

MIT Open Access Articles

*Wind Uncertainty Modeling and Robust
Trajectory Planning for Autonomous Parafoils*

The MIT Faculty has made this article openly available. **Please share**
how this access benefits you. Your story matters.

Citation: Luders, Brandon et al. "Wind Uncertainty Modeling and Robust Trajectory Planning for Autonomous Parafoils." *Journal of Guidance, Control, and Dynamics* 39.7 (2016): 1614–1630.

As Published: <http://dx.doi.org/10.2514/1.G001043>

Publisher: American Institute of Aeronautics and Astronautics

Persistent URL: <http://hdl.handle.net/1721.1/105151>

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Terms of use: Creative Commons Attribution-Noncommercial-Share Alike



Wind Uncertainty Modeling and Robust Trajectory Planning for Autonomous Parafoils

Brandon Luders¹ and Aaron Ellertson² and Jonathan P. How³
Massachusetts Institute of Technology, Cambridge, MA, 02139, USA

Ian Sugel⁴
Orbital Sciences Corporation, Dulles, VA, 20166, USA

A key challenge facing modern airborne delivery systems, such as parafoils, is the ability to accurately and consistently deliver supplies into difficult, complex terrain. Robustness is a primary concern, given that environmental wind disturbances are often highly uncertain and time-varying. This paper presents a new on-line trajectory planning algorithm that enables a large, autonomous parafoil with under-actuated dynamics to robustly execute collision avoidance and precision landing on mapped terrain, even with significant wind uncertainties. This algorithm is designed to handle arbitrary initial altitudes, approach geometries, and terrain surfaces, and is robust to wind disturbances that may be highly dynamic throughout terminal approach. Real-time wind modeling and classification is used to anticipate future disturbances, while a novel uncertainty-sampling technique ensures that robustness to future variation is efficiently maintained. The designed cost-to-go function enables selection of partial paths which intelligently trade off between current and reachable future states, while encouraging upwind landings. Simulation results demonstrate that this algorithm reduces the worst-case impact of wind disturbances relative to state-of-the-art approaches.

Nomenclature

(A, B, C, D)	= vehicle lag dynamics
(A_c, B_c)	= wind model dynamics for c th wind class
I	= identity matrix
\mathbb{I}	= indicator function
L_D	= vehicle lift-to-drag ratio
$\mathcal{N}(\hat{a}, P_a)$	= normal/Gaussian distribution (mean \hat{a} , covariance P_a)
N_c	= number of wind classes
N_p	= number of propagated heading rates (cost-to-go)
N_s	= number of covariance samples
P	= 2D state covariance
\mathbb{P}_a	= probability over random distribution a
Q	= position covariance, m ²
Q_I	= initial position covariance, m ²
R_{\min}	= vehicle minimum turning radius, m
\mathcal{U}	= input constraints
\mathcal{X}	= state constraints
dt	= planning timestep duration, s
m	= mean wind filter window width
\mathbf{p}	= (p_x, p_y, p_z) = vehicle position, m
$\hat{\mathbf{p}}$	= nominal/noise-free vehicle position, m

¹ PhD, Department of Aeronautics and Astronautics, Member AIAA, luders@mit.edu

² SM Candidate, Department of Aeronautics and Astronautics, ace2@mit.edu

³ Richard C. Maclaurin Professor of Aeronautics and Astronautics, Associate Fellow AIAA, jhow@mit.edu

⁴ Engineer, Control System Engineering, Sugel.Ian@orbital.com

\mathbf{p}_G	= goal position, m
\mathbf{p}_I	= vehicle initial position, m
p_{safe}	= minimum required probability of feasibility
\mathbf{s}	= vehicle lag states
t	= timestep, s
t_F	= terminal planning time, s
t_I	= initial planning time, s
\mathbf{u}, \mathbf{u}	= actuation/control inputs
\mathbf{v}	= vehicle velocity, m/s
\mathbf{v}	= wind model noise
v_0	= nominal vehicle velocity at sea level, m/s
\mathbf{w}	= (w_x, w_y, w_z) = wind disturbance/state, m/s
$\hat{\mathbf{w}}$	= nominal/noise-free wind state, m/s
$\bar{\mathbf{w}}$	= $(\bar{w}_x, \bar{w}_y, \bar{w}_z)$ = 3D mean wind estimate, m/s
\mathbf{w}_I	= most recent wind observation, m/s
\mathbf{x}	= vehicle state
$\hat{\mathbf{x}}$	= nominal/noise-free vehicle state
\mathbf{x}_G	= goal state
\mathbf{x}_I	= vehicle initial state
Σ	= wind covariance, m^2/s^2
Σ_I	= initial wind covariance, m^2/s^2
Φ	= wind feature vector
β	= mean wind filter optimization coefficient
$\delta \mathbf{w}$	= $(\delta w_x, \delta w_y)$ = 2D wind variation estimate
$\delta \mathbf{x}$	= $(\delta p_x, \delta p_y, \delta w_x, \delta w_y)$ = 2D state variation
θ	= wind direction, rad
λ	= wind regularization coefficient
ρ	= wind magnitude, m/s
σ	= uncertainty ellipse standard deviation(s)
τ	= cost-to-go propagation time, s
τ_{max}	= maximum cost-to-go propagation time, s
ϕ	= trajectory path cost
ϕ_F	= trajectory terminal cost
ψ	= vehicle heading, rad
ψ_I	= vehicle initial heading, rad
$\dot{\psi}_d$	= desired heading rate, rad/s

I. Introduction

Modern airborne delivery systems must be able to accurately and consistently deliver supplies into difficult, complex terrain. For delivery systems such as parafoils, the terminal guidance problem – guiding the parafoil from a potentially high altitude to land precisely with a desired position and heading – presents significant technical challenges, particularly for large/heavy parafoils such as those considered in this work. Parafoil dynamics are highly nonlinear and under-actuated, with large turning radii and severely limited (if any) vertical control, resulting in a descent rate which is influenced primarily by atmospheric conditions and disturbances. Parafoil drop locations are often difficult to reach and may be surrounded by arbitrary, non-convex terrain that can pose a significant problem for constraint satisfaction, even if mapped in advance. Parafoils are also subject to uncertain and variable wind environments, which, if uncompensated, can result in large deviations between predicted and actual trajectories, leading to unacceptable landing errors. Finally, many

airborne delivery applications often have tight landing restrictions; missing the target location, even by a small distance, can lead to unintended collisions with natural or man-made hazards, or even theft of cargo. Precise delivery is essential to avoid loss of supplies or unacceptably dangerous recovery efforts.

Work on terminal guidance for autonomous resupply can be largely subdivided into two categories. The first category, glide-slope-based planning, utilizes the concept of the glide-slope surface or cone: the set of all position/heading states which, assuming constant velocity and disturbances, would guide the parafoil to the goal state. Calise and Preston [1] utilize a series of scripted maneuvers online to estimate glide-slope parameters, then execute turning maneuvers to drive the parafoil to the glide-slope. This provides a useful approach trajectory, but the framework heavily constrains the solution space and requires long-term glide-slope tracking beginning from a large initial distance. Additionally, the presence of terrain obstacles is not considered during the path planning process. This makes the approach sensitive to uncertainty in both the vehicle dynamics and environment, especially given that the glide-slope surface shifts as a function of current wind conditions. Slegers et al. [2] track the glide-slope using nonlinear model predictive control (MPC), improving rejection of small-scale disturbances, but also require long-term glide-slope tracking. Bergeron et al. [3] use feedback control, known as glide-slope guidance (GSG), to drive the approach to the goal based on the estimated glide-slope and wind conditions. This minimizes the effect of coupled system uncertainty and ensures a maximum heading deviation from the estimated wind direction. Recent efforts by Ward and Costello [4] have also demonstrated the potential for improved glide-slope tracking through both online system identification, and by exploiting the longitudinal control coupling between incidence angle and symmetric brake deflection. To summarize, while the above approaches [2–4] take some measures to account for the effect of wind uncertainty on the parafoil landing position, they offer no robustness to interaction with terrain obstacles, and are subject to the fundamental constraining of the solution space imposed by the glide-slope approach paradigm.

Trajectory-based approaches, on the other hand, generate arbitrary reference trajectories online to optimize a pre-specified cost function, utilizing various control schemes to track these trajectories. Gimadieva [5] formulates parafoil terminal guidance as an optimal control problem and establishes the necessary conditions for optimality, but the resulting approach lacks the computational efficiency needed for real-time implementation, thus making it unable to adjust for varying wind conditions and model uncertainties during flight.

The band-limited guidance (BLG) algorithm [6] uses direct optimization via Nelder-Mead simplex search to minimize a cost function of the predicted terminal vehicle state. BLG guarantees that control bandwidth constraints are satisfied to ensure accurate trajectory following, and its computational efficiency enables the use of online replanning, making it effective for many nominal wind and terrain conditions. However, BLG is fundamentally limited in its starting altitude due to high dimensionality and optimization scalability, thus constraining mission flexibility. BLG incorporates no notion of wind variation in its planner, instead relying on reactive replanning to address unexpected wind effects. Additionally, its direct optimization technique does not consider the possibility of off-nominal, adverse terrain interactions caused by changing wind conditions, particularly on complex terrain maps.

The Inverse Dynamics in the Virtual Domain (IDVD) algorithm developed by Yakimenko and Slegers [7] utilizes inverse dynamics to connect the initial vehicle state to the target terminal state, while guaranteeing the terminal conditions of the resulting nonlinear boundary value problem (BVP) are satisfied. While computationally efficient, this approach cannot guarantee satisfaction of control bandwidth constraints, requiring iteration in order to ensure the planned trajectory can be tracked by the controller. This method also relies on rapid, reactive replanning to offset uncertainties during execution, but assumes constant wind during planning. Recent extensions to the IDVD algorithm [8] consider altitude-varying and/or higher-dimensional wind profiles during terminal descent, accounting for both cross-track winds and updrafts/downdrafts. For each of these formulations, however, the BVP assumes a known, deterministic wind profile; robustness to future wind variations is captured only through replanning, rather than explicit modeling. Additionally, the effect of terrain geometry is not considered during the design of feasible descent trajectories.

Subsequent work by Rogers and Slegers considers robustness to wind variations by utilizing graphics processing units (GPUs) to parallelize a Monte Carlo simulation of possible future winds, and the resulting parafoil trajectories, based on available measurements [9]. However, significant computational effort is required to run these GPU simulations online. Within each set of simulations, the solution space is restricted to a limited number of candidate solutions of the original BVP, where each candidate assumes a constant-rate turn and terminal heading constraint, and is simulated over a set of constant wind profiles. Based on these assumptions, online replanning is used to correct for the effect of future wind disturbances. While such an approach effectively incorporates the overall, trajectory-wide wind effect, each simulation assumes a deterministic wind. As a result, this method

does not model the possibility of dynamic wind changes during the planning process, potentially making it overly optimistic.

Recent work by Fowler and Rogers considers the use of Bezier curves to perform optimized path planning/replanning for a small parafoil in three-dimensional obstacle fields [10]. This approach offers geometric flexibility to online trajectory planning by adjoining multiple cubic Bezier curves, which are used to navigate constrained terrain environments. However, the computation of the proposed optimization is shown to scale poorly with the number of degrees of freedom (*i.e.*, the control points of the adjoined Bezier curves) and convergence is sensitive to the initial guess. These factors may limit the effectiveness of this approach for environments of increased complexity and/or initial altitude. In addition, path feasibility and terrain collisions are only assessed for the nominal planned trajectory under the assumption of mean wind, with reactive replanning used to mitigate the effects of future wind disturbances. Because some replanning iterations may require significant computation time in order to converge, such an approach can render the parafoil vulnerable to terrain collisions due to uncompensated wind effects. In these situations, safety in complex terrain environments cannot be guaranteed.

In summary, the general body of parafoil terminal guidance algorithms is subject to some or all of the following limitations: (1) an artificially-constrained solution space, often based on preconceived notions of the solution form; (2) implicit or explicit constraints on the initial altitude, which require a prior descent phase to bring the parafoil to initial conditions suitable for successful terminal guidance; and/or (3) a reactive approach to handling the effect of wind uncertainty.

The algorithm developed in this paper addresses these limitations, giving a real-time trajectory planner that enables a large, autonomous parafoil to robustly execute collision avoidance and precision landing on mapped terrain, even with significant wind uncertainties. This algorithm is designed to handle arbitrary initial altitudes, approach geometries, and terrain surfaces, and is robust to wind disturbances that may be highly dynamic throughout terminal approach, including updrafts and downdrafts. Though the proposed algorithm is not optimal, it can quickly identify robust feasible solutions taking the parafoil near the target location (including possible terminal heading constraints), and then use the remaining computation time to refine and update the solution quality to reflect subsequent observations.

This paper builds upon chance-constrained rapidly-exploring random trees (CC-RRT), an online framework for robust motion planning in cluttered, non-convex environments [11]. CC-RRT

leverages the benefits of sampling-based algorithms and particularly rapidly-exploring random trees (RRT) [12] (*e.g.*, incremental construction, trajectory-wise constraint checking, rapid exploration, dynamically feasible trajectories), while using chance constraints to ensure probabilistic feasibility with guaranteed, user-specified bounds. Through trajectory-wise constraint checking, CC-RRT can efficiently evaluate the risk of constraint violation online due to multiple sources of both internal and external uncertainty.

