2013 American Control Conference (ACC)
Washington, DC, USA, June 17-19, 2013

UAV Flight Path Planning in Time Varying Complex Wind-fields

Anjan Chakrabarty and Jack Langelaan

Abstract— **This paper addresses the problem of path planning for a small UAV operating in a complex four dimensional (time and spatially varying) environment. A kinematic tree path planner that explicitly accounts for time is presented. This kinematic planner is shown to be resolution complete through comparison with the RC-RRT, and simulation results demonstrating planning in a complex time- and spatially-varying wind field are presented. The simulation considers an unpowered (i.e. gliding) aircraft, hence exploitation of vertical components of wind is critical for feasibility of flights to the goal.**

I. Introduction

Wind is a potential source of energy which can be exploited to increase the range and duration of flights. Birds and human glider pilots regularly utilize this wind energy to stay aloft and reach their destination. However, wind (like all natural occurring phenomena), varies both temporally and spatially. The spatial variation of wind makes the problem of planning an efficient trajectory extremely complicated. The temporal variation further adds to the complexity. Thus the energy optimal path is both time and space dependent.

Energy is available from atmosphere mainly through vertical air motion, spatial velocity gradients and gusts (essentially temporal velocity gradients). The time scale and magnitudes of these energy gains also vary considerably. Static soaring exploits vertical air motion, whose time scale is of order minutes to hours; dynamic soaring exploits spatial gradients and vehicle dynamics become important; and gust soaring requires high rate control. Static soaring phenomenon suits both the time scale and the amount of energy that can be extracted from the atmosphere for long range path planning.

Vertical air motion mainly occurs due to uneven heating of the ground or by deflection of air by the side of mountain ranges. Cyclic oscillations of winds are noticed in the lee of mountains called the mountain wave are also potential source of upward moving air.

Path planning for air vehicles in the presence of winds has been widely studied. In 1931 Zermelo solved the problem of optimal navigation of small ships in presence of currents [1]. Recently the time optimal problem has been adopted as a Markov Dubin problem as suggested by Sussmann [2]. In presence of steady uniform winds the Zermelo-markov-dubins (ZMD) problem has been solved by McGee and Hedrick using Maximum Principle of Pontryagin [3], [4]. In [5] Bakolas has also given a time optimal synthesis for the ZMD problem. However in all these cases only steady uniform wind is considered; furthermore only minimum time trajectories are examined.

This paper focuses on the problem of trajectory planning in non-uniform, unsteady wind fields. Further, the focus is on *minimum energy* flight paths, but the techniques developed here can be extended to other optimization criteria. Because of the significant spatial and temporal variation in the wind field the state space of the vehicle is very large, leading to a computationally intensive trajectory planning problem. Sampling based path planning techniques [6] are widely used to deal with this "curse of dimensionality". Variants such as Rapidly exploring Random Trees (*RRT*) [7] are widely successful in solving path planning problems in static and dynamic environments. Although these planning techniques are probabilistically complete there is no guarantee of optimality for the solutions. Variants such as *RRT** [8] has been used for optimal motion planning in static environments. In [9] Ardiyanto et al. used arrival time fields and heuristically random trees to deal with time complexity. Since there is no guarantee of reaching the goal from a particular starting time, arrival time fields cannot be used this application.

A tree-based approach to planning in complex wind fields was introduced in [10]. Here the kinematic tree is extended by: (1) adding an explicit representation of time; (2) adding a continuous turn motion primitive, so that altitude can change with only a small change in horizontal position; (3) demonstrating resolution completeness of this kinematic tree.

Resolution complete means that if a path exists then a tree of sufficient resolution will find it: this is analogous to probabilistic completeness of randomized motion planners.

