drones

MDPI

Article

Gradient-Guided Search for Autonomous Contingency Landing Planning †

Huseyin Emre Tekaslan ‡ and Ella M. Atkins *,‡

Kevin T. Crofton Aerospace and Ocean Engineering Department, Virginia Tech, Blacksburg, VA 24061, USA; tekaslan@vt.edu

* Correspondence: ematkins@vt.edu

† This paper is an extended version of our paper published in Tekaslan, H.E.; Atkins, E.M. Gradient Guided Search for Aircraft Contingency Landing Planning. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Atlanta, GA, USA, 19–23 May 2025.

‡ These authors contributed equally to this work.

Abstract

The growing reliance on autonomy in uncrewed aircraft systems (UASs) necessitates a real-time solution for assured contingency landing management during in-flight emergencies. This paper presents a novel gradient-guided search algorithm for risk-aware emergency landing trajectory generation with a wing-lift UAS loss-of-thrust use case. This framework integrates a compact four-dimensional discrete search space with aircraft kinematic and ground-risk cost. A multi-objective cost function is employed, combining flight envelope feasibility, optimal descent, and overflown population risk terms. To ensure discrete search convergence, a constrained hypervolume definition is introduced around the destination. A holding pattern identification algorithm is defined to minimize risk during the necessary flight path angle-constrained descent to final approach. Planner effectiveness is validated through randomly generated case studies over a region of Long Island, NY, under steady wind conditions. Benchmark comparisons with a 3D Dubins solver demonstrate the approach's improved risk mitigation and acceptable real-time computation overhead. Future development will focus on integrating collision avoidance into the discrete search-based landing planner.

Keywords: drone safety; assured contingency landing management; discrete search; real-time risk mitigation; path planning

Academic Editor: Abdessattar Abdelkefi

Received: 28 July 2025
Revised: 8 September 2025
Accepted: 11 September 2025
Published: 13 September 2025

Citation: Tekaslan, H.E.; Atkins, E.M. Gradient-Guided Search for Autonomous Contingency Landing Planning. *Drones* **2025**, *9*, 642. https://doi.org/10.3390/drones9090642

1. Introduction

Uncrewed aircraft systems (UASs) are increasingly being developed with electric vertical takeoff and landing (eVTOL) featuring a lift–cruise configuration that dramatically reduces energy consumption compared to rotorcraft and multicopter designs [1,2]. Cargo transport UASs are expected to have a weight and volume similar to today's general aviation aircraft, and full integration into the airspace will enable them to utilize existing vertiports and runways. A variety of conditions, ranging from propulsion module failure to unexpectedly low battery energy or fuel, can render any UAS unable to remain aloft long-term. A lift–cruise eVTOL with low battery energy can safely fly longer, with more of a margin, if it remains in a wing-lift (cruise, fixed-wing) configuration, rather than converting to vertical flight. Whether operating in wing-lift mode, in the case of low energy or an eVTOL propulsion unit failure condition, a contingency landing plan can be the safest for the drone and people on the ground below when the UAS relies on wing-lift throughout

a stable planned descent to land on a nearby runway. Because UASs can glide without consuming fuel in wing-lift mode, it is the most energy-efficient configuration. Gliding is also feasible, regardless of which propulsion unit fails. Therefore, this paper models the UAS in wing-lift mode during contingency descent to landing. The algorithm's reliance on aerodynamic gliding and discrete ground-risk data enables autonomous onboard planning for drones conducting logistics, surveillance, or emergency response missions—applications that are becoming increasingly common in densely populated areas [3–5].

During urgent or emergency landings, the distressed UAS must deal with whatever winds and weather are present. This work, therefore, focuses on a consideration of a steady wind and flight envelope margin and appropriately assumes that other UASs will avoid collision with the distressed drone as it lands. Path planning for urgent or emergency landing requires a four-dimensional solution that meets strict spatiotemporal constraints and minimizes safety risks. Along with the implementation of degraded wing-lift aircraft performance constraints for flight safety, ground risk is quantified by employing discrete population census data in this paper.