This paper presents three contributions to the RRT framework which, compared to state-of-the-art algorithms, yields superior performance on the parafoil terminal guidance problem. First, a novel wind uncertainty model is developed, using real-time observed wind data to classify and anticipate the wind uncertainty environment online. The resulting model is shown to accurately represent true wind behavior, while adjusting the conservatism of the guidance algorithm to reflect prevailing conditions. Second, this multi-class wind model is utilized to derive the analytic *a priori* uncertainty distribution over future possible trajectories. This is leveraged in a novel variation of the CC-RRT path planner, which performs analytic sampling of uncertainty distributions for constraint checking, ensuring robust avoidance of undesirable collisions with arbitrary, possibly aggressive terrain maps. Finally, the relative value of paths is assessed via a novel terminal cost-to-go function, which utilizes a fixed-horizon discrete approximation of the parafoil reachability set. This enables selection of partial paths from any altitude that intelligently trade off between current and reachable future states.

Extensive simulation results show the effectiveness of each of these components, and demonstrate that the full parafoil CC-RRT algorithm can achieve superior landing accuracy in both average-case and worst-case performance relative to state-of-the-art algorithms such as band-limited guidance [6]. In particular, we demonstrate that the analytic-sampling approach achieves higher robustness to wind than replanning and/or mean-wind estimation alone. Building on previous work [13], results show that parafoil CC-RRT is capable of operating in real-time while preserving these robustness properties. The planner is also shown to be largely invariant to changes in altitude or terrain. Lastly, this work extends previous developments [13] to consider terminal heading constraints, by incorporating landing speed penalties into the cost-to-go function. Simulation results demonstrate that these penalties can significantly reduce landing speed in order to encourage upwind landings, with minimal effect on accuracy.

II. Problem Statement

The terminal guidance paradigm typically utilizes a combination of a homing phase, designed to steer the parafoil directly toward the target, and an energy management phase, designed to descend the parafoil above the target until an appropriate altitude is reached for terminal guidance [6, 7, 10]. Such algorithms typically assume that terminal guidance will begin in relative proximity to the target location, in both lateral distance and altitude. Though the approach in this work often operates under similar conditions, such assumptions are not necessary. The parafoil CC-RRT algorithm, introduced below, will guide the parafoil as close to the target as possible from any initial conditions.

The vehicle state is represented as $\mathbf{x} = \begin{bmatrix} \mathbf{p}^T & \psi & \mathbf{s}^T \end{bmatrix}^T$, where $\mathbf{p} = (p_x, p_y, p_z)$ is the position, ψ is the heading, and \mathbf{s} represents a vector of any additional states needed to characterize the parafoil's motion – in this case, the lag dynamics. In the terminal guidance problem, the objective is to guide the parafoil from some initial position \mathbf{p}_I and heading ψ_I (full state \mathbf{x}_I) to some target location \mathbf{p}_G (full state \mathbf{x}_G). The parafoil dynamics are represented as the nonlinear state-space system

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{w}), \quad \mathbf{x}(t_I) = \mathbf{x}_I, \quad (1)$$

where t_I is the initial time, \mathbf{u} are the control inputs and $\mathbf{w} = (w_x, w_y, w_z)$ are the wind disturbances. The specific parafoil model used in this work is detailed in Section II A.

The wind disturbances are unknown at current and future times; denote the most recent wind observation as \mathbf{w}_I (if none have been taken, a prior value from forecasting data may be applied, or simply assume $\mathbf{w}_I = 0$). In this work, we choose to represent the wind disturbances using the generalized model

$$\dot{\mathbf{w}} = f_w(\mathbf{w}, \bar{\mathbf{w}}, \mathbf{v}), \quad \mathbf{w}(t_I) = \mathbf{w}_I, \quad (2)$$

where $\bar{\mathbf{w}}$ is an estimate of the mean wind, assumed to be available to the planner, and \mathbf{v} is unknown model noise. The wind model developed for this work is derived in Section III.

The parafoil terminal guidance problem is a specific case of a more general trajectory planning problem. At each timestep, the path planner attempts to solve the optimal control problem

$$\min_{\mathbf{u}} \quad \phi_f(\hat{\mathbf{x}}(t_f), \mathbf{x}_G) + \int_{t_I}^{t_f} \phi(\hat{\mathbf{x}}, \mathbf{x}_G) \quad (3)$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{w}), \quad \mathbf{x}(t_I) = \mathbf{x}_I, \quad (4)$$

$$\dot{\hat{\mathbf{x}}} = f(\hat{\mathbf{x}}, \mathbf{u}, \hat{\mathbf{w}}), \quad \hat{\mathbf{x}}(t_I) = \mathbf{x}_I, \quad (5)$$

$$\dot{\mathbf{w}} = f_w(\mathbf{w}, \bar{\mathbf{w}}, \mathbf{v}), \quad \mathbf{w}(t_I) = \mathbf{w}_I, \quad (6)$$

$$\dot{\hat{\mathbf{w}}} = f_w(\hat{\mathbf{w}}, \bar{\mathbf{w}}, 0), \quad \hat{\mathbf{w}}(t_I) = \mathbf{w}_I, \quad (7)$$

$$\mathbf{u} \in \mathcal{U} \quad \forall t, \quad (8)$$

$$\mathbb{P}_{\mathbf{v}}(\mathbf{x} \in \mathcal{X}) \geq p_{\text{safe}}. \quad (9)$$

The parafoil state $\hat{\mathbf{x}}$ and wind state $\hat{\mathbf{w}}$ evolve according to (1) and (2) respectively assuming $\mathbf{v} \equiv 0$, representing deterministic, nominal propagation of the dynamics under the assumption of constant wind $\bar{\mathbf{w}}$. This is utilized simply to ensure that the objective (3) being optimized is deterministic (Section VI), though stochastic forms may be used.

The sets \mathcal{U} and \mathcal{X} represent constraints on the input and state, respectively. The state constraints \mathcal{X} must be *probabilistically* satisfied, *i.e.*, satisfied with probability of at least p_{safe} over all possible \mathbf{v} , as represented in (9) by $\mathbb{P}_{\mathbf{v}}$. These constraints include the terrain map $T(p_x, p_y)$, which is assumed to be perfectly known; the terminal time t_F is the time at which $p_z \leq T(p_x, p_y)$. Additional state constraints may be included, such as internal state bounds or no-fly zones, though this is not explored further in this work. The stochastic elements of this optimization manifest themselves only in the final chance constraint (9); Section IV details the implementation of this chance constraint, yielding a deterministic optimization.

In practice, the optimization (3) is solved repeatedly during the descent, with \mathbf{x}_I and \mathbf{w}_I being set to the most recent state and wind measurements, respectively, at current time t_I . The algorithm developed in this work functions identically regardless of the extent of planning that previously took place. For example, it is not required to be preceded by a guidance phase which places the parafoil in a favorable initial state. As a result, the framework developed in this paper can be incorporated within a variety of planning architectures.

A. Parafoil Model

The parafoil is modeled as a Dubins vehicle [14] descending at a rate governed by atmospheric conditions subject to updrafts/downdrafts, with the input-to-heading-rate mapping governed by complex lag dynamics. The lack of altitude control, coupled with a large minimum turning radius and slow turning rate, necessitates significant advance planning for precision guidance and landing. This is exacerbated by the presence of heavy winds, which can lead to loss of goal reachability

and/or premature terrain collisions if not properly anticipated.

The parafoil velocity $v(p_z)$ is assumed to be a function of the vehicle altitude p_z , via [15, 16]

$$v(p_z) = v_0 e^{p_z/2\tau_z}, \quad (10)$$

where $\tau_z = 10^4$ m, and v_0 is the nominal vehicle velocity at sea level. In this work, we adopt the 10,000-pound Dragonfly parafoil used by Carter et al. [17], with $v_0 = 17.8$ m/s and lift-to-drag ratio $L_D = 2.8$.

The heading rate of the parafoil is modeled as a second-order approximation of the canopy Dutch roll lateral mode; our specific model selects time constant $\tau = 11.5$ s and damping ratio $\zeta = 0.5$ as suggested by Carter et al. [17]. A first-order lag is used to model the differential toggle control input mechanism with $\tau = 5$ s [18], while the controller is a PID with feedforward gains tuned to achieve comparable performance [17]. In total, this yields a 5th order state \mathbf{s} and dynamics (A, B, C, D) , augmented to the state vector \mathbf{x} and dynamics (1), respectively. The control input is a scalar, $\mathbf{u} \equiv u \equiv \dot{\psi}_d$, representing the desired heading rate, subject to the symmetric input bounds $\mathcal{U} = \{u \mid |u| \leq \omega_{\max}\}$. The overall parafoil dynamics (1) thus take the form

$$\dot{p}_x = v(p_z) \cos \psi + w_x, \quad (11)$$

$$\dot{p}_y = v(p_z) \sin \psi + w_y, \quad (12)$$

$$\dot{p}_z = \frac{-v(p_z)}{L_D} + w_z, \quad (13)$$

$$\dot{\mathbf{s}} = A\mathbf{s} + B\mathbf{u}, \quad (14)$$

$$\dot{\psi} = \text{sat}(C\mathbf{s} + D\mathbf{u}, -\omega_{\max}, \omega_{\max}), \quad (15)$$

where the saturation function $\text{sat}(a, b, c)$ bounds a between b and c , and $\omega_{\max} = \pi/15 = 0.2094$ rad/s [18] (such that the vehicle's minimum turning radius R_{\min} equals $v_0/\omega_{\max} = 85$ m). In this formulation, only the position states \mathbf{p} are affected by the wind disturbance uncertainty \mathbf{w} , including possible effects on altitude p_z by updrafts and downdrafts via w_z . Within the planning framework, which operates in discrete time, the optimization (3) is discretized with time step $dt = 0.1$ s.

III. Real-Time Wind Modeling

The wind model detailed in this section is utilized by the planner to improve prediction accuracy and robustness for the parafoil terminal guidance problem. The development of this wind model is based on satisfying three main objectives. First, the wind model should improve predictability of future wind effects; improved predictability, especially in scenarios where there is significant prevailing wind, can mitigate the amount of replanning needed and improve the quality of solutions provided by the proposed algorithm. Second, the wind model should capture the uncertainty of future wind effects, giving the planner knowledge of a distribution over possible outcomes of a planned trajectory. Characterizing and utilizing such an uncertainty distribution in a probabilistic framework (Section IV) strengthens planner robustness to terrain obstacles. Finally, the wind model should be kept simple, to maintain real-time planner operation and discourage data overfitting.

Given the importance of wind modeling in many engineering applications, there has been considerable work on developing wind prediction and estimation models, including the case of online estimation [19–22]. However, as described in [13], none of these modeling approaches [19–22] address the wind prediction problem over the short timescales and limited datasets inherent in parafoil precision guidance. This section fits an uncertainty model to the wind which can be incorporated into the planner to enforce robustness. This approach includes online learning to determine, in real time, the class of wind scenario being experienced by the parafoil and the corresponding parameters of the variational estimate associated with each class. Each model is tuned to capture the amount of uncertainty typical to wind profiles within its corresponding class. In this manner, the level of conservatism in the planner can be adjusted online to reflect the wind conditions being observed. Draper Laboratories has released 194 altitude-dependent wind profiles from parafoil drops [13], collected using the sensor configuration and estimation procedure outlined in work by Carter et al. [6], which are used as training data in this work.

A. Model Form

The wind model is assumed to take the form (2), written in discrete time as

$$\mathbf{w}_{t+1} = f_w(\mathbf{w}_t, \bar{\mathbf{w}}, \mathbf{v}_t), \quad \mathbf{w}_0 = \mathbf{w}_I, \quad (16)$$

where timestep 0 occurs at system time t_I . The 3-D wind estimate at timestep t , \mathbf{w}_t , is assumed to take the form

$$\mathbf{w}_t = \bar{\mathbf{w}} + \delta\mathbf{w}_t, \quad (17)$$

comprising the sum of a 3-D persistent estimate $\bar{\mathbf{w}}$ and a 2-D variational estimate $\delta\mathbf{w}_t$.

The persistent estimate $\bar{\mathbf{w}}$ reflects the notion that there typically exists a prevailing wind which acts on the parafoil throughout the entire mission, and must be accounted for during the state prediction. It is represented using a finite impulse response filter,

$$\bar{\mathbf{w}} = \frac{1}{m} \sum_{i=t_I-m-1}^{t_I} \mathbf{w}_i, \quad (18)$$

where m is the filter window width.

The filter width m is chosen by optimizing a metric representing the filter predictive accuracy [23]. Consider propagating the parafoil dynamics from some initial state \mathbf{p}_I to the ground, assuming zero input ($u \equiv 0$), no lag dynamics ($\mathbf{s} \equiv 0$), and flat terrain ($T(p_x, p_y) \equiv 0$). Additionally, assume that previous observations of the wind profile have been observed prior to the parafoil reaching \mathbf{p}_I , such that the filter (18) can be applied in full. For each available 3D wind profile, the dynamics are propagated from the same initial state and observations. For the w th wind profile, three possible landing positions are of interest:

- The landing position under the true wind, $\mathbf{p}_T^{(w)}$;
- The landing position under zero wind, $\mathbf{p}_0^{(w)}$; and
- The landing position under constant wind as predicted by (18) with width m , $\mathbf{p}_m^{(w)}$.