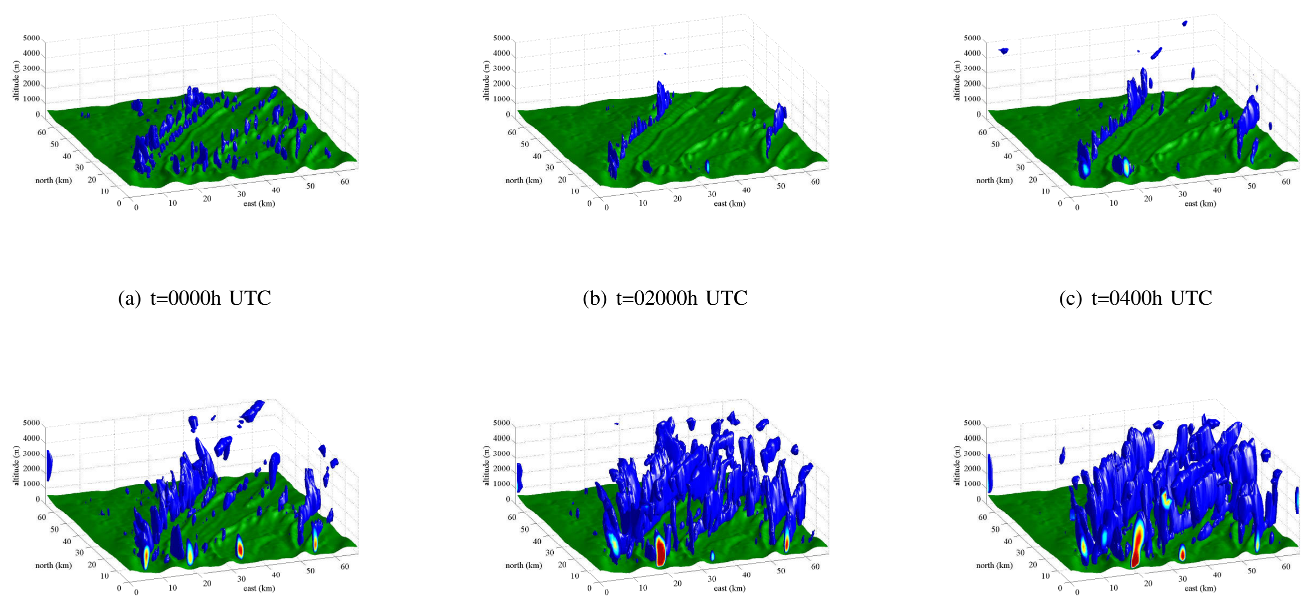
This paper is organized as follows: Section II describes the time- and spatially-varying wind field that is used as the motivating example; Section III describes the kinematics of soaring flight; Section IV describes the kinematic tree algorithm; Section V compares the kinematic tree and RRT-based approaches, demonstrating resolution completeness of the kinematic tree; Section VI presents results of planning in the 4D wind field; finally Section VII presents concluding remarks.

II. A Dynamic Wind field

The wind field used as the unsteady example for this research is shown in Figure 1. The wind field data was generated using WRF (Weather Research and Forecasting tool, currently the state of the art numerical weather prediction tool) version 2.2. The wind field was provided by the

J Langelaan is with Faculty of Aerospace Engineering, The Pennsylvania State University, University Park, State College,PA, USA jlangelaan@psu.edu

A Chakrabarty is a Graduate Student of Aerospace Engineering, The Pennsylvania State University, University Park, State College,PA, USA anjan@psu.edu



(a) t=0000h UTC (b) t=02000h UTC (c) t=0400h UTC

(d) t=0600h UTC (e) t=0800h UTC (f) t=1100h UTC

Fig. 1. Visualization of wind field data for a complex time varying wind field.

authors of [11], and it is a computational analysis of an actual event that occurred on October 7, 2007. This wind field shows the development of ridge lift and mountain wave over central Pennsylvania. The wind field gives east, north, up components of wind at 0.44km grid spacing horizontally. The vertical resolution is descending in nature with more density near the surface and gradually decreasing with altitude. Wind is provided in intervals of 15 minutes starting at 0000 UTC on October 7,2007 and ending at 1200 UTC on October 7, 2007 [11].

The visualization of Figure 1 shows regions where the vertical component of the wind vector is greater than the minimum sink rate of the glider used in this research, i.e. regions where energy can be harvested. Blue isosurfaces bound energy harvesting regions, with subfigures (a) through (e) showing the time evolution of the wind field. Note the significant spatial as well as temporal variation of the wind field, leading to a particularly challenging planning problem. Clearly a "good" path planning algorithm will find trajectories that fly through these regions while avoiding regions of downwards moving air.

III. Kinematics of soaring Flight

It is assumed that an on-board controller is capable of several control modes, including constant airspeed flight, constant heading flight and constant bank angle (i.e. turning) flight. As in [10] it is also assumed that the response to step changes in commands is fast compared with the duration of a particular command. A kinematic model is therefore sufficient to describe vehicle motion. Kinematics of steady straight flight (i.e. constant heading, constant airspeed) are derived in [10]; this is briefly summarized here and extended to steady turning flight.