Wing-lift aircraft involve non-holonomic motion constraints that require maintaining forward motion at a minimum (stall) speed and limiting the turn radius. Urgent landing descent paths must avoid obstacles, minimize risk, and align with suitable runways while accounting for steady winds in planning and adjusting to actual variable winds during plan execution. There are several methods for aircraft path planning. However, not all of them are suitable for constrained minimum-risk path planning in complex environments. For instance, although they are fast to compute, geometric solutions such as Dubins paths [6–8] solely incorporate path length and curvature metrics. Despite the success of optimal control (OC) and model predictive control (MPC), these methods suffer from initial guess dependency, non-convergence, and low computational efficiency in complex environments [9–13]. The reduced dimensionality that can be achieved through constraints in a search space cannot be applied with OC and MPC. In addition, it is difficult to adapt inherently discrete ground-risk metrics to these methods, as solutions may become stuck at local minima that may not meet feasibility constraints. Similarly, Hamilton–Jacobi–Bellman (HJB) reachability does not scale efficiently due to the curse of dimensionality [14]. Moreover, the HJB value function requires spatial continuous differentiability [15], a requirement that the discrete ground-risk cost violates. Even though such metrics can be approximated by projecting data onto smooth hyperplanes (e.g., Fourier series expansion) this task increases complexity while reducing the data resolution. Note that dynamic cost metric updates, such as ground-risk values derived from mobile phone activity [16] and real-time ground-traffic data [17], render data projection computationally unaffordable for real-time risk quantification as part of contingency planning.

Tree-search-based path planners are well suited for diverse, discrete cost metrics such as ground risk. As an example of related work, Ref. [18] applies a modified A* algorithm to find a shortest feasible path. In [19,20], Monte Carlo tree search was exploited to plan UAS flight paths. A rapidly exploring random tree (RRT) was used in [21–23] to safely navigate a space with obstacles. Ref. [24] used pre-calculated maneuvers for emergency landing planning for a gliding aircraft. The Markov decision process (MDP) formulations in Refs. [25,26] focused on uncertainty in aircraft performance and prognostics-based margins, respectively, in emergency landing planning. A three-dimensional A* variant in [27] balanced flight-path and landing-site risk but faced significant complexity challenges when the path length was substantial. This work advances previous search-based and MDP approaches by integrating risk, kinematic, and flight envelope constraints within a solver that efficiently generates feasible long-distance four-dimensional (4D) landing paths. This paper's approach accounts for operational altitude ranges with a minimum-risk loitering approach and wind effects while maintaining a compact heading-constrained search space.

The discrete search contingency landing planner in this paper adopts a priority queue and operators inspired by the Theta* search algorithm [28] to accommodate flight envelope constraints as a function of steady wind. By fusing feasible path and population risk cost metrics, the presented method simultaneously considers the safety of the landing (e.g., path margin) and risk due to a distressed UAS passing over an urban population. The multi-objective cost function forms a gradient field that significantly reduces state-space exploration thus to minimize computational overhead to support real-time implementation. A constrained 4D position and heading volume around the approach fix is introduced to guarantee search convergence. Since ground risk is negligible at a high altitude, the planner incorporates a three-dimensional (3D) Dubins path solver with a holding point identification scheme to efficiently generate a path to a minimum-risk holding pattern that dissipates altitude. The algorithm's effectiveness is demonstrated through use cases under steady wind at different initial states. Search-based planning results are benchmarked against 3D Dubins path solvers, evaluating overflown population risk, deviations from optimal gliding trajectories, and computational overhead. The key contributions of this work include the following:

1. Efficient and feasible contingency landing planning with a compact 4D discrete search framework guided by a cost function with a constraint margin gradient field.
2. A constrained hypervolume definition around an approach fix to ensure discrete search convergence.
3. A real-time minimum-risk holding pattern placement algorithm and its integration into contingency landing planning.
4. Assured contingency landing plan generation within a prescribed time limit.