Define the quantity

$$\delta d_m^{(w)} = \left\| \mathbf{p}_0^{(w)} - \mathbf{p}_T^{(w)} \right\| - \left\| \mathbf{p}_m^{(w)} - \mathbf{p}_T^{(w)} \right\|, \quad (19)$$

which takes the difference in accuracy between the zero-wind model and the impulse-filter model in predicting the true landing position. For those wind profiles in which prediction accuracy degrades with the impulse-filter model, $\delta d_m^{(w)} < 0$; denote $D_m = \{\delta d_m^{(w)} \mid \delta d_m^{(w)} < 0\}$. The filter width is then

chosen as

$$m = \operatorname{argmax}_{m>0} \{ \min(D_m) + \beta \operatorname{mean}(D_m) - \lambda m \}, \quad (20)$$

where $\beta, \lambda > 0$; in this work, $\beta = 2$ and $\lambda = 1$. This cost function includes terms for the worst-case and average-case accuracy in D_m , as well as a regularization term [23].

The variational estimate $\delta \mathbf{w}_t$ is represented as multi-modal linear dynamics subject to Gaussian noise,

$$\delta \mathbf{w}_{t+1} = (I + dt A_c) \delta \mathbf{w}_t + dt B_c \mathbf{v}_t, \quad c \in \{1, \dots, N_C\}, \quad (21)$$

where N_C is the number of modes/classifications used, A_c and B_c are the tuned matrices used for the c th wind classification (Section III B), and $\mathbf{v}_t \in \mathcal{N}(0, 1)$, *i.e.*, zero-mean, unit-variance Gaussian noise. This colored noise process reflects the idea that, while wind at lower altitudes is correlated with the wind measured at the current altitude, this correlation tends to degrade with increasing separation.

By substituting (17) into (21), the wind model function (16) can be written as

$$f_w(\mathbf{w}_t, \bar{\mathbf{w}}, \mathbf{v}_t) = \bar{\mathbf{w}} + (I + dt A_c)(\mathbf{w}_t - \bar{\mathbf{w}}) + dt B_c \mathbf{v}_t, \quad c \in \{1, \dots, N_C\}. \quad (22)$$

The remaining questions, then, are (i) how to identify an appropriate number of classifications N_C and the corresponding wind model dynamics (A_c, B_c) , $c \in \{1, \dots, N_C\}$ for each, and (ii) how to select the appropriate classification online. These topics are discussed next.

B. Wind Model Training

The wind profiles used for training parameterize each component of the measured wind velocity vector as a function of altitude, *i.e.*, $\{w_x(p_z), w_y(p_z), w_z(p_z)\}$, over a set of altitude data points. This can pose problems for clustering and classification algorithms, which are typically designed to operate on observations, rather than functions. By using feature selection, the dimension of the system model can be reduced, allowing for the use of efficient clustering and classification schemes [24]. For each data point, denote $\rho = \sqrt{w_x^2 + w_y^2 + w_z^2}$ and $\theta = \operatorname{atan2}(w_y, w_x)$; for this

work, we use the feature vector

$$\Phi = \left[\text{mean}(\rho) \quad \text{max}(\rho) \quad \text{mean}\left(\frac{d\rho}{dp_z}\right) \quad \text{max}\left(\frac{d\rho}{dp_z}\right) \quad \text{mean}\left(\frac{d\theta}{dp_z}\right) \quad \text{max}\left(\frac{d\theta}{dp_z}\right) \right]. \quad (23)$$

For each wind profile, this feature vector takes the mean and maximum value over all data points of three quantities: the wind magnitude, the rate of change of wind magnitude, and the rate of change of wind direction. Collectively, these features were chosen to represent the power and variability inherent in each profile.

The objective is then to use the feature-based representation of each wind profile (23), denoted for the w th wind profile as x_w below, to classify the N_W wind profiles into N_C classes. We represent each possible disjoint partition of these profiles as $\mathbf{S} = \{S_1, S_2, \dots, S_{N_C}\}$. The partition is chosen so as to minimize the squared sum of the distances to the mean within each cluster, μ_i , such that

$$\mathbf{S}^* = \underset{\mathbf{S}}{\text{argmin}} \left(\sum_{i=1}^{N_C} \sum_{x_w \in S_i} \|x_w - \mu_i\|^2 \right) + \lambda k, \quad (24)$$

with the last term λk being included for regularization. This optimization is solved using the DP-means algorithm [25], which extends k -means clustering [26] such that the appropriate number of clusters N_C can be incrementally identified, rather than being assumed *a priori*. During the DP-means assignment step, if an observation is further than λ from the nearest cluster center, a new cluster is added with its center defined as the observation x_w which yielded it. Using the aforementioned Draper wind profiles [13], three distinct classes were identified.

For each classification, the variational wind model dynamics (A_c, B_c) are constructed by matching the analytic uncertainty distribution to the empirical distribution identified from the wind profiles. To simplify uncertainty sampling (Section IV), the variational wind model is constructed to be two-dimensional, *i.e.*, $\delta w_z = 0$; observed updrafts and downdrafts are still incorporated via the mean wind $\bar{\mathbf{w}}$. We further assume that δw_x and δw_y are independent and symmetric, such that the variational wind model dynamics $A_c \in \mathbb{R}^{3 \times 3}$ and $B_c \in \mathbb{R}^{3 \times 2}$ can be written as

$$A_c = \alpha_c \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B_c = \beta_c \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad (25)$$

where $\alpha_c, \beta_c \in \mathbb{R}$.

For each wind profile in the cluster, compute the miss distances $\|\mathbf{p}_m^{(w)} - \mathbf{p}_T^{(w)}\|$ as used in (19). Over all wind profiles, let d_i denote the i th largest miss distance, and n_i the fraction of profiles with a miss distance less than or equal to d_i . These characteristics of the cumulative density function (CDF) will be compared against the analytic wind model for tuning, as described next.

Represent the wind model (21) in continuous-time form as

$$\dot{\delta \mathbf{w}} = A_c \delta \mathbf{w} + B_c \mathbf{v}, \quad (26)$$

and define the position variation $\delta \mathbf{p} = \begin{bmatrix} p_x - E[p_x] & p_y - E[p_y] & p_z - E[p_z] \end{bmatrix}^T$. We can then construct the augmented dynamics

$$\begin{bmatrix} \dot{\delta \mathbf{p}} \\ \dot{\delta \mathbf{w}} \end{bmatrix} = \underbrace{\begin{bmatrix} 0_3 & I_3 \\ \mathbf{0}_3 & A_c \end{bmatrix}}_{A_{\text{aug}}} \begin{bmatrix} \delta \mathbf{p} \\ \delta \mathbf{w} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{0}_{3 \times 2} \\ B_c \end{bmatrix}}_{B_{\text{aug}}} \mathbf{v}. \quad (27)$$

The covariance at impact time t_F , $\Sigma_F \equiv \Sigma(t_F)$, can be propagated forward using the dynamics

$$\dot{\Sigma} = A_{\text{aug}} \Sigma + \Sigma A_{\text{aug}}^T + B_{\text{aug}} B_{\text{aug}}^T, \quad \Sigma(t_I) = 0. \quad (28)$$

For comparison with the empirical data, the lateral position covariance is isolated via

$$\Sigma'_F = C_T \Sigma_F C_T^T, \quad (29)$$

$$C_T = \begin{bmatrix} I_2 & 0_{2 \times 4} \end{bmatrix}. \quad (30)$$

Given the independence and symmetry assumptions on δw_x and δw_y , Σ'_F can be written as $\Sigma'_F = \sigma^2 I_2$, where $\sigma > 0$ is a scalar. This represents a chi distribution on landing miss distances with standard deviation σ ; denote its CDF for dynamics (A_c, B_c) as $\chi(x, A_c, B_c)$. This CDF can be matched directly to the empirical wind profile CDF with characteristics (d_i, n_i) as described above. For the c th wind classification, the dynamics (A_c, B_c) are identified by minimizing the root mean square error between the two CDFs,

$$(A_c, B_c) = \underset{(A, B)}{\operatorname{argmin}} \sum_{i \in S_c} (n_i - \chi(d_i, A, B))^2. \quad (31)$$

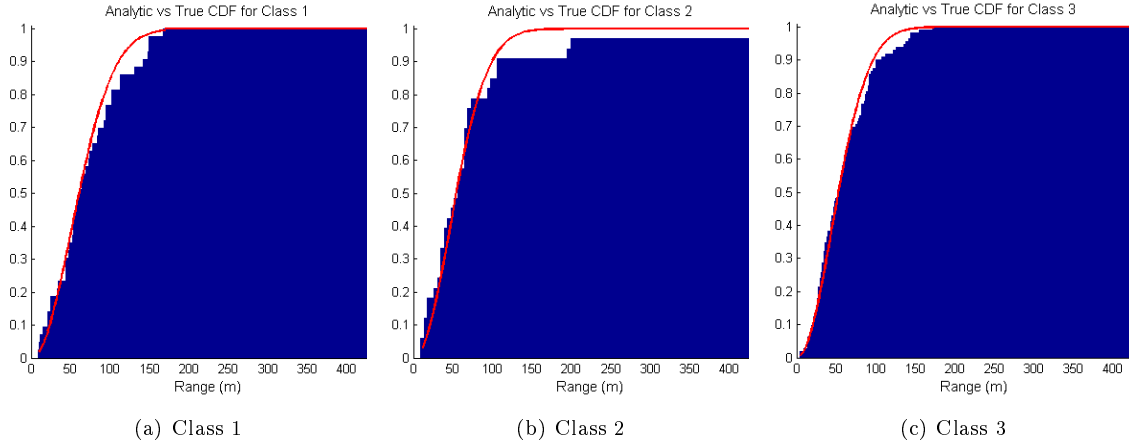


Fig. 1 Analytic (red) and empirical (blue) miss distance CDFs for wind profiles in each class

In our implementation, this is performed in MATLAB via `fminunc`. Figure 1 compares the analytic and “true” (empirical) CDFs for each of the classes identified from the Draper wind profiles.

C. Online Classification Selection

In order to utilize the varying levels of uncertainty associated with the N_C classifications identified in Section IIIB, the planner uses statistical classification to classify each wind estimate that is observed by the vehicle. This is accomplished using support vector machines (SVM) [27]. For each of the N_C classes, the planner generates an SVM binary inclusion classifier, which can be used to identify if the wind estimates being received are a member of a particular class. Online, the trained SVM classifier compares the feature vector produced by the most recent wind observations (23) against the $N_C - 1$ hyperplanes separating the members of the N_C wind classes. Each hyperplane determines in sequence whether the vector Φ does or does not belong to Class X. In the case of Fig. 1, any feature vector which fails the binary inclusion after comparison against the dividing Class 1 and Class 2 hyperplanes is assigned to the remaining Class 3 cluster.

IV. Analytic Uncertainty Sampling and Robust Planning

This section presents a novel framework for modeling future uncertainty in trajectory predictions, based on CC-RRT, such that robustness to possible future variation in disturbances can be achieved. Recall that in the formulation of the parafoil terminal guidance problem (Section II), satisfaction of state constraints is specified via the chance constraint (9). This represents a minimum likelihood that all state constraints, here consisting of the terrain surface $p_z \geq T(p_x, p_y)$, be

satisfied with a minimum probability of p_{safe} along each trajectory. In the CC-RRT algorithm, (9) is converted to a tightened, deterministic constraint at each timestep [11]. Under the assumptions of linear dynamics and Gaussian noise, these tightened constraints are shown to guarantee probabilistic feasibility to polyhedral constraints at each timestep. Further, due to CC-RRT’s trajectory-wise constraint checking, a risk bound can be explicitly computed online against each uncertainty source at each timestep.

In previous work, removing the assumptions of linear dynamics and/or Gaussian dynamics required either linearizing the dynamics at each timestep or performing full time-series simulations of particle-based uncertainty approximations [28]. While such formulations allow the evaluation of path-wise feasibility, they can be computationally intensive, cannot be simulated *a priori* (*i.e.*, independently of the individual RRT trajectories), and can only approximate theoretical guarantees for probabilistic feasibility. It is shown here that, though the parafoil dynamics are nonlinear, the effect of the wind uncertainty is linear-Gaussian. As a result, uncertainty distributions can be derived analytically *a priori* at all future timesteps, and theoretical guarantees maintained – but subject to polyhedral state constraints.

In subsequent developments, we choose to take equi-spaced samples of the uncertainty distributions, such that they can be checked for collision against arbitrary (*i.e.*, not necessarily polyhedral), and possibly aggressive terrain map functions. Though probabilistic guarantees are approximated statistically, uncertainty samples are obtained without dynamic state propagation. As a result, this variant of CC-RRT is more efficient, and better representative of uncertainty distributions, than previous particle-based formulations [28]. Furthermore, the use of sampling allows for path-wise probabilistic feasibility to be quickly evaluated.