While only gliding flight will be considered later, here kinematics are derived including thrust. Later thrust will be set to zero. Vehicle kinematics are given by

$$\begin{array}{rcll}
\dot{x} & = & v_a \cos\gamma \cos\psi + w_x & (1)\\
\dot{y} & = & v_a \cos\gamma \sin\psi + w_y & (2)\\
\dot{z} & = & v_a \sin\gamma + w_z & (3)\\
\dot{\psi} & = & \eta & (4)
\end{array}$$

where v_a is the air speed, γ the glide path angle ψ is the heading and and η is the rate of change of heading. Components of the wind velocity vector are denoted by w_x, w_y, and w_z

Consider the force equation of the aircraft which is in a steady bank angle of ϕ and flight path angle γ (Figure 2). Equating the forces in parallel and perpendicular to the the flight path,

$$\begin{array}{rcll}
mg\sin\gamma & = & D - T\cos\alpha_i & (5)\\
mg\cos\gamma & = & L\cos\phi + T sin\alpha_i & (6)
\end{array}$$

where α_i is the incidence angle between thrust vector and the flight path (note that if the thrust axis is aligned with the aircraft's body x axis, $\alpha_i = \alpha$, the aircraft's angle of attack). Assuming small flight path angle γ and $\alpha_i = 0$ (i.e, thrust is aligned to the flight path angle, so that thrust has negligible

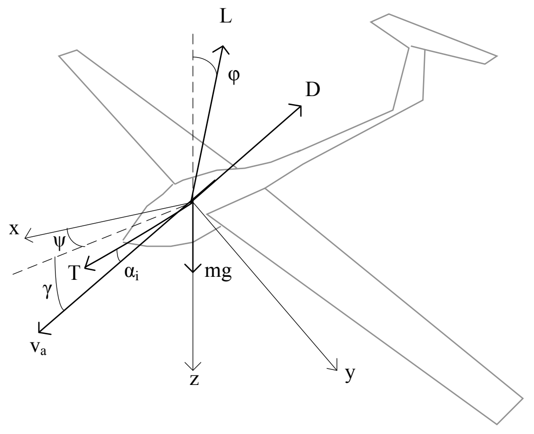


Fig. 2. Point mass model.

contribution to force perpendicular to flight path),

$$mg\gamma = D - T \tag{7}$$

$$mg = L\cos\phi \tag{8}$$

Using lift coefficient defined as $L = \frac{1}{2}\rho v^2 S C_L$,

$$C_L = \frac{2mg}{\rho v^2 S \cos\phi} \tag{9}$$

The drag coefficient can be expressed as a polynomial function of lift coefficient:

$$C_D = \sum_{i=0}^{n} a_i C_L^i \tag{10}$$

and with $D = \frac{1}{2}\rho v^2 S C_D$ the flight path angle is thus

$$\gamma = \frac{\rho v^2 S}{2mg} \sum_{i=0}^{n} a_i C_L^i - \frac{T}{mg} \tag{11}$$

To find the turn rate consider the horizontal components of the forces perpendicular to the flight path.

$$L\sin\phi = m v_a \dot{\psi} \tag{12}$$

Dividing Equation 8 by Equation 12 gives,

$$\tan\phi = \frac{v_a \dot{\psi}}{g} \tag{13}$$

thus,

$$\dot{\psi} = \frac{g\tan\phi}{v_a} \tag{14}$$

Thus the general equations of motion become

$$\dot{x} = v_a \cos\gamma \cos\psi + w_x \tag{15}$$

$$\dot{y} = v_a \cos\gamma \sin\psi + w_y \tag{16}$$

$$\dot{z} = v_a \sin\gamma + w_z \tag{17}$$

$$\dot{\psi} = \frac{g\tan\phi}{v_a} \tag{18}$$

Thus the flight path is completely specified by the inputs $\mathbf{u} = [T\ v_a\ \psi\ \phi]^T$ and the wind vector $\mathbf{w} = [w_x\ w_y\ w_z]^T$.

IV. Tree Based Approach to Path Planning

The kinematic tree was outlined in [10]. The tree considered in this paper has explicit representation of time, which enables handling of complex time varying wind fields and is briefly outlined here.

The tree is initialized at the vehicle start position and time. From this start configuration the tree is expanded by computing a set of reachable configurations based on vehicle kinematics. At the start position one of the allowable motion primitives is zero velocity, so that the position remains constant but time varies. This encodes the possibility of delaying launch until a more favorable time.

The set of configurations that are reachable after some time interval Δt defines nodes in the tree. Each node encodes inertial position, time, heading, airspeed, a cost for that node and the distance from the node to the goal.

$$n_i = [x_i\ y_i\ z_i\ t_i\ \psi_i\ v_{a,i}\ C_i\ r_{goal,i}] \tag{19}$$

The tree is expanded incrementally by selecting a node and computing the set of configurations reachable from that node. To save computation time the motion primitives are pre-defined and computed based on a set of allowable inputs.

A. Motion Primitives

The set of motion primitives used to build the tree is pre-computed to reduce the computational time during incremental build of the tree.