This paper is organized as follows. Section 2 introduces fundamental concepts that this work builds upon, including the gliding flight envelope determination of an aircraft in wing-lift mode, reachability, and geometric path planning. Section 4 and 5 present the key components of the contingency landing planner, such as the building blocks of the gradient-guided search incorporating flight feasibility and landing risk assessment with a convergence guarantee. Section 6 details studies over a Long Island, NY, region with a Cessna 182 (C182) model serving as a surrogate for a large eVTOL cargo drone, as in [29].

2. Preliminaries

This section summarizes the principles of wing-lift aircraft flight, reachability, and geometric path planning. These concepts are essential for understanding and developing emergency landing planning procedures.

2.1. Wing-Lift Aircraft Performance

An aircraft in wing-lift configuration is exposed to three fundamental forces [30]: aerodynamic, gravitational, and propulsive, as shown in Figure 1.

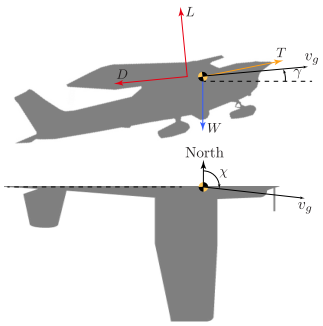

Figure 1. Fundamental forces acting in the aircraft longitudinal plane [31].

A high wing-lift, L, to-drag, D, ratio is the key to efficient fixed wing or "cruise" mode eVTOL flight [32]. Lift is mainly produced by the pressure difference between upper and lower wing surfaces and requires forward flight above a minimum stall speed. In cruise flight, forward thrust, T, generated by propulsion units must counter drag, D, while lift, L, balances the total weight, W. The four primary forces (L, D, T, W) define the flight path angle γ, the angle between the ground velocity, v_g, and the horizontal plane, with $\gamma = 0$ in level flight. The aircraft course angle, denoted as χ, is the angle between north and the horizontal projection of v_g.

The best-glide flight path angle in forward flight, $\gamma_{bg} \in \mathbb{R}^+$, Equation (1), defines the most shallow descent possible without thrust. It is defined by the wing-lift aircraft nonlinear equations of motion, $f(\mathbf{x}, \mathbf{u})$, where $\mathbf{x}$ and $\mathbf{u}$ denote state and control input vectors, respectively.

$$\gamma_{bg} = \underset{L \in [0, L_{max}]}{\arg\max} \frac{L}{D} \quad \text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \quad T = 0 \tag{1}$$

The best-glide for a two-minute standard constant radius turn, $\gamma_{bg,t} \in \mathbb{R}^+$, under steady wind conditions is as formulated in Equation (2), where the bank angle is $\mu \neq 0$. An additional inequality constraint [16] on the maximum bank angle μ_{max} limits the maximum load factor, L/W, to meet structural integrity constraints.

$$\gamma_{bg,t} = \underset{C_L \in [0, C_{L,max}]}{\arg\max} \frac{L}{D} \quad \text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \quad T = 0, \quad |\mu| - |\mu_{\max}| \leq 0 \tag{2}$$

The best-glide angle is important for any emergency situation where an aircraft needs to descend rapidly, even if the engines are still operational but delivering a reduced maximum output.

A good landing begins with a good approach, so it is important to define a stabilized approach speed. Wing flaps are extendable only below the maximum flap extended speed, v_{FE}. The steepest flight path angle, $\gamma_{max} \in \mathbb{R}^+$, for straight gliding, Equation (3), and turning flight, Equation (4), are identified to ensure that the airspeed, v_a, is upper-bounded by v_{FE}.

$$\gamma_{max} = \min -\gamma \quad \text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \quad T = 0, \quad v_a - v_{FE} = 0 \tag{3}$$

The steepest gliding angle for turning, $\gamma_{max,t} \in \mathbb{R}^+$, is per Equation (3), where $\mu \neq 0$.

$$\gamma_{max,t} = \min -\gamma \quad \text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \quad T = 0, \quad v_a - v_{FE} = 0, \quad |\mu| - |\mu_{\max}| \leq 0 \tag{4}$$