A. Analytic Uncertainty Derivation

Consider the parafoil state dynamics (11)-(15), written in discrete-time form as

$$p_{x,t+1} = p_{x,t} + dt [v(p_{z,t}) \cos \psi_t + w_{x,t}], \quad (32)$$

$$p_{y,t+1} = p_{y,t} + dt [v(p_{z,t}) \sin \psi_t + w_{y,t}], \quad (33)$$

$$p_{z,t+1} = p_{z,t} + dt [v(p_{z,t}) (-L_D^{-1}) + w_{z,t}], \quad (34)$$

$$\mathbf{s}_{t+1} = \mathbf{s}_t + dt [A\mathbf{s}_t + B\mathbf{u}_t], \quad (35)$$

$$\psi_{t+1} = \psi_t + dt \cdot \text{sat}(C\mathbf{s}_t + D\mathbf{u}_t, -\omega_{\max}, \omega_{\max}). \quad (36)$$

The final two equations (35)-(36) are unaffected by the uncertainty \mathbf{v}_t , which manifests itself only through \mathbf{w}_t . As in Section IIIB, take the variation $\delta\mathbf{p} = \begin{bmatrix} p_x - E[p_x] & p_y - E[p_y] & p_z - E[p_z] \end{bmatrix}^T$. Recalling that $\delta w_z = 0$ (and thus $p_z = E[p_z]$), combining (32)-(34) with (26) in discrete-time form yields

$$\delta p_{x,t+1} = \delta p_{x,t} + dt(\delta w_{x,t}), \quad (37)$$

$$\delta p_{y,t+1} = \delta p_{y,t} + dt(\delta w_{y,t}), \quad (38)$$

$$\delta p_{z,t+1} = \delta p_{z,t}, \quad (39)$$

$$\delta w_{x,t+1} = (I + dt\alpha_c)\delta w_{x,t} + dt\beta_c v_{x,t}, \quad (40)$$

$$\delta w_{y,t+1} = (I + dt\alpha_c)\delta w_{y,t} + dt\beta_c v_{y,t}; \quad (41)$$

as a result, (39) has decoupled from the other variational dynamics. By defining the 2D variation state vector $\delta\mathbf{x}_t = \begin{bmatrix} \delta p_{x,t} & \delta p_{y,t} & \delta w_{x,t} & \delta w_{y,t} \end{bmatrix}^T$, the variation dynamics (37)-(38), (40)-(41) can be written in the linear form

$$\delta\mathbf{x}_{t+1} = \mathbb{A}\delta\mathbf{x}_t + \mathbb{B}\mathbf{v}_t, \quad (42)$$

$$\mathbb{A} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 + dt\alpha_c & 0 \\ 0 & 0 & 0 & 1 + dt\alpha_c \end{bmatrix}, \quad \mathbb{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ dt\beta_c & 0 \\ 0 & dt\beta_c \end{bmatrix}. \quad (43)$$

Because the linear system (42) is driven by Gaussian noise, all future state distributions take the form $\mathbf{x}_t \in \mathcal{N}(\hat{\mathbf{x}}_t, P_t)$, *i.e.*, Gaussian with mean $\hat{\mathbf{x}}_t$ and covariance $P_t \equiv E[\delta\mathbf{x}_t\delta\mathbf{x}_t^T]$. The mean can be computed using the disturbance-free dynamics (50)-(51), while the covariance can be represented either implicitly as

$$P_{t+1} = \mathbb{A}P_t\mathbb{A}^T + \mathbb{B}\mathbb{B}^T \quad (44)$$

or explicitly as

$$P_t = \mathbb{A}^t P_0 (\mathbb{A}^T)^t + \sum_{k=0}^{t-1} \mathbb{A}^{t-k-1} \mathbb{B} \mathbb{B}^T (\mathbb{A}^T)^{t-k-1}, \quad (45)$$

$$P_0 = \begin{bmatrix} Q_I & 0 \\ 0 & \Sigma_I \end{bmatrix}, \quad (46)$$

where Q_I and Σ_I are the initial covariance for the position and wind, respectively. Either quantity is zero if perfectly known, but may be non-zero if, for example, an estimator is providing data to the system. As in the conventional CC-RRT framework, (44)-(45) can be computed *a priori*, independently of any individual simulated trajectory [11]. Finally, the covariance of the position states can be isolated via the transformation

$$Q_t = C_T P_t C_T^T, \quad C_T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (47)$$

B. Covariance Sampling Generation

In order to efficiently check path-wise feasibility of the terrain constraint $p_z \geq T(p_x, p_y)$, the likelihood of terrain collision is approximated by generating equi-spaced samples at specified levels of the uncertainty distribution at each prospective trajectory node. Sampling the distribution in this way allows for coverage of the uncertainty space with relatively few samples, as well as removing the need to dynamically propagate each sample, significantly reducing computation. The discretization level of the samples N_s , as well as the minimum probability of feasibility p_{safe} , can both be specified by the user to allow for tunable levels of robustness.

The covariance samples are placed at a series of equi-spaced points along uncertainty ellipses. The covariance matrix Q_t describes a contour of equal probability of points $\delta \mathbf{p}_t = \begin{bmatrix} \delta p_{x,t} & \delta p_{y,t} \end{bmatrix}^T$, relative to the nominally propagated trajectory $\hat{\mathbf{x}}_t$, via the conic relaxation $\delta \mathbf{p}_t^T Q_t^{-1} \delta \mathbf{p}_t = 1$. Denote the elements of Q_t as

$$Q_t = \begin{bmatrix} \sigma_{x,t}^2 & \sigma_{xy,t} \\ \sigma_{xy,t} & \sigma_{y,t}^2 \end{bmatrix}, \quad (48)$$

with eigenvalues σ_a^2 and σ_b^2 ($\sigma_a > \sigma_b$). The principle axis of the uncertainty ellipse has angle $\theta_P = \frac{1}{2} \tan^{-1} \left(\frac{2\sigma_{xy,t}}{\sigma_{x,t}^2 - \sigma_{y,t}^2} \right)$. The N_s samples are spaced at equal angular intervals relative to this principle axis; the j th sample $\delta \mathbf{p}_t^{(j)}$ has angle $\theta_j = \frac{2\pi}{N_s-1}j$, relative to θ' . The samples are thus placed at

$$\delta \mathbf{p}_t^{(j)} = \sigma R_Q \mathbb{R}(-\theta') \begin{bmatrix} \cos(\theta_j - \theta') \\ \sin(\theta_j - \theta') \end{bmatrix}, \quad (49)$$

$$R_Q = \frac{\sigma_a \sigma_b}{\sqrt{(\sigma_b \cos(\theta_j - \theta'))^2 + (\sigma_a \sin(\theta_j - \theta'))^2}},$$

where $\sigma > 0$ is the covariance scale factor and $\mathbb{R}(\alpha)$ is the 2D rotation matrix for a counterclockwise rotation of α .

The parameter σ represents the number of standard deviations within the uncertainty ellipse; samples may also be distributed across multiple values of σ . In subsequent results, two rings of covariance samples are used, one at 0.6σ and another at 1.5σ for $\sigma = 1.75$. These values were found to work well empirically.

V. Parafoil CC-RRT Path Planning

This section presents the parafoil CC-RRT algorithm, for robust motion planning using the previously-developed wind model (Section III) and covariance sampling method (Section IV). The core algorithm upon which the parafoil CC-RRT framework builds is RRT, which incrementally constructs a tree of dynamically feasible trajectories from the current state [12]. While many algorithms are available for motion planning problems of this nature [29, 30], an RRT-based approach is particularly well-suited to this application. The lack of controllability in the altitude state p_z limits the utility of graph-based approaches. An RRT can quickly identify and refine feasible solutions online within the 9-dimensional configuration space (3 for position, 1 for heading, 5 for lag dynamics) without discretizing the solution space.

Let the current timestep be t ; the tree is rooted at the current vehicle state, \mathbf{x}_t , and the most recent wind measurement is denoted as \mathbf{w}_t . Each simulated trajectory within the tree uses the nominal dynamics (5) and wind model (7), written as

$$\hat{\mathbf{x}}_{t+k+1|t} = f(\hat{\mathbf{x}}_{t+k|t}, \mathbf{u}_{t+k|t}, \hat{\mathbf{w}}_{t+k|t}), \quad \hat{\mathbf{x}}_{t|t} = \mathbf{x}_t, \quad (50)$$

$$\hat{\mathbf{w}}_{t+k+1|t} = f_w(\hat{\mathbf{w}}_{t+k|t}, \bar{\mathbf{w}}_t, 0), \quad \hat{\mathbf{w}}_{t|t} = \mathbf{w}_t, \quad (51)$$

where the subscript $(\cdot)_{\alpha|\beta}$ denotes simulation timestep α and execution timestep $\beta < \alpha$.

Whereas the nominal RRT algorithm grows a tree of states which are known to be feasible, with any uncertainties assumed to maintain nominal values (here $\mathbf{v} \equiv 0$), CC-RRT grows a tree of state distributions which are checked for feasibility by satisfying an upper bound on the probability of collision at each timestep [11]. The parafoil CC-RRT algorithm similarly generates a tree of uncertainty distributions around trajectories, but further performs analytic sampling via the approach introduced in Section IV A. This allows path-wise probabilistic feasibility checks to be enforced

against arbitrary terrain maps.

As shown in Section IV A, the uncertainty at each prediction timestep $t+k$, relative to execution timestep t , can be represented as a Gaussian state distribution

$$\mathbf{x}_{t+k|t} \sim \mathcal{N}(\hat{\mathbf{x}}_{t+k|t}, P_{t+k|t}). \quad (52)$$

The mean state $\hat{\mathbf{x}}_{t+k|t}$, with position $(\hat{p}_{x,t+k|t}, \hat{p}_{y,t+k|t}, \hat{p}_{z,t+k|t})$, can be simulated using the disturbance-free dynamics (50)-(51), while the covariance $P_{t+k|t}$ can be computed offline via (44). Using (49), the samples are placed at offsets $\delta \mathbf{p}_{t+k|t}^{(j)} = (\delta p_{x,t+k|t}^{(j)}, \delta p_{y,t+k|t}^{(j)}, 0)$, $j \in \{1, \dots, N_s\}$.

Probabilistic feasibility is checked statistically by determining whether the fraction of covariance samples for a given trajectory point $\mathbf{x}_{t+k|t}$ that have intersected with the terrain, at this or any previous simulation step, exceeds $1 - p_{\text{safe}}$. Given the terrain map $T(p_x, p_y)$, the probability of terrain collision p_{collide} at simulation timestep $t+k$ is approximated as

$$p_{\text{collide}} = \frac{1}{N_s} \sum_{j=1}^{N_s} \mathbb{I} \left[\bigwedge_{i=0}^k \hat{p}_{z,t+i|t} \leq T \left(\hat{p}_{x,t+i|t} + \delta p_{x,t+i|t}^{(j)}, \hat{p}_{y,t+i|t} + \delta p_{y,t+i|t}^{(j)} \right) \right], \quad (53)$$

where $\mathbb{I}[\cdot]$ is the indicator function, *i.e.*, 1 if the contained statement is true and 0 otherwise, and \bigwedge represents a conjunction of the indexed constraints. If $p_{\text{collide}} > 1 - p_{\text{safe}}$, then the trajectory is considered to have landed on the terrain.

In addition to the uncertainty-based feasibility check, a trajectory is also considered to have landed if the nominal trajectory intersects the terrain. In other words,

$$\hat{p}_{z,t+k|t} > T(\hat{p}_{x,t+k|t}, \hat{p}_{y,t+k|t}) \quad (54)$$

is added as an additional state constraint.

As with the conventional RRT algorithm, the parafoil CC-RRT algorithm consists of two core components: a “tree growth” step (Luders et al [13], Algorithm 1) that incrementally constructs the tree, and an “execution” step (Luders et al [13], Algorithm 2) that selects paths from the tree for parafoil execution with some frequency. Each time Algorithm 1 is called, a state is first sampled from the environment, and the nodes nearest to this sample in terms of some heuristics are identified as candidates for tree expansion. An attempt is made to form a connection from the nearest node(s) to the sample by generating a probabilistically feasible trajectory between them. Let the

vehicle state and wind state at the nearest node be denoted by $(\hat{\mathbf{x}}_{t+k|t}, \hat{\mathbf{w}}_{t+k|t})$. This trajectory is incrementally simulated by selecting some feasible input $\mathbf{u}_{t+k|t} \in \mathcal{U}$, then simulating the disturbance-free vehicle/wind dynamics via (50)-(51) to yield $(\hat{\mathbf{x}}_{t+k+1|t}, \hat{\mathbf{w}}_{t+k+1|t})$. This input may be selected at the user's discretion, such as through random sampling or a closed-loop controller [31], but should guide the state distribution toward the sample (Section V A). Intermediate nodes may be occasionally inserted during the trajectory generation process, to encourage future expansion.

To check feasibility, the algorithm computes/retrieves the covariance $P_{t+k+1|t}$ at each simulation step using (44). It then computes p_{collide} (initialized to 0) based on the covariance samples that are feasible up to that simulation step, via (53) in order to check whether both $p_{\text{collide}} < 1 - p_{\text{safe}}$ and (54) are satisfied. Trajectory simulation continues until either constraint is violated or the state has reached the sample. Unlike conventional RRT algorithms, every simulated node is added to the tree, even if it does not reach its target sample before intersecting the terrain. As each node is added to the tree, an attempt is made to connect it directly to the goal state \mathbf{x}_G .

After completing the "tree growth" step of the current iteration, the "execution" step (Luders et al [13], Algorithm 2) executes some portion of the tree while continuing to grow and update the tree in subsequent iterations. The execution step updates the current best path to be executed by the system every Δt seconds (here, $\Delta t = 1$ s).

The execution step first retrieves the current vehicle state, wind measurement, and mean wind estimate, then updates the tree via re-propagation. In this update, all nodes are re-checked for probabilistic feasibility and any nodes which have become infeasible are pruned, along with their descendants. Additionally, costs are updated for each node using the cost function (3), written in discrete form as

$$J(N_{\text{target}}) = \phi_F(\mathbf{x}_{t_F|t+\Delta t}, \mathbf{x}_G) + \Delta t \sum_{i=t+\Delta t}^{t_F} \phi(\mathbf{x}_{i|t+\Delta t}, \mathbf{x}_G). \quad (55)$$

After these updates, the tree is repeatedly expanded using Algorithm 1 for the duration of the timestep. Following this tree growth, (55) is used to select the lowest-cost path in the tree for execution. If no path exists in the tree, some "safe" motion primitive can be applied to attempt to keep the vehicle in a safe state.