A particular input $\mathrm{u} \in \mathrm{U}$ (where U is the set of allowable inputs) is

$$\mathrm{u}_{ijkl} = [T_i\ v_j\ \Delta\psi_k\ \phi_l] \tag{20}$$

where each component is chosen from a discrete set of allowable inputs:

$$T_i \in [T_1\ T_2\ T_3\ \dots\ T_I] \tag{21}$$

$$v_j \in [v_1\ v_2\ v_3\ \dots\ v_J] \tag{22}$$

$$\Delta\psi_k \in [\Delta\psi_1\ \Delta\psi_2\ \Delta\psi_3\ \dots\ \Delta\psi_K] \tag{23}$$

$$\phi_l \in [0\ \phi_L] \tag{24}$$

where $\Delta\psi$ represents a change a change in heading followed by a straight flight at the beginning of the flight segment. To ensure “reasonableness” of the reachable configurations certain restrictions are are placed on allowable combinations:

$$0 \le T_i \le T_{max} \tag{25}$$

$$v_{min} \le v_j \le v_{max} \tag{26}$$

$$\text{if } \phi \ne 0 \text{ then } \Delta\psi = 0 \text{ and } v = v_{min} \tag{27}$$

Here v_{min} is chosen to be the minimum power flight condition (equivalent to minimum sink speed for a glider.)

Given a choice of input $\mathrm{u} \in \mathrm{U}$ and a time Δt the motion primitives are computed using numerical integration.

The motion primitives for the SB-XC glider used in simulations are computed using the following sets of allowable

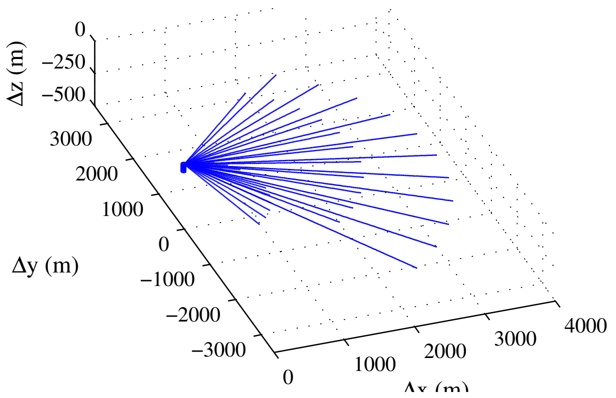


Fig. 3. Motion Primitives in zero Wind.

inputs and $\Delta t = 120$ seconds.

$$T = 0 \tag{28}$$

$$V_a = [15\ 20\ 25\ 30\ 35] \tag{29}$$

$$\Delta\psi = [-50^\circ\ -40^\circ\ \ldots\ 40^\circ\ 50^\circ] \tag{30}$$

$$\phi = [0^\circ\ 30^\circ] \tag{31}$$

Figure 3 shows the motion primitives in zero wind. Note the small spiral motion primitive given by the steady bank angle motion. In steady state bank, the air velocity of the glider is the best sink rate airspeed. Thus the loss in altitude for this motion primitive is smallest compared to other motion primitives. This spiral motion essentially allows altitude change with no change in x and y co-ordinates.

B. Node Selection and Expansion

Nodes are selected on the basis of survival of the fittest. A cost function C_i dictates which nodes are selected for expansion. The cost function C_i assigned to each node is energy altitude h_E divided by the distance to goal. where,

$$h_E = h + \frac{v_a^2}{2g} \tag{32}$$

and

$$C_i = \frac{h_E}{r_{goal}} \tag{33}$$

This cost function ensures that the nodes chosen minimize the distance to the goal and maximize the total energy (either in terms of altitude gain or velocity gain).

Note that selecting the best node for expansion is a greedy approach and may lead to dead nodes but this leads to fast solutions if they exist. Another possible approach to node selection is a weighted random method, where "good" nodes are more likely to be chosen [10]. This results in more exploration of the space at the cost of slower convergence to a solution if there are few "dead ends" in the environment.

The selected node is expanded using the motion primitives and wind speeds at each node. The net motion is the sum of wind vector and zero-wind motion primitives. The wind vector is obtained for the selected node by linear interpolation in both space and time from the wind field, and is assumed to be constant over Δt. The reachable configuration is computed for the selected node and the net motion as in [10]. Feasible (i.e, collision free) reachable configurations are added to the tree.