The best-glide and steepest-descent flight path angle pair of a C182 [33] are defined as functions of wind speed, v_w, and direction, χ_w, and obtained as in [17]. Given no forward thrust ($T = 0$), the best-glide angle, γ_{bg}, represents the shallowest gliding angle an aircraft can fly at without stalling. While aerodynamically efficient, this angle is susceptible to stall due to disturbances such as wind gusts. Conversely, flying at the steepest glide angle, γ_{max}, requires a higher approach speed, leading to a longer deceleration path. To balance safety and efficiency during emergency approaches, the flight path angle, γ_{opt}, is defined per Equation (5), which lies halfway between γ_{bg} and γ_{max}. γ_{opt} offers an optimal safety margin. This value can be adjusted to provide flexibility in the safe landing path descent profile.

$$\begin{aligned} \gamma_{opt}(v_w, \chi_w) &= \frac{1}{2}\Big[\gamma_{bg}(v_w, \chi_w) + \gamma_{max}(v_w, \chi_w)\Big] \\ \gamma_{opt,t}(v_w, \chi_w) &= \frac{1}{2}\Big[\gamma_{bg,t}(v_w, \chi_w) + \gamma_{max,t}(v_w, \chi_w)\Big] \end{aligned} \tag{5}$$

For clarity, the flight path angle notation is simplified by omitting its input arguments, i.e., $\gamma_{\cdot} = \gamma_{\cdot}(v_w, \chi_w)$. Note that all flight path angles are functions of steady wind conditions, as used to define the aircraft turn radius, R, per Equation (6) [32] for path planning.

$$R = \frac{v_a^2 \cos\gamma}{g \cos\mu} \tag{6}$$

Here, g and μ are gravitational acceleration and the wing bank angle, respectively.

2.2. Reachable Footprint

For emergency landing, potential landing sites must be identified within a reachable area. This area, known as the reachable footprint [16], is determined by the drone's capabilities and the specific emergency. For example, a complete loss of thrust limits the reachable footprint to the gliding range, while a medical emergency prioritizes nearby landing sites that are also close to healthcare facilities to minimize the total transit time. Various methods have been proposed to determine the reachable area for distressed aircraft landing. Aircraft dynamics analysis [34,35], optimal control theory [36], and Hamilton–Jacobi methods [37,38] have been employed with different cost/benefit trade-offs. In this paper, the reachability of the target destination is assumed to be confirmed to focus this research effort on path planning to a designated site.

2.3. Geometric Aircraft Path Planning

The Dubins path [6] provides a geometric solution for transitioning between two states while respecting a minimum turning radius constraint. Originally formulated for two-dimensional paths, assuming constant altitude, this solution has been extended to three-dimensional aircraft path-planning applications [8,39,40]. The literature offers several contingency landing management frameworks with Dubins paths. Ref. [7] modified the kinematic model of Dubins paths to include transient effects such as translational and rotational acceleration in emergency path planning. Ref. [41] extends Dubins paths by incorporating real-time updates for dynamically feasible landing plans under degraded aircraft performance. Dubins path and RRT* are fused in [42] to create emergency landing procedures. Ref. [17] incorporated datalink and the Dubins path to coordinate distressed aircraft and ground traffic for road-based emergency landings.

Conventional Dubins path solutions are formed by an initial minimum radius turn segment, a final minimum radius turn segment, and a straight segment connecting tangents of the initial and final turns. There are four options for the turn–straight–turn type of paths: right–straight–right (RSR), right–straight–left (RSL), left–straight–right (LSR), and left–straight–left (LSL). Dubins [6] also defined solutions with intermediate turns (RLR and LRL), but these are not practical for long-distance traversals between arbitrary configurations. Most Dubins solvers typically evaluate all four path types and select the shortest, assuming constant velocity. Path geometry and length derivations can be found in [30].