As described above, we have chosen to re-propagate the entire tree for both feasibility and cost, rather than simply re-check the feasibility of the lowest-cost path via "lazy check" [31] without

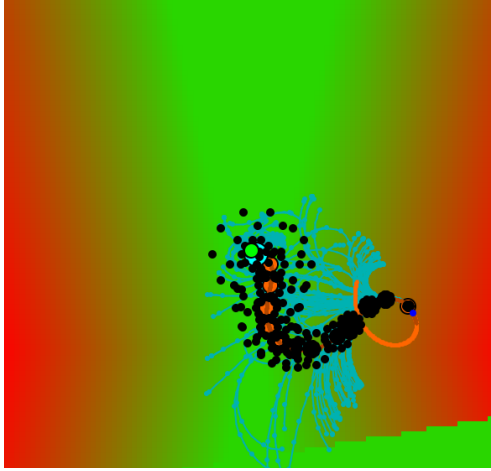


Fig. 2 Parafoil CC-RRT simulation in progress, showing covariance samples (black)

updating costs. In the context of the parafoil terminal guidance problem, where feasibility and cost are inextricably linked and highly dynamic as a function of the uncertainty, it is useful to update all possible trajectories in order to achieve reliable performance. While additional computation is required to perform this update, in practice this computation is balanced over time by the tree size, via the amount of time spent in Algorithm 1. Section VII demonstrates the effectiveness of the analytic CC-RRT algorithm in improving worst-case planning for parafoil terminal guidance, particularly subject to complex terrain and pathological wind conditions.

Figure 2 shows an example of a parafoil CC-RRT tree generated for the valley terrain (Section VII A) for $\sigma = 1.5$. The planner constructs a tree of feasible trajectories (teal), which guides the parafoil (blue) to land on the terrain at the goal (green circle). In Fig. 2, the terrain background changes from green to red with increasing altitude, and the covariance samples (black) are shown for the path currently selected for execution (orange).

A. Reference Model

A reference model is used in the parafoil CC-RRT algorithm to generate a sequence of inputs $u = \dot{\psi}_d$ for each trajectory. Each trajectory connects a nearest node, with position \mathbf{p}_n and heading ψ_n , to a sample with position \mathbf{p}_s . Since the altitude state p_z is uncontrollable, the proposed reference model generates the 2D circular arc which connects the nearest node and sample and is tangent to the heading at the nearest node (Fig. 3). Such a circular arc can be followed by the ideal (*i.e.*, no lag dynamics) parafoil in 2D, by applying a constant input \bar{u} for some time duration \bar{t} . Resulting trajectories are thus sequences of piecewise-constant-angular-rate segments. The number

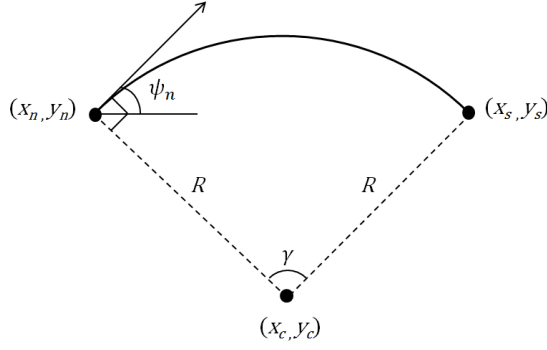


Fig. 3 Reference generation model using 2D circular arcs

of segments and the duration of each segment are not fixed, such that complex trajectories can still be specified within this reference model. Defining $\delta_y = y_s - y_n$, $\delta_x = x_s - x_n$ and $\delta = \sqrt{\delta_x^2 + \delta_y^2}$, the radius of this circle is

$$R = \frac{\delta^2}{2(\delta_y \cos \psi_n - \delta_x \sin \psi_n)}, \quad (56)$$

where the sign of R in (56) encodes the turn direction. This is used with the velocity model (10) to yield the desired angular rate, $\bar{u} = v(p_{zn})/R$. The duration of the reference command is determined by computing the arc subtended angle γ :

$$\gamma = 2 \sin^{-1} \left(\frac{\delta}{|2R|} \right), \quad (57)$$

$$\bar{t} = \gamma / \bar{u}. \quad (58)$$

Alternative reference generation models could be applied, such as Bezier curves [10], B-splines, or piecewise linear (rather than piecewise constant) angular rate commands. Closed-loop propagation may also be used to limit uncertainty growth over time.

VI. Reachability-based Cost-to-go

One of the advantages of using RRTs, due to their incremental construction, is the capability to select a path which has not yet terminated in planning, use it as the basis for vehicle execution, and complete the path during later planning cycles. Critical to this capability is an informative cost-to-go function for assessing path quality. In previous work, Luders et al. [13] proposed a cost-to-go formulation which combines the cost at a point of interest – the end of a trajectory within the tree – with a discrete approximation of the reachability set beyond that point. In this manner, the

cost-to-go function weighs the value of the system's present state against possible near-term future states, which are heavily constrained by the present state and particularly the parafoil heading [13].

This work extends the previous cost-to-go formulation with the option to include an additional penalty term on the parafoil ground speed. Because the planner can only affect its velocity by changing its heading relative to the wind direction, such a penalty term encourages the selection of paths with upwind landings to reduce landing speed.

The cost-to-go function is constructed by assigning costs J_i to each of a set of states \mathbf{x}_i , $i \in \{0, \dots, N_P\}$, where \mathbf{x}_0 is the vehicle's current state and \mathbf{x}_i , $i \in \{1, \dots, N_P\}$ are drawn from the boundary of a finite-time reachability set for the parafoil [13]. The cost J_i used is the Euclidean distance from the point \mathbf{x}_i to the goal \mathbf{x}_G at position $\mathbf{p}_G = (p_{xG}, p_{yG}, p_{zG})$, after accounting for drift due to the persistent wind estimate $\bar{\mathbf{w}} = (\bar{w}_x, \bar{w}_y, \bar{w}_z)$. This takes the form

$$J_i = \sqrt{(p_{xi} - p_{zG} - t_{zG}\bar{w}_x)^2 + (p_{yi} - p_{yG} - t_{zG}\bar{w}_y)^2 + (p_{zi} - p_{zG})^2}, \quad (59)$$

where $t_{zG} = (p_z - p_{zG})/v_z$ is the time to reach the goal altitude from the current state.

The full cost-to-go function takes the form

$$\phi_F(\mathbf{x}_0, \mathbf{x}_G) = C_v \max(J_0, \min(J_1, J_2, \dots, J_{N_P})) + J_v(\mathbf{x}_0, \bar{\mathbf{w}}), \quad (60)$$

where $C_v > 0$. The first term of (60) takes the maximum between the cost of the initial point, J_0 , and the minimum of the cost of points on the reachability set approximation. The J_0 portion encourages the planner to situate the vehicle directly above the goal (after correcting for wind) to facilitate disturbance rejection, while the $\min(J_1, J_2, \dots, J_{N_P})$ portion represents the most favorable state the vehicle can reach within the (approximated) finite-time reachability set [13].

The second term of (60) directs the desired heading at landing ψ_G toward the upwind direction by penalizing the node's terminal ground speed. Since the ground speed is minimized when the parafoil heading and wind direction are opposing, and maximized when they align, this induces the desired behavior. This cost term is defined as

$$J_v(\mathbf{x}_0, \bar{\mathbf{w}}) = \sqrt{(v(p_{z0}) \cos \psi_0 + \bar{w}_x)^2 + (v(p_{z0}) \sin \psi_0 + \bar{w}_y)^2}, \quad (61)$$

incorporating both the parafoil air speed and estimated wind effect. The relative importance of

landing accuracy vs. landing speed is controlled by the weighting coefficient C_v ; in this work, a weight of $C_v = 10$ is used.

VII. Simulation Results

This section presents simulation results demonstrating that parafoil CC-RRT, also referred to here as “analytic CC-RRT,” achieves superior landing accuracy to both nominal RRT and BLG [6] on challenging terrain. In particular, parafoil CC-RRT demonstrates significant improvement in mean accuracy over BLG, and superior worst-case landing accuracy to both RRT and BLG. Adding a penalty on landing speed to the cost-to-go (Section VI) reduces landing accuracy slightly, but significantly reduces typical landing speeds by better orienting the parafoil upwind. Further analysis shows that while BLG performance tends to degrade as the difficulty of the terrain increases, parafoil CC-RRT is largely invariant to changes in terrain, in both the mean and worst-case. Parafoil CC-RRT is also shown to be not only capable of use at higher initial drop altitudes, but also invariant to initial drop altitude. Additional simulations, found in [13], show that the multi-classification wind model (Section III) and max-min cost-to-go formulation (Section VI) yield comparable or better accuracy than their individual components.

A. Implementation

Three algorithms are compared throughout this section:

- 1) **RRT with mean wind**, which represents a nominal RRT planner using the mean wind estimate $\bar{\mathbf{w}}$, but assumes no future wind variation (*i.e.*, $\delta \mathbf{w} \equiv 0$, $\mathbf{w}_t \equiv \bar{\mathbf{w}}$). This approach makes no active attempt at robustness against uncertainty, but does utilize replanning at every timestep to try to counteract system disturbances.
- 2) **Analytic CC-RRT**, the full parafoil CC-RRT algorithm presented throughout this paper and specified in Algorithm 1 and Algorithm 2 [13].
- 3) **BLG**, or band-limited guidance, which utilizes band-limited control to ensure accurate tracking and prediction, as well as knowledge of the mean wind estimate $\bar{\mathbf{w}}$ and replanning to account for system disturbances [6]. (The implementation of BLG is detailed below.)

For each scenario, each algorithm is tested over a large series of simulation trials, which vary in the combination of wind profile and initial conditions used. Of the wind profiles released by Draper

Laboratories [13], a set of 25 representative wind profiles are used. These 25 wind profiles consist of 18 profiles from collected drop data and 7 artificially-generated profiles. Of the 7 artificially-generated profiles, 6 are constant-wind profiles moving in the cardinal directions, varying in intensity from zero wind to 25 knots (over 70% of the parafoil airspeed). The final artificially-generated profile represents an exponentially-decaying wind, with average and maximum wind speed changes (with respect to altitude) of $0.0025 \frac{\text{m/s}}{\text{m}}$ and $0.05 \frac{\text{m/s}}{\text{m}}$, respectively. The actual drop wind profiles are significantly more aggressive, with an average overall intensity of 6.7 m/s and gusts up to 17.1 m/s (nearly matching the parafoil airspeed). These profiles are subject to average and maximum wind speed changes (with respect to altitude) of $0.025 \frac{\text{m/s}}{\text{m}}$ and $2.4 \frac{\text{m/s}}{\text{m}}$, respectively. They are also subject to rapid directional changes, potentially as large as $115^\circ/\text{m}$.

In each trial, the parafoil state is initialized 500 meters above the goal (assumed in all scenarios to be located at $\mathbf{p}_G = (0, 0, 0)$) with a random heading and a lateral distance between 100 and 400 meters from the goal. Each algorithm is subject to the same sequence of wind profile/initial condition combinations. A total of 500 trials are performed for each algorithm in each scenario, representing 20 uses of each wind profile for different initial conditions.

The primary terrain used in the simulations is the $1.5 \text{ km} \times 1.5 \text{ km}$ valley terrain, $T_{\text{valley}}(p_x, p_y)$, pictured in Fig. 4. The green shades in Fig. 4 indicate areas of lower altitude, while the goal is located at the yellow diamond. This represents a particularly challenging terrain for the parafoil terminal guidance problem, for several reasons. First, the slope of the valley is greater than the glide-slope of the parafoil, limiting planning options at lower altitudes by making approach from either side impossible. Second, the large low-altitude regions away from the goal (bottom-right and top in Fig. 4(a)), where terrain collisions can be avoided for longer path durations, are likely to lead to terrain interactions as the parafoil’s path crosses in and out of those regions. Finally, placing the goal near a terrain “bottleneck,” makes planning near the goal more difficult than planning away from the goal. To test how algorithmic performance varies with terrain “difficulty,” this section also considers scaled-down versions of the valley terrain, *i.e.*, $\alpha T_{\text{valley}}(p_x, p_y)$ for $\alpha \in [0, 1]$. In particular, simulations are performed for $\alpha = 0$, representing completely flat terrain, and $\alpha = 0.75$, representing intermediate conditions.

The CC-RRT algorithm has been implemented as a single-threaded Java application. To simplify comparisons, a fixed number of samples, or iterations of Algorithm 1, are performed per loop of Algorithm 2. In the subsequent results, 165 samples per loop are used, representing the average

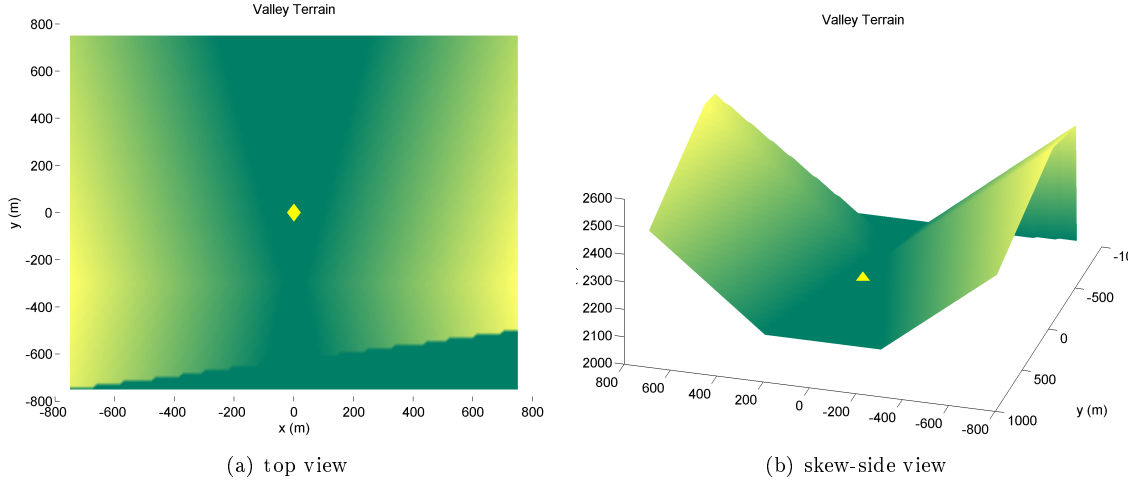


Fig. 4 Valley terrain used in several of the parafoil terminal guidance scenarios

number of samples generated in a 1 Hz planning cycle with 60% duty cycle by the nominal RRT algorithm. The mean wind impulse filter (Section III) has a width $m = 10$, while two rings of 10 covariance samples each are used with an overall $p_{\text{safe}} = 0.9$ (Section IV).

The BLG algorithm, against which parafoil CC-RRT is compared, determines an optimal control by choosing coefficients ψ_k for the heading rate profile,

$$\psi'(h) = \sum_{k=0}^N \psi_k \frac{\sin(\pi(z - k\Delta z)/\Delta h)}{\pi(z - k\Delta z)/\Delta h}, \quad (62)$$

based on simulating forward simplified dynamics,

$$\begin{aligned} x' &= -L_D \cos(\psi) + w_x/\dot{z}, \quad \text{and} \quad y' = -L_D \sin(\psi) + w_y/\dot{z}, \\ (\cos(\psi))' &= -\psi(z)' \sin(\psi), \quad \text{and} \quad (\sin(\psi))' = \psi(z)' \cos(\psi), \end{aligned} \quad (63)$$

where $(\cdot)'$ denotes a derivative with respect to altitude z [6]. It then performs an optimization on a cost function consisting of a weighted sum of x^2 , y^2 , and $(\sin(\Delta\psi/2))^2$, where $\Delta\psi$ is the difference between final heading and desired heading.

This vehicle model is fundamentally different from the one used by analytic CC-RRT (Section III A) in the way heading rate is handled. Whereas CC-RRT assumes heading rate is the output of a linear lag-dynamics model, the BLG vehicle model assumes lag-free control over the heading rate, provided that the controls are bounded by (62). Additionally, the BLG algorithm is constrained to enforce a landing direction, chosen to be the upwind direction at the moment planning begins.

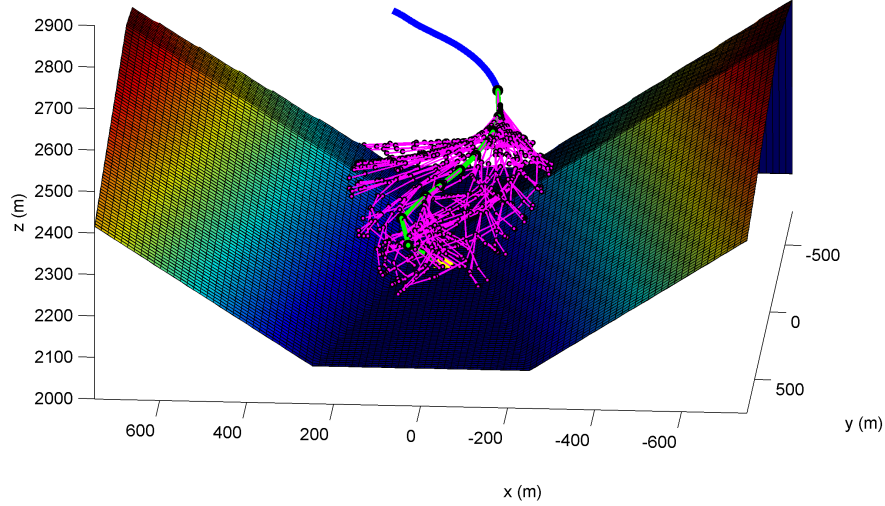


Fig. 5 Screenshot of Analytic CC-RRT simulation on the valley terrain

The BLG algorithm has been implemented in MATLAB for comparison. In this work, the optimization is performed using iterations of MATLAB’s `fmincon` function, instead of Nelder-Mead simplex optimization as in Ref. [6]. The parameter values $\delta h = 200$ m and $N = 4$ used in Ref. [6] are used in this implementation, as well. For comparison with CC-RRT, rather than using a tolerance-based stopping criterion with a maximum 100 iterations [6], BLG is permitted to simulate the parafoil to the ground through 75 iterations per planning cycle. This number of iterations requires a comparable amount of computation to the number of RRT samples per planning cycle, as described above.

B. Valley Terrain Simulations

First, 500 trials were run for each algorithm on the valley terrain scenario (Fig. 4). Figure 5 provides a simulation example of analytic CC-RRT’s online execution in the valley terrain environment, where the parafoil trajectory (blue) and tree of feasible trajectories (magenta) are shown, including the current best path (green) to the goal location (yellow circle). Figure 6 and Table 2 show the CDF and statistics, respectively, over all trials. The results indicate that analytic CC-RRT demonstrates matching or improved landing accuracy, relative to RRT with mean wind and BLG, at nearly all percentiles.

Both RRT-based algorithms show significant improvement over BLG for all but the worst-case trials. The mean landing accuracy for both is lower than BLG by a factor of 2. This ratio continues approximately up to the 95th percentile, and increases to a factor of 3–4 by the 98th percentile

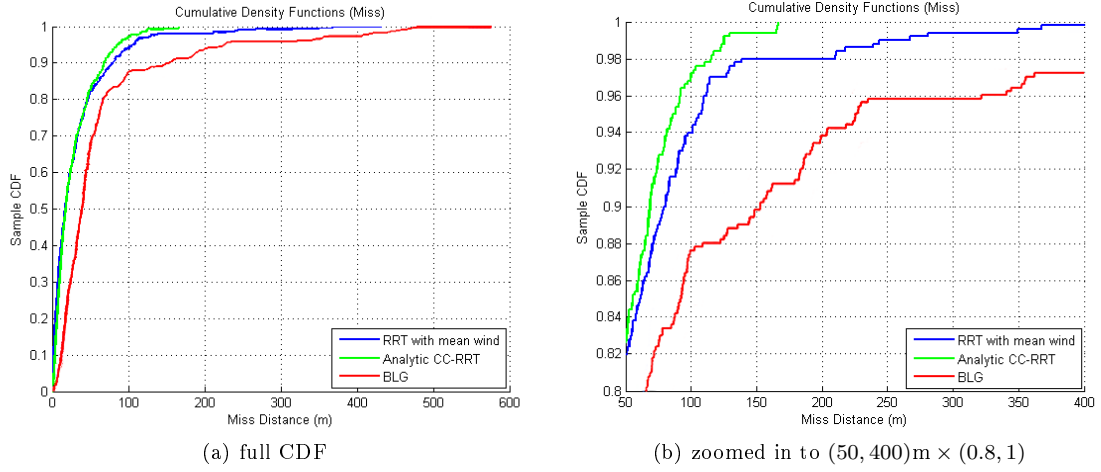


Fig. 6 Miss distance CDF for valley terrain comparison, over 500 trials

Table 2 Miss distance data for valley terrain comparison, over 500 trials (in meters)

Algorithm	Mean	StDev	50%	80%	90%	95%	98%	Max
RRT with mean wind	32.3	47.0	17.0	46.9	79.6	106	139	431
Analytic CC-RRT	28.2	28.5	18.0	45.7	68.6	86.5	115	167
BLG	63.5	89.0	37.9	66.1	153	227	431	581

(Table 2). In particular, about 12% of BLG trials have a miss distance exceeding 100 m, whereas less than 5% of the analytic CC-RRT trials, and less than 6% of the RRT trials, have a miss distance exceeding 100 m (Fig. 6(b)). The BLG algorithm also demonstrates a “long tail”: 4% of trials have a miss distance of 300 m or worse, while the worst-case trial misses by 581 m.

Landing accuracy is comparable between RRT with mean wind and analytic CC-RRT up to the 80th percentile; however, analytic CC-RRT demonstrates superior performance over both RRT and BLG in the worst 20% of trials. This suggests that for those trials in which terrain interaction is unlikely, the robustness-based enhancements in analytic CC-RRT do not significantly influence performance relative to RRT with mean wind prediction alone. However, the two CDF curves diverge near the 82nd percentile (Fig. 6(b)). At the 95th and 98th percentiles, analytic CC-RRT miss distance is 17-18% lower than RRT. By the worst-case trial (*i.e.*, 99.8%), analytic CC-RRT miss distance is 61% lower. All trials of analytic CC-RRT have an accuracy under 170 m, whereas RRT demonstrates some trials exceeding 430 m (Table 2). This “shorter tail” for the analytic CC-RRT distribution, relative to RRT with mean wind (Fig. 6), demonstrates the robustness of the algorithm to pathological uncertainty conditions, which might otherwise drive the vehicle prematurely into the terrain.

Both RRT with mean wind and BLG encounter trials where landing accuracy exceeds 400 m – with BLG sometimes encountering accuracies approaching 600 m. Such situations are the product of interaction between the uncertain wind and the difficult terrain encountered by the parafoil. Figures 7 and 8 demonstrate how changing wind conditions can cause selected/executed paths from RRT and BLG, respectively, to become infeasible despite replanning.

Figure 7 shows the planned paths (green) for nominal RRT on successive timesteps. On the first timestep (Fig. 7(a)), the RRT planner has identified a semi-circular path which brings the parafoil relatively close to the goal (yellow circle). However, after a new wind measurement, this trajectory is now predicted to collide with the terrain only about halfway through this path (Fig. 7(b)). This causes the second half of the path to be pruned, leaving the parafoil on a trajectory which now has poor terminal accuracy. The issue, in this case, is that several of the intermediate path nodes are very close to the terrain, such that a wind shift causes them to become infeasible.

Figure 8 compares a planned trajectory (solid red line) and executed trajectory (dashed blue line) for a parafoil using a BLG planner. The planned, nominal trajectory (based on the mean wind estimate) takes the parafoil very close to the goal, but also comes very close to the terrain surface on the right side of the valley before turning back toward the goal (Fig. 8(a)). About one-quarter of the way through execution, a small wind shift takes place, resulting in a deviation between prediction and execution (Fig. 8(b)) that yields a mismatch of less than 1 m (yellow line). Yet this mismatch is sufficient to cause the parafoil to collide with the terrain (blue star), resulting in a miss distance exceeding 450 m. The direct optimization technique of BLG does not consider off-nominal, future terrain interactions caused by changing wind conditions. As a result, such adverse terrain interactions are possible in the worst case. In both cases (Fig. 7 and 8), such proximity to this sloping terrain would be captured by the analytic samples (Section IV) using the CC-RRT formulation, such that the original path with low robustness margin would not have been chosen for execution.

Table 3 shows the average computation time per node generated for both RRT with mean wind (*i.e.*, 0 samples) and analytic CC-RRT, for various numbers of covariance samples. The time per node for analytic CC-RRT scales favorably both with the number of covariance samples and relative to RRT alone. Table 4 demonstrates that the desirable properties of the RRT-based algorithms are largely preserved when each algorithm is run in real-time, rather than using a fixed number of iterations. In these tests, 500 trials are again performed. However, rather than running a fixed

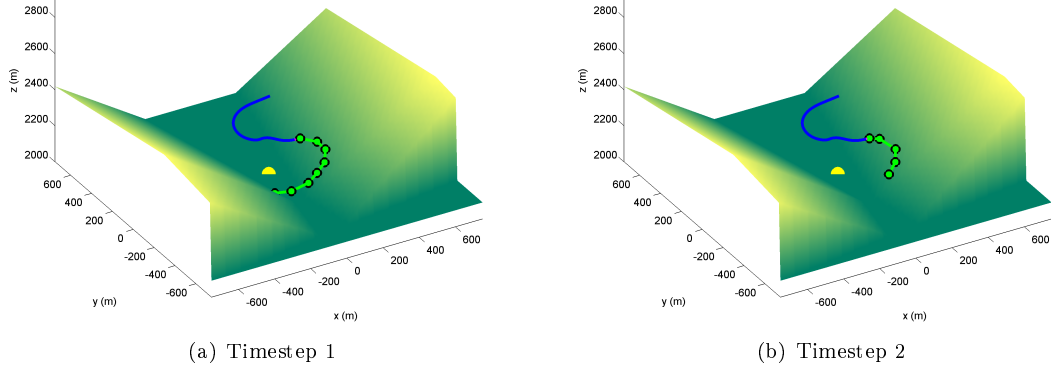


Fig. 7 Trajectory planned using RRT with mean wind during the first two timesteps

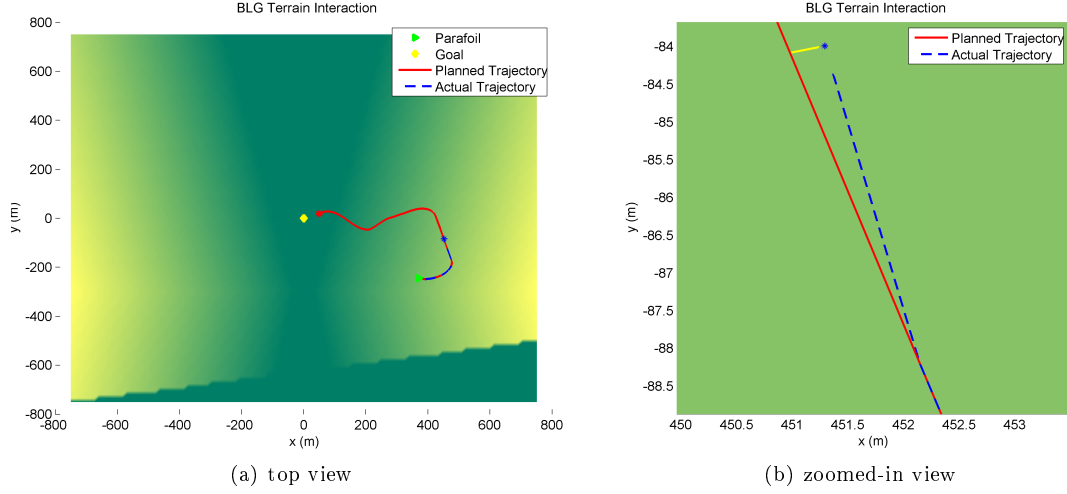


Fig. 8 Trajectory planned using BLG (red) vs. executed trajectory (blue)

number of iterations, the "execution" step (Algorithm 2) is run on a $\Delta t = 1\text{s}$ or $\Delta t = 2\text{s}$ update cycle; during each cycle, a 60% duty cycle (*i.e.*, 0.6 s of each 1 s) is available for the planner to update and grow the tree.

Compared to Table 2, the performance of RRT with mean wind is similar for both sets of real-time results. Additionally, the extra computation available by doubling the planning time does not appreciably improve performance. Because of the additional computation needed to check covariance samples, the performance of analytic CC-RRT degrades slightly when subject to real time planning constraints. Regardless, it still yields the best performance in the worst-case relative to RRT with mean wind alone.

Table 3 Average node generation times for nominal RRT and analytic CC-RRT

# samples	Time per Node (ms)
0 (RRT)	10.4
10	17.00
20	26.24
30	33.19
50	46.00

Table 4 Miss distance comparison in real time (60% duty cycle), over 500 trials (in meters)

Algorithm	Δt (s)	Mean	StDev	50%	80%	90%	95%	98%	Max
RRT with mean wind	1	30.6	39.6	18.5	48.3	76.4	94.6	135	353
	2	29.1	37.1	17.7	45.6	68.5	90.0	113	417
Analytic CC-RRT	1	42.1	38.1	30.4	65.7	91.7	115	151	296
	2	33.1	32.7	22.4	50.5	80.9	92.0	122	237

C. Landing Speed Penalty

Table 5 compares results for analytic CC-RRT in the valley terrain over a set of 500 trials, in which the landing speed penalty of Section VI is applied ($C_v = 10$). Results are compared in terms of both their miss distance and the ground speed of the parafoil at landing, noting that the nominal speed of the parafoil in the absence of wind is 19.79 m/s.

Without the landing speed penalty, more than 50% of trials land at the nominal speed or higher. In the worst case, the ground speed is increased by 36% relative to the air speed, leading to potentially precarious landing conditions. With the landing speed penalty turned on, however, the mean wind speed is reduced by 17%, well below nominal speed. Less than 10% of trials yield a landing speed higher than the nominal speed, implying that more than 90% of trials are oriented within 90 degrees of the upwind direction. In the worst case, the landing speed is only 14% larger than nominal. Additionally, while the miss distances are increased due to incorporation of the penalty, the increase is not large. The mean miss distance increases by less than 20%, while the worst-case miss distance remains under 300m.

Figure 9 gives polar scatter plots representing the spread of landing orientations over each set of 500 trials in Table 5. In these images, the heading of each data point corresponds to the parafoil’s

Table 5 Miss distance and landing speed data for CC-RRT on valley terrain over 500 trials

Metric	Speed penalty?	Mean	StDev	50%	80%	90%	95%	98%	Max
Miss distance (m)	No	28.2	28.5	18.0	45.7	68.6	86.5	115	167
	Yes	33.6	32.2	24.9	54.1	71.9	92.2	119	265
Landing speed (m/s)	No	19.2	3.8	19.8	21.6	23.3	25.3	28.3	30.0
	Yes	16.0	3.7	16.6	19.8	19.8	20.1	21.8	23.1

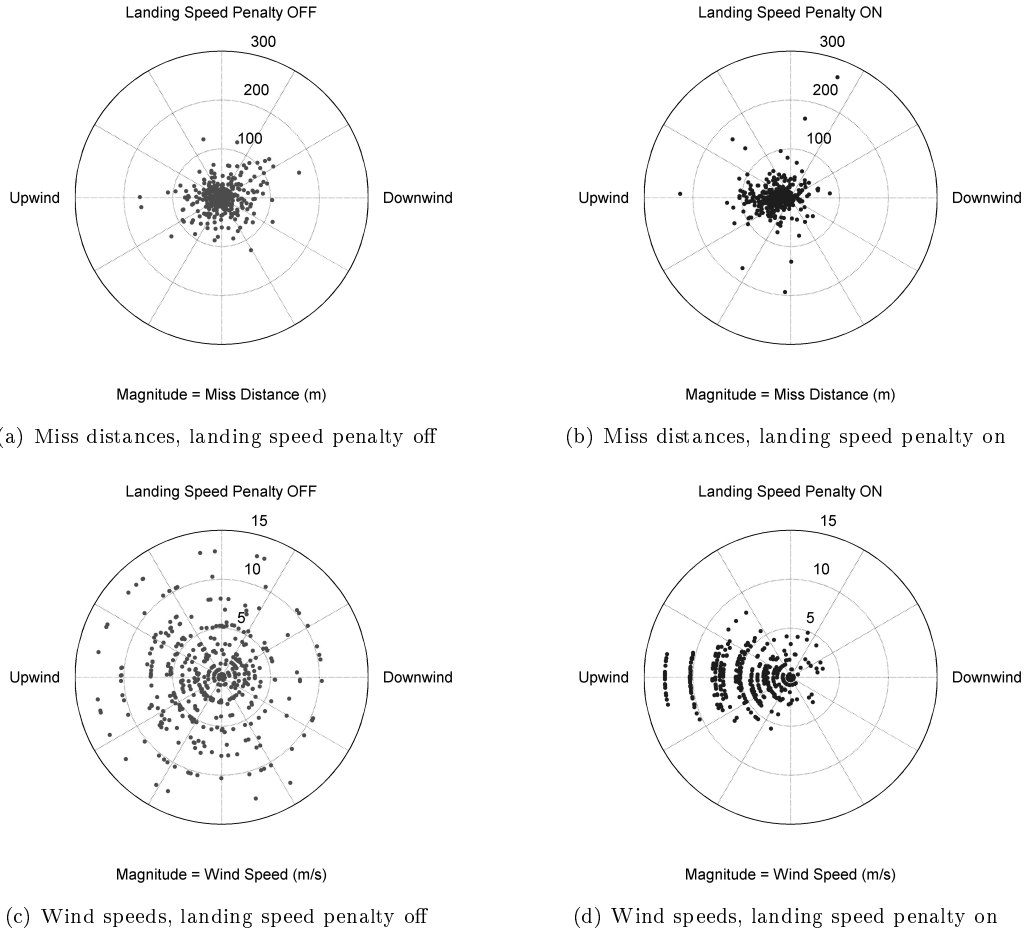


Fig. 9 Radial scatter plots of each trial of Analytic CC-RRT performed on the valley terrain

heading at landing relative to the wind direction in that trial; the most desirable heading is upwind (left side of each figure), while the least desirable heading is downwind (right side of each figure). With the landing speed penalty off (Fig. 9(a) and 9(c)), the 500 trials are widely distributed across all landing orientations, with only a very slight bias toward the upwind direction. In particular, in cases where the terminal wind exceeds 10 m/s, many trials can be seen to result in a downwind landing, significantly increasing landing speed (Fig. 9(c)).

In contrast, when the landing speed penalty is enabled, trials are heavily biased toward landing in the upwind direction. Most trials land within 30 degrees of the upwind direction, and no trials with a terminal wind exceeding 5 m/s land downwind (Fig. 9(d)). However, some outliers are introduced in terms of miss distance (Fig. 9(b)).

D. CC-RRT Invariance to Terrain

Figure 10 and Table 6 show the CDF and statistics, respectively, for 500 trials performed on a completely flat terrain, *i.e.*, $T(p_x, p_y) \equiv 0$. Although this terrain proves less complex for the planner, the parafoil’s underactuated dynamics still yield a challenging planning problem. In Fig. 10, the miss distance accuracy of RRT with mean wind and analytic CC-RRT have converged to approximately the same CDF, with analytic CC-RRT showing only a slight decrease in miss distance relative to RRT with mean wind at most percentiles, including the worst-case. This is consistent with using obstacle-free terrain: because the covariance sampling is performed in the 2D horizontal plane relative to the prospective trajectories, said samples will either all be feasible ($p_z > 0$) or all be infeasible ($p_z \leq 0$) at any given timestep, and thus provide no new information to the planner. As a result, analytic CC-RRT functions identically to RRT with mean wind in this scenario.

On the other hand, the BLG algorithm demonstrates significant improvements in accuracy, relative to both the RRT-based algorithms and its own performance on the valley terrain. Compared to the valley terrain, the mean miss distance has decreased by 75%, while the worst-case miss distance has decreased by over 80% (from nearly 600m to just over 100m). BLG also demonstrates up to 40% better accuracy than RRT with mean wind and analytic CC-RRT, in both mean and worst-case miss distance (Table 6). The improvement is most noticeable between the 50th and 90th percentiles (Fig. 10). Because of the absence of terrain features, the BLG algorithm is no longer susceptible to off-nominal terrain interactions in this scenario, creating the ideal environment for the algorithm to converge on optimal solutions. Here, finding feasible solutions (the primary strength of RRT-based algorithms) is a relatively simple task compared to more complex terrain environments, such that optimizing the planned trajectory (a task BLG is more effective at) is a more efficient use of available computational resources. Replanning alone, without robustness modifications, is sufficient to counteract the shifting wind conditions.

Based on this analysis, we consider how the performance of both BLG and analytic CC-RRT varies as the “difficulty” of the terrain is changed. As stated in Section VII A, this is done by considering scalings of the valley terrain, $\alpha T_{\text{valley}}(p_x, p_y)$. In addition to the cases of $\alpha = 1$ and $\alpha = 0$ already considered, we also consider “75% Valley Terrain,” in which $\alpha = 0.75$, representing a terrain of intermediate difficulty. Figures 11(a) and 11(b) show the CDFs over 500 trials across all three terrains for BLG and analytic CC-RRT, respectively, with tabular data provided in Tables 7 and 8, respectively.

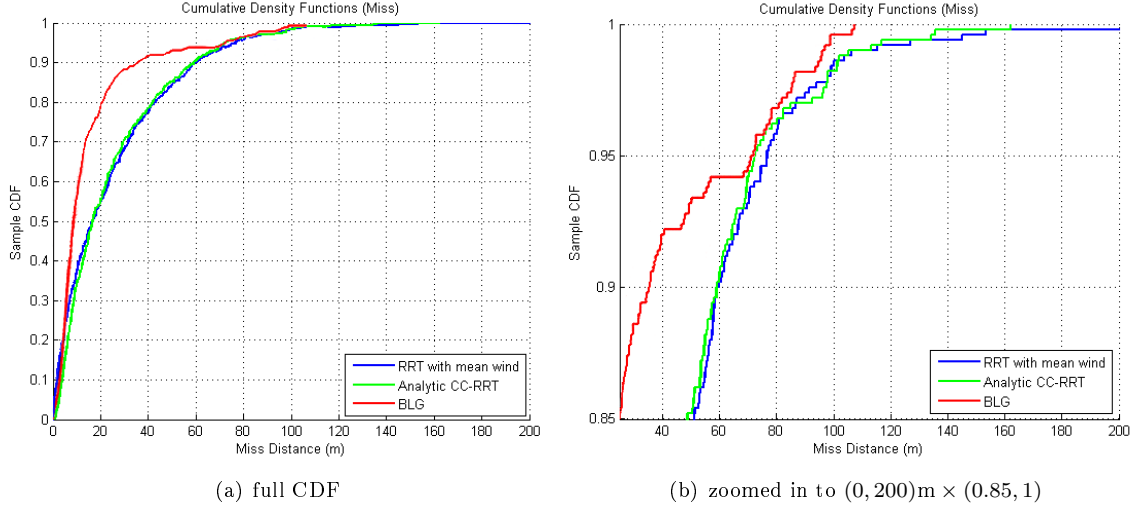


Fig. 10 Miss distance CDF for flat terrain comparison, over 500 trials

Table 6 Miss distance data for flat terrain comparison, over 500 trials (in meters)

Algorithm	Mean	StDev	50%	80%	90%	95%	98%	Max
RRT with mean wind	25.2	26.6	16.5	42.5	59.2	76.7	98.0	220
Analytic CC-RRT	25.3	24.5	16.5	41.8	58.9	72.2	97.8	162
BLG	15.9	19.7	8.9	20.7	35.2	71.3	86.1	107

Based on Fig. 11(a) and Table 7, it is clear that the BLG algorithm is highly sensitive to the complexity and steepness of the terrain. As the terrain becomes more complex, feasible paths become more difficult to find, and thus cannot be optimized to the same extent. On the 75% valley terrain, there exists a regime of nominal performance, up to around the 80th percentile, where BLG performance matches or even exceeds performance on flat terrain, with a miss distance under 40 m (Fig. 11(a)). In these cases, finding a feasible solution is relatively straightforward, and BLG is able to spend significant time optimizing the solution. For the remaining approximately 100 trials, however, terrain interactions are a serious concern, and BLG miss distance begins to increase significantly relative to the flat terrain scenario (Fig. 11(a)). At the 98th percentile, both the miss distance and worst-case miss distance on 75% valley terrain are more than twice their flat-terrain counterparts. Once the terrain scaling is increased to the full valley terrain (Section VIIB), the possibility of terrain interactions becomes much more significant, and miss distances increase at all percentiles, especially the worst-case trials.