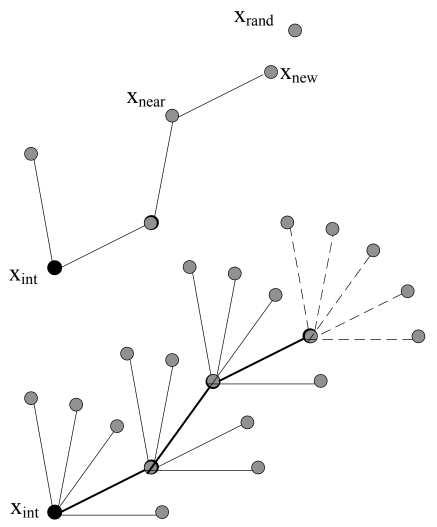


Fig. 4. Comparison of growth of RC- RRT (upper) and Kinematic Tree (lower)

C. End Game

The tree is terminated and successful path is reported once the tree finds a node within the gliding distance of the goal. The end game region (X_{goal}) is defined as

$$X_{goal} = \left\{ x : \frac{r_{goal}}{\delta t} \leq V_{nom}\ \&\ \frac{r_{goal}}{\Delta h} \leq \frac{L}{D}\Big|_{nom} \right\} \tag{34}$$

One can choose V_{nom} and $\frac{L}{D}\big|_{nom}$ to specify the size of the end game region in terms of nominal vehicle performance.

V. Comparing the Kinematic Tree and RRT

In this section the rationale behind using the tree based planner in comparison to other popular path planning techniques like the RRT is shown. A theoretical behavior of the planner is analyzed and that the planner inherits the resolution completeness of an RC-RRT is shown.

The resolution complete rapidly exploring random tree (RC-RRT) was introduced by Chang et al. in [12] and was shown to be resolution complete in [13]. However the RC-RRT has not been used in time varying cases.

The RC-RRT algorithm incrementally builds upon a tree $G = (V, E)$ defined by its set of vertices V and edges E. A random configuration is picked from the robots free configuration space (X_{free}). The vertex that is closest to the random configuration is chosen for expansion. Then all the available control inputs are checked to see which control brings the robot closest to the random configuration.If the path joining them is collision free then that new point is added to the vertex set V and the edge joining the points is added to the edge set E. This process is repeated until the goal is reached.

```
Function :Kinematic_tree($x_{init}$);
G.not_extended($x_{init}$);
for $i=1$ to $K$ do
    $x_{rand\_state} \leftarrow random\_state(G.not\_extended)$;
    G.extended($x_{rand\_state}$);
    Extend($x_{rand\_state}$)
end
Return
Function :Extend($x_{rand\_state}$);
$X_{next\_states} \leftarrow Steer(x_{rand\_state}, U, \Delta t)$;
for all $x_{state} \in X_{next\_states}$ do
    if $collision\_free\_path(x_{random\_state}, x_{state})$ then
        $G.not\_extended.add\_node(x_{state})$;
        $G.add\_edges(x_{rand\_state}, x_{state}, u)$)
    end
end
Return G
```

Algorithm 1: The Kinematic Tree Algorithm

The kinematic tree algorithm not only includes the resolution completeness of an RC-RRT but also explicitly encodes time and grows towards favorable regions. Instead of picking a random configuration from X_{free} a random state (x_{rand_state}) is picked directly from the non-extended part of the Tree. Then the *Steer* function generates the reachable configurations (X_{next_states}) by expansion of the random state using the motion primitives. Now from all the states the ones that are collision free are added to the tree. This process is repeated until the end game region is reached.

Lemma 5.1: for all $i \in N$, $V_i^{RC-RRT} \subset V_i^{tree}$ and $E_i^{RC-RRT} \subset E_i^{tree}$

Lemma 1 implies that the paths discovered by RC-RRT algorithm after each iteration is subset of those discovered by the tree algorithm.

Proof: To prove the above lemma let us assume that the function $random\_state(G)$ in algorithm 1 picks the same state for expansion as determined by the *near* procedure by algorithm of RC-RRT. Note that x_{near} "can" also be randomly chosen. This is again guaranteed by the resolution complete property of RC-RRT. Taking x_{near} as x_{goal} one can always find a vertex in RC-RRT which gives us the desired node. If this assumption holds true for all i it is easy to see $V_i^{RC-RRT} \subset V_i^{tree}$ and $E_i^{RC-RRT} \subset E_i^{tree}$ ■

Theorem 5.2: If there exists a feasible solution from x_{int} to x_{goal} then $lim_{i \to \infty} P(V_i^{RC-RRT} \cap X_{goal} \neq \emptyset) = 1$

For proof of theorem 5.2 a see [13]

From Lemma 5.1 and theorem 5.2 the following theorem is immediate.

Theorem 5.3: The probability that the TREE initialized at x_{int} will contain x_{goal} as a vertex approaches 1 as the number of vertices approach infinity

Remark 5.4: The kinematic tree algorithm thus provides a solution if one exists. It must be noted that the computation of obstacle avoidance for each of the branches is computationally expensive, but in this case only height above ground is concerned. Moreover the tree is biased in such a way that the nodes away from obstacles (the ground in this case) are chosen to be expanded. Also a lot of computation time is saved by avoiding sorting of nodes in *near* function like in RC-RRT. The big advantage of using the tree based planner is that time is inherently integrated in the way the tree is built. Also note that the motion primitives are precomputed to reduce computation time.

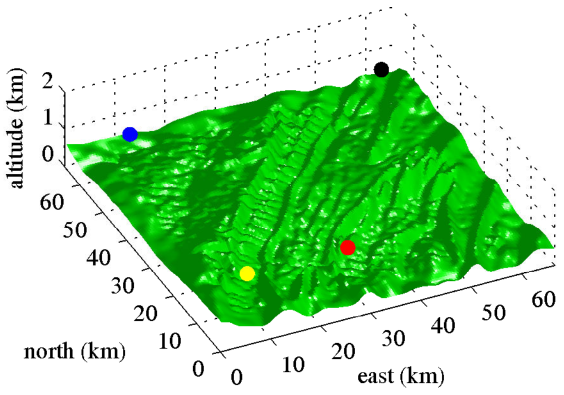


Fig. 5. Scenario For simulation. The goal is shown by the black dot and start regions are shown by blue,yellow and red dots.

VI. Simulation Results

Here a flight to a distant goal is considered from different starting positions (Figure 5). The starting locations are *blue* [10 60 1.5] km, *yellow* [10 10 1] km and *red* [30 10 1] km. The goal location is a distant [60 60 1]km given by the *black* dot. The average ground elevation is approximately 500 m. Thus given the starting altitude of 1km (500 m from the ground), the maximum gliding distance is only 12 km (at best L/D). Clearly to reach the goal wind energy has to be utilized.

A vehicle representative of the RnR Products SB-XC is used here: parameters are given in Table I. Simulations were carried out on 2.6GHz dual core Intel processor.

TABLE I
Parameters for SB-XC glider.

variable	value	description
m	10 kg	mass
S	1 m^2	wing area
$f(C_L)$	$0.1723C_L^4 - 0.3161C_L^3 + 0.2397C_L^2$ $-0.0624C_L + 0.0194$	
$v_{a,min}$	12 m/s	
$v_{a,max}$	35 m/s	
$L/D\vert_{max}$	25	best glide ratio

Figure 6 shows all the paths starting from different starting locations at different starting times. Reachability of goal varies considerably with starting location and start time. While almost all the paths starting from start location *yellow* reach the goal, paths starting only at specific times make

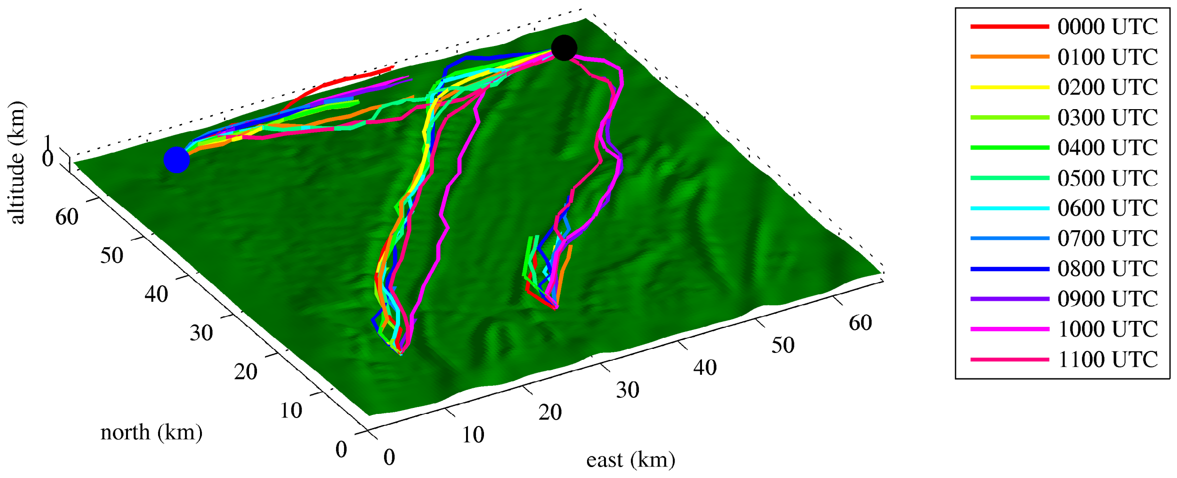


Fig. 6. Trajectories starting at Different Times of Day.

TABLE II
SIMULATION RESULTS FOR DIFFERENT START POINTS.