To enhance discrete search-based path planning, the proposed algorithm incorporates 3D Dubins path solutions. For instance, Dubins paths can be employed to establish a connection between an initial emergency state and a designated holding point, as well as to link a discrete search-generated trajectory to final approach to the landing runway. Also, S-turn Dubins paths [41] are employed for benchmarking.

UAS state is defined as a tuple , $s = (\varphi, \lambda, h, \chi)$, where $\varphi \in \Phi$, $\lambda \in \Lambda$, $h \in \mathbb{R}_{\geq 0}$, and $\chi \in [0, 2\pi)$, respectively, correspond to latitude, longitude, the altitude above the mean sea

level (MSL), and the course angle. The symbols Φ and Λ denote valid sets of latitude and longitude. The set of all possible states, $\mathcal{S}$, is defined per Equation (7).

$$\mathcal{S} = \{(\varphi, \lambda, h, \chi) \mid \varphi \in \Phi, \lambda \in \Lambda,\ h \in [0, \infty), \chi \in [0, 2\pi)\} \tag{7}$$

Let the initial and final states, respectively, be $s_0 \in \mathcal{S}$ and $s_N \in \mathcal{S}$. A contingency landing path, $\mathcal{P} \subset \mathcal{S}$, consists of a sequence of states from s_0 to s_N, where $\mathcal{P} = \{s_0, s_1, \cdots, s_N\}$. A set of Dubins path solutions between arbitrary states, s_i and s_j, is denoted as $\mathcal{D}(s_i, s_j)$, per Equation (8).

$$\mathcal{D}(s_i, s_j) = \{\mathcal{P}_{\text{type}} \mid \text{type} \in \{\text{RSR, RSL, LSR, LSL}\}\} \tag{8}$$

The shortest Dubins solution, $\mathcal{P}_D$, is defined in Equation (9), where $\mathcal{L}(\mathcal{P})$ is the length of path $\mathcal{P}$.

$$\mathcal{P}_D = \begin{cases} \arg\min_{\mathcal{P} \in \mathcal{D}(s_i, s_j)} \mathcal{L}(\mathcal{P}), & \mathcal{D}(s_i, s_j) \neq \varnothing, \\ \varnothing, & \mathcal{D}(s_i, s_j) = \varnothing \end{cases} \tag{9}$$

Note that $\mathcal{D}(s_i, s_j)$; thus, $\mathcal{P}_\mathcal{D}$ can be an empty set if s_j is unreachable, given s_i.

3. Problem Statement

Ensuring a safe and feasible contingency landing for UASs in emergency scenarios is a critical problem that demands both efficiency and reliability. The objective of this paper is to present and analyze a contingency landing planner that minimizes risk to the overflown population while ensuring the timely definition of a feasible path. This problem presents several key challenges. First, the feasibility of the landing path is constrained due to the distressed drone's flight envelope, necessitating a careful assessment of its aerodynamic limitations, given degraded performance. Second, minimizing risk along the trajectory requires an accurate evaluation and incorporation of risk data. Another fundamental challenge lies in the computational complexity of search-based path planning. The need to explore a continuous search space with theoretically infinite extent complicates real-time decision-making and solution convergence guarantees. This trade-off between runtime efficiency, search completeness, and trajectory risk must be carefully managed to ensure a practical implementation. For this study, the following assumptions are made. First, designated landing sites are assumed to be within the aircraft-reachable footprint. Second, this paper assumes a known steady wind throughout the descent to landing.

4. Methodology

This section introduces the contingency landing planner and its building blocks, including a search-based path-planning algorithm, a solution identification criterion, and a method for defining a minimum-risk holding pattern.

4.1. Contingency Landing Path Planning

Because a Dubins solution is purely geometric, risk metrics cannot be considered. A low-flying large UAS in distress poses a risk to an overflown population. However, an aircraft at high altitude poses a negligible risk to people on the ground below because it will fly well past its current position before reaching the ground, even in a steep descent. Discrete search with a population risk metric is, therefore, only useful when the UAS descends below a crossover altitude, $h_c \in \mathbb{R}_{>0}$. The handling of high and low altitude cases is illustrated in Figure 2.