On the other hand, the performance of the analytic CC-RRT algorithm (Fig. 11(b) and Table 8) is largely invariant to the terrain scaling considered. The gap in mean performance between all terrains considered is only 4 m or about 12%, while the gap in the 98th-percentile is only 17 m

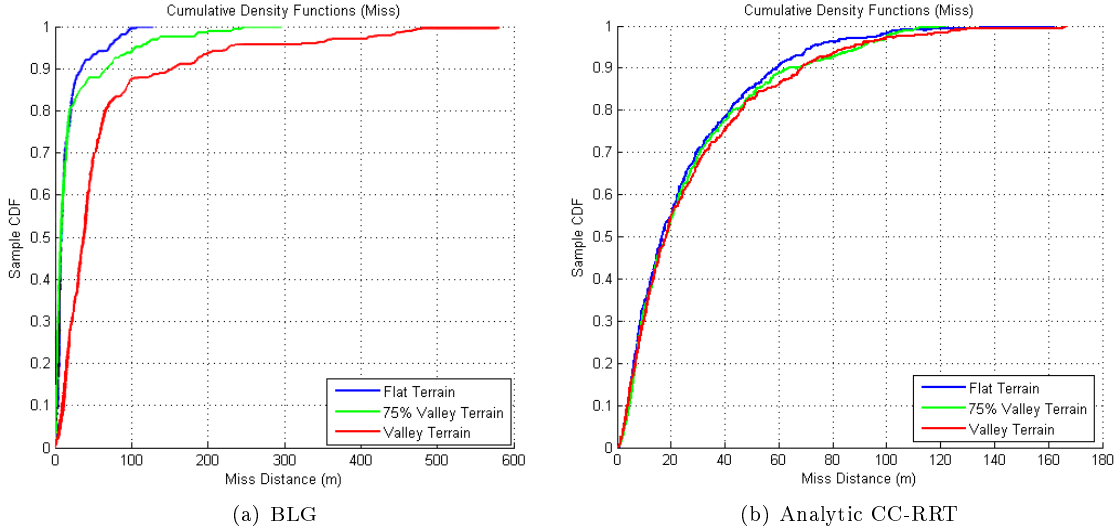


Fig. 11 Miss distance CDF for BLG and Analytic CC-RRT on various terrain, 500 trials

Table 7 Miss distance data for BLG on various terrain, 500 trials (in meters)

BLG	Mean	StDev	50%	80%	90%	95%	98%	Max
Flat Terrain	15.9	19.7	8.9	20.7	35.2	71.3	86.1	107
75% Valley Terrain	22.5	39.5	8.5	20.1	67.1	105	184	247
Valley Terrain	63.5	89.0	37.9	66.1	153	227	431	581

Table 8 Miss distance data for Analytic CC-RRT on various terrain, 500 trials (in meters)

Analytic CC-RRT	Mean	StDev	50%	80%	90%	95%	98%	Max
Flat Terrain	25.3	24.5	16.5	41.8	58.9	72.2	97.8	162
75% Valley Terrain	27.1	26.1	18.1	42.6	63.8	90.4	101	122
Valley Terrain	28.2	28.5	18.0	45.7	68.6	86.5	115	167

or about 9%. Indeed, there is little discernible difference between the shapes of the CDF curves for each terrain environment (Fig. 11(b)). This suggests that analytic CC-RRT is able to maintain consistent performance, regardless of the difficulty of the terrain scenario. While other algorithms may be able to leverage highly simplified terrain to improve accuracy, such as via BLG’s direct optimization, analytic CC-RRT can ensure reasonable performance even under worst-case terrain and wind conditions.

E. CC-RRT Invariance to Initial Altitude

One of the key advantages of analytic CC-RRT relative to other parafoil terminal guidance algorithms is the ability to start planning from any initial altitude. Other approaches in the literature often require an upper limit on the initial altitude for terminal guidance to remain computationally tractable [1–3, 6, 7, 9]. Figure 12 and Table 9 present simulation results on the valley terrain when the initial altitude, p_{zI} , is varied from 500 m (as in Section VII B) to 1000 m and 2000 m; all other

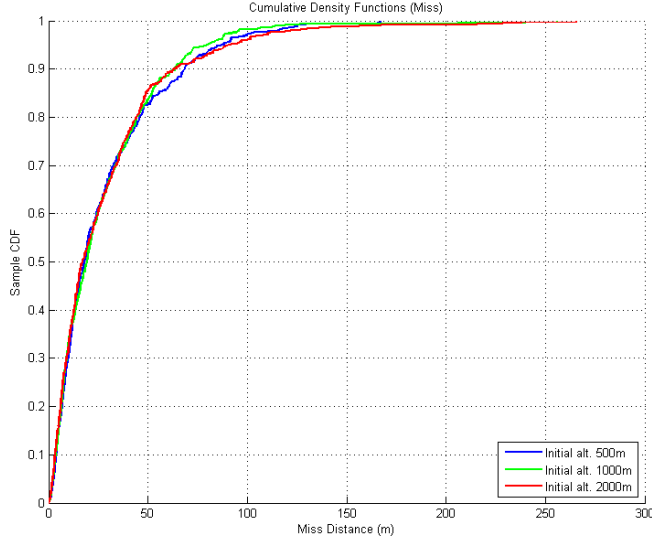


Fig. 12 Miss distance CDF for Analytic CC-RRT from various init. altitudes, 500 trials

Table 9 Miss distance data for Analytic CC-RRT from various init. altitudes, 500 trials (m)

Init. Alt.	Mean	StDev	50%	80%	90%	95%	98%	Max
$p_{zI} = 500\text{m}$	28.2	28.5	18.0	45.7	68.6	86.5	115	167
$p_{zI} = 1000\text{m}$	27.5	28.9	19.3	44.2	63.7	79.5	95.4	256
$p_{zI} = 2000\text{m}$	28.6	33.2	17.0	44.0	63.3	91.2	122	266

conditions are the same as in Section VII B.

The performance of analytic CC-RRT is seen to be largely independent of the starting altitude. Compared to starting at 500 m, both the mean and worst-case accuracy increase only slightly at higher initial altitudes (Table 9). Again, the shape of the CDF curves for all three cases are nearly the same (Fig. 12). This data suggests that analytic CC-RRT is capable of operating at higher altitudes without major deterioration in performance.

VIII. Conclusions

This paper has presented a new approach to online trajectory planning and robust obstacle avoidance for the parafoil terminal guidance problem. The proposed parafoil chance-constrained rapidly-exploring random trees (CC-RRT) algorithm has been shown to demonstrate several key strengths compared to existing approaches in the parafoil terminal guidance literature. By building on rapidly-exploring random trees (RRT), the proposed approach can quickly generate robust solutions of near-arbitrary complexity. Through the use of an intelligent cost-to-go function, the algorithm can operate from nearly any initial condition, and in particular has been demonstrated to be largely invariant to initial altitude. Speed penalties within the cost-to-go ensure that terminal

heading conditions can be incorporated without heavily affecting landing accuracy. By utilizing a multi-modal wind model learned from wind observations, the planner achieves robustness to highly dynamic wind conditions that cannot be obtained through replanning alone. Finally, a novel approach for sampling analytic state uncertainty distributions ensures robust collision avoidance with arbitrary, non-convex, mapped terrain, demonstrated in simulation results to maintain real-time tractability.

Acknowledgments

Research funded by Draper Laboratories through their Independent Research & Development (IR&D) program.

References

- [1] A. Calise and D. Preston. Swarming/flocking and collision avoidance for mass airdrop of autonomous guided parafoils. In *Guidance, Navigation, and Control and Co-located Conferences*. American Institute of Aeronautics and Astronautics, August 2005. doi: 10.2514/1.28586
- [2] N. Slegers, E. Beyer, and M. Costello. Use of variable incidence angle for glide slope control of autonomous parafoils. *Journal of Guidance, Control, and Dynamics*, 31(3):585–596, May 2008. doi: 10.2514/1.32099
- [3] S. Bergeron, K. Tavan and A. Fejzic. Accuglide: Precision airdrop guidance and control via glide slope control. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, May 2011. doi: 10.2514/6.2011-2530
- [4] M. Ward and M. Costello. Adaptive glide slope control for parafoil and payload aircraft. *Journal of Guidance, Control, and Dynamics*, 36(4):1019–1034, July 2013. doi: 10.2514/1.59260
- [5] T. Gimadieva. Optimal control of a gliding parachute system. *Journal of Mathematical Sciences*, 103(1):54–60, 2001.
- [6] D. Carter, L. Singh, L. Wholey, S. Rasmussen, T. Barrows, S. George, M. McConley, C. Gibson, S. Tavan, and B. Bagdonovich. Band-limited guidance and control of large parafoils. In *AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, 2009. doi: 10.2514/6.2009-2981
- [7] O. Yakimenko and N. Slegers. Using direct methods for terminal guidance of autonomous aerial delivery systems. Technical report, DTIC Document, 2009.
- [8] O. Yakimenko and N. Slegers. Optimization of the ads final turn maneuver in 2d and 3d. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, May 2011. doi: 10.2514/6.2011-2604
- [9] J.D. Rogers and N. Slegers. Terminal guidance for complex drop zones using massively parallel processing. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, March 2013. doi: 10.2514/6.2013-1343
- [10] L. Fowler and J. Rogers. Bezier curve path planning for parafoil terminal guidance. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, March 2013. doi: 10.2514/1.1010124
- [11] Brandon Luders, Mangal Kothari, and Jonathan P. How. Chance constrained RRT for probabilistic robustness to environmental uncertainty. In *AIAA Guidance, Navigation, and Control Conference (GNC)*, Toronto, Canada, August 2010. doi: 10.2514/6.2010-8160
- [12] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Iowa State University, October 1998.
- [13] Brandon Luders, Ian Sugel, and Jonathan P. How. Robust trajectory planning for autonomous parafoils under wind uncertainty. In *AIAA Infotech@Aerospace Conference*, Boston, MA, August 2013. doi:10.2514/6.2013-4584
- [14] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):497–516, July 1957.
- [15] D. Jacob. *Introduction to Atmospheric Chemistry*. Princeton University Press, 1999, pp. 16–18.

- [16] Branden J. Rademacher, Ping Lu, Alan L. Strahan, and Christopher J. Cerimele. In-flight trajectory planning and guidance for autonomous parafoils. *Journal of Guidance, Control, and Dynamics*, 32(6):1697–1712, 2008. doi: 10.2514/1.44862
- [17] D. Carter, S. George, P. Hattis, M. McConley, S. Rasmussen, L. Singh, and S. Tavan. Autonomous large parafoil guidance, navigation, and control system design status. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, May 2007. doi: 10.2514/6.2007-2514
- [18] D. Carter, S. George, P. Hattis, L. Singh, and S. Tavan. Autonomous guidance, navigation and control of large parafoils. In *Aerodynamic Decelerator Systems Technology Conferences*. American Institute of Aeronautics and Astronautics, May 2005. doi: 10.2514/6.2005-1643
- [19] D. Delahaye and S. Puechmorel. Aircraft local wind estimation from radar tracker data. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, pages 1033–1038, 2008. doi: 10.1109/ICARCV.2008.4795661
- [20] J. Petrich and K. Subbarao. On-board wind speed estimation for uavs. In *Guidance, Navigation, and Control and Co-located Conferences*. American Institute of Aeronautics and Astronautics, August 2011. doi: 10.2514/6.2011-6223
- [21] K. Hunt and G. P. Nason. Wind speed modelling and short-term prediction using wavelets. *Wind Engineering*, 25, 2001. doi: 10.1.1.15.8764
- [22] X. Jiang, B. Dong, L. Xie, and L. Sweeney. Adaptive gaussian process for short-term wind speed forecasting. In *Proceedings of the European Conference on Artificial Intelligence*, pages 661–666, Amsterdam, The Netherlands, 2010.
- [23] Ian Sugel. Robust planning for autonomous parafoil. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge, MA, September 2013.
- [24] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach, 3rd Edition*. Prentice-Hall, 2010, pp. 693–768.
- [25] Brian Kulis and Michael I. Jordan. Revisiting k-means: New algorithms via Bayesian nonparametrics. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, Edinburgh, Scotland, 2012.
- [26] J. Hartigan and M. Wong. Algorithm AS 136: A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.
- [27] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st edition, 2007, pp. 325–359.
- [28] Brandon Luders and Jonathan P. How. Probabilistic feasibility for nonlinear systems with non-Gaussian uncertainty using RRT. In *AIAA Infotech@Aerospace Conference*, St. Louis, MO, March 2011. doi: 10.2514/6.2011-1589
- [29] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006, pp. 3–432. URL: <http://planning.cs.uiuc.edu/> [cited 22 September 2014].
- [30] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005, pp. 485–604.
- [31] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, September 2009. doi: 10.1109/TCST.2008.2012116