Start Point	**Earliest departure time for successful arrival**	**Time of flight**
blue	0600 UTC	36 mins
red	0900 UTC	42 mins
yellow	0100 UTC	32 mins

TABLE III
TIME OF TRAVEL AND MINIMUM DISTANCE TO GOAL FOR TRAJECTORIES STARTING AT DIFFERENT TIMES OF THE DAY

Start Time	**Time of Travel**	**Closest Distance to goal**	**Status**
UTC	(minutes)	(km)	
0000	12	52.030	failed
0100	42	0.0	X_{goal} reached
0200	40	0.0	X_{goal} reached
0300	12	51.220	failed
0400	42	0.0	X_{goal} reached
0500	48	0.0	X_{goal}reached
0600	50	0.0	X_{goal} reached
0700	12	54.898	failed
0800	48	0.0	X_{goal} reached
0900	4	61.123	failed
1000	36	0.0	X_{goal} reached
1100	42	0.0	X_{goal}reached

it to the goal from start locations *blue* and *red*. Because the winds are both temporally and spatially varying the feasibility of gliding (i.e. unpowered flight) from the various start positions to the goal changes with time. Note that the starting altitude of the *blue* starting point is higher than the other two. Simulation results have shown that no paths reach the goal for *blue* starting points with altitude of 1 km.

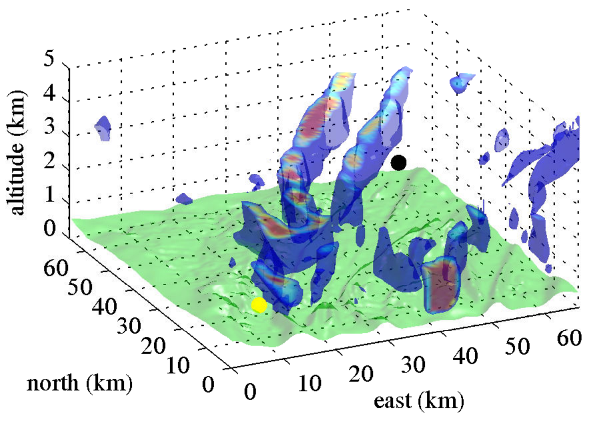
Table II tabulates the earliest start times from the different locations that successfully reach the goal. Paths starting at the point *blue* reaches the goal only for starting times of 0600 UTC , 0700 UTC and 1100 UTC and the fastest to reach the goal starts at 0600 UTC. Paths starting at any other times fail to rach the goal from *blue* starting point. Paths starting at *red* reaches the goal only after the wave has fully developed. All the paths starting after 0900 UTC reach the goal. Paths starting at *yellow* are analyzed in details next.

Table III shows the comparison of the flights forced to start at different times of day (i.e. the null transition was disallowed) for the start point *yellow*. As seen from the results trajectories starting only at specific time of the day actually reach the goal. Among the paths that reach the goal, the one starting at 0100 UTC reach the goal fastest. Thus to make this flight the optimal start time of travel would be 0100 UTC, while the time of travel is shortest for trajectories starting at 1000 UTC. The effect of wind on the paths computed is clearly visible(Figure 6). The paths tend to follow the ridges showing evidence of ridge lift along the ridges. Some of the paths, the ones starting at 0300, 0700 and 0900 UTC, end very quickly because there is no vertical air motion of sufficient strength near the start point at those times..

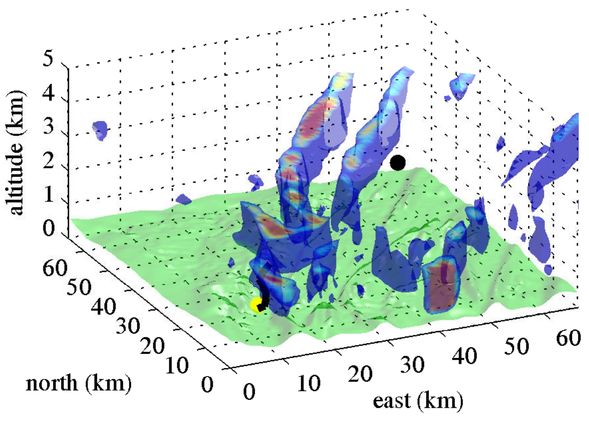
Figure 7 shows the time evolution of the trajectory of the flight starting at at 1100 UTC. Recall that the blue isosurface regions are the regions where the glider can gain energy. Clearly the planner was able to utilize these regions of upward moving air. Note that the wind updates after every 15 mins and thus the the change in environment is updated only in 15 minutes intervals. This represents approximately 14 km travelled between wind field time updates. Note that if higher update rate is available it can be exploited.