In a low-altitude emergency, a discrete search planner generates a low-risk, energy-managed path $\mathcal{P}_s(s_0, s'_N)$ from the initial emergency state s_0 to the final search state s'_N.

Final search state s'_N is constrained to lie in a predefined hypervolume around s_N. Then, a final turn to join a final approach path to touchdown state s_t is computed with a 3D Dubins path, $\mathcal{P}_D(s'_N, s_t)$, considering s_N as a fly-by waypoint. Note that the exact initial and final states of the Dubins path for the final turn are, respectively, computed via the Dubins path solver between s'_N and s_N, as well as s_N and s_t, to ensure feasibility with respect to wing-lift aircraft dynamics and kinematic continuity. Formally, a low-altitude solution is defined as follows:

$$\mathcal{P} = \mathcal{P}_s(s_0, s'_N) \cup \mathcal{P}_D(s'_N, s_t) \tag{10}$$

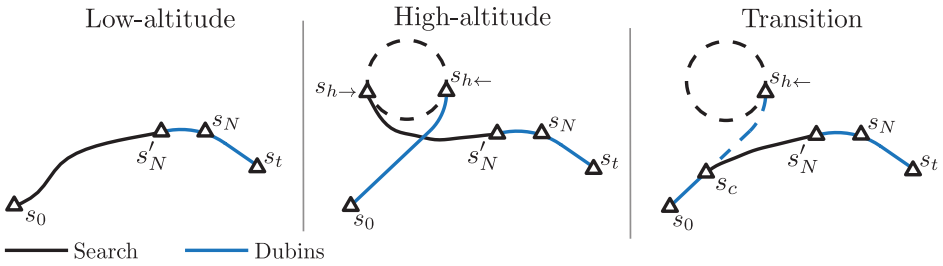

Figure 2. Contingency landing paths for low-altitude, high-altitude, and transition cases. A circular holding pattern is used for high-altitude cases.

In a high-altitude emergency, the distressed aircraft must descend at a flight path angle near γ_{opt} before initiating an approach to landing. Altitude dissipation is achieved by defining a circular holding pattern with inbound state $s_{h\leftarrow}$ and outbound state $s_{h\rightarrow}$. The planner identifies the optimal holding pattern location using a utility function specified below. It connects s_0 to $s_{h\leftarrow}$ with the shortest Dubins path, $\mathcal{P}_D(s_0, s_{h\leftarrow})$. The holding pattern is traversed until reaching $s_{h\rightarrow}$, where a discrete search is initiated to find a low-risk low-altitude landing path to s'_N. A high-altitude solution can be expressed as follows:

$$\mathcal{P} = \mathcal{P}_D(s_0, s_{h\leftarrow}) \cup \mathcal{P}_s(s_{h\rightarrow}, s'_N) \cup \mathcal{P}_D(s'_N, s_t) \tag{11}$$

A transition case arises if $h_0 > h_c$, yet $s_{h\leftarrow}$ is unreachable above h_c. Similar to high-altitude cases, a Dubins path is generated from s_0 to $s_{h\leftarrow}$. The crossover state, $s_c \in \mathcal{P}_D(s_0, s_{h\leftarrow})$, at the crossover altitude on the path to $s_{h\leftarrow}$ is identified. The search-based planner then generates a low-risk path from s_c to s'_N. Then, the solution is finalized with a 3D Dubins path, $\mathcal{P}_D(s'_N, s_t)$. The solution of a transition case is given as follows:

$$\mathcal{P} = \mathcal{P}_D(s_0, s_c) \cup \mathcal{P}_s(s_c, s'_N) \cup \mathcal{P}_D(s'_N, s_t) \tag{12}$$

4.2. Search-Based Path Planning

A UAS flight envelope defines its safe operating limits. These limits include but are not limited to stalling for controllability, the load factor, and the maximum airspeed for structural integrity. Let the degraded wing-lift aircraft flight envelope $\mathbb{A}$ be defined as follows:

$$\mathbb{A} = \{(v_a, R, \gamma, