National Sea Grant College Program under Grant NA16RG1044, project code R/0T-22.

The first author would also like to acknowledge the support received from CONACYT (Mexico).

References

[1] McGeer, Tad, Juris Vagners, 1999, Historic crossing: an unmanned aircraft's Atlantic flight, GPS World 10(2): 24-30

[2] Fogel, D.B., and L.J. Fogel, 1990, Optimal Routing of Multiple Autonomous Underwater Vehicles through Evolutionary Programming, Proc. of the 1990 Symposium on Autonomous Underwater Vehicle Technology, Washington, D.C., pp. 44-47.

[3] Capozzi, Brian, Juris Vagners, 2001, Evolving (semi)-autonomous vehicles, Proc. of the 2001 AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada.

[4] Rathbun, David, et.al., 2002, An Evolution Based Path Planning Algorithm for Autonomous Motion of a UAV through Uncertain Environments, Proc. of the 21st Digital Avionics Systems Conference, Irvine, CA.