VII. CONCLUSIONS

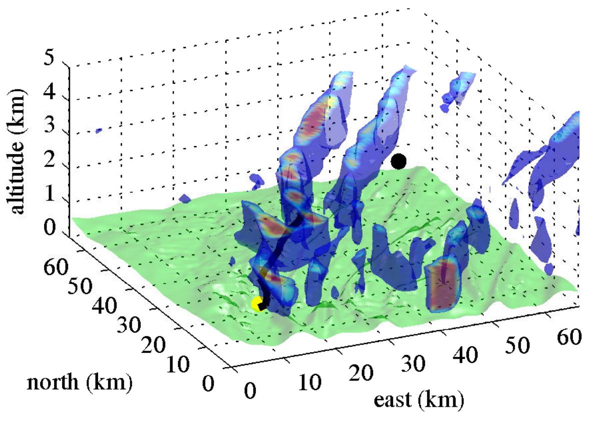
This paper has described the kinematic tree, an approach to path planning that can explicitly model time varying, complex environments. The paper also has demonstrated some theoretical insights into the resolution completeness of the tree based planner. The tree based planner inherits the resolution completeness of an RC-RRT and at the same time,



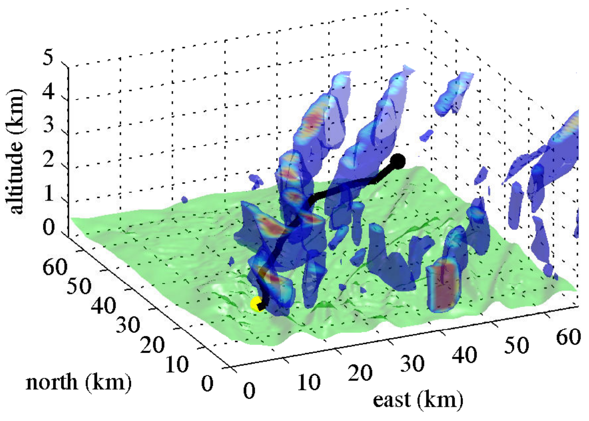
(a) t=1100h UTC



(b) t=115h UTC



(c) t=1130h UTC



(d) t=1145h UTC

Fig. 7. Flight starting at 1100 UTC for *yellow* starting position.

time is inherently embedded in the tree structure. The time varying tree has demonstrated the successful utilization of energy available from the atmosphere to get to the destination only by utilizing winds aloft.

VIII. ACKNOWLEDGMENTS

This research was funded by the National Science Foundation under Grant IIS-0746655.

REFERENCES

[1] Zermelo, E., “Über das Navigationsproblem bei ruhender oder veränderlicher Windverteilung,” *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, Vol. 11, No. 2, 1931, pp. 114–124.

[2] Sussmann, H. J., “The Markov-Dubins problem with angular acceleration control,” *In Proccedings of the 36th IEEE Conference on Decision and Control*, IEEE Publications, 1997, pp. 2639–2643.

[3] McGee, T. and Hedrick, J., “Optimal path planning with a kinematic airplane model,” *Journal of guidance, control, and dynamics*, Vol. 30, No. 2, 2007, pp. 629–633.

[4] McGee, T., Spry, S., and Hedrick, J., “Optimal path planning in a constant wind with a bounded turning rate,” *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Citeseer, 2005, pp. 1–11.

[5] Bakolas, E. and Tsiotras, P., “Time-Optimal Synthesis for the Zermelo-Markov-Dubins Problem: the Constant Wind Case,” *Proceedings of the American Control Conference*, Baltimore, Maryland, June 2010.

[6] LaValle, S., *Planning algorithms*, Cambridge Univ Pr, 2006.

[7] LaValle, S. and Kuffner Jr, J., “Randomized kinodynamic planning,” *The International Journal of Robotics Research*, Vol. 20, No. 5, 2001, pp. 378–400.

[8] Karaman, S. and Frazzoli, E., “Incremental sampling-based algorithms for optimal motion planning,” *Proc. Robotics: Science and Systems*, 2010.

[9] Ardiyanto, I. and Miura, J., “3D Time-space Path Planning Algorithm in Dynamic Environment Utilizing Arrival Time Field and Heuristically Randomized Tree,” 2012.

[10] Langelaan, J. W., “Tree-Based Trajectory Planning to Exploit Atmospheric Energy,” *Proceedings of the American Control Conference*, Seattle, Washington, June 2008.

[11] Young, G., Gaudet, B., Seaman, N. L., and Stauffer, D. R., “Interaction of a mountain lee wave with a basin cold pool,” *13th Conference on Mesoscale Processes*, AMS, American Meteriological Society, Chicago, Illinois, 2009.

[12] Cheng, P. and LaValle, S., “Reducing metric sensitivity in randomized trajectory design,” *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, Vol. 1, IEEE, 2001, pp. 43–48.

[13] Cheng, P. and LaValle, S., “Resolution complete rapidly-exploring random trees,” *Robotics and Automation, 2002. Proceedings. ICRA’02. IEEE International Conference on*, Vol. 1, IEEE, 2002, pp. 267–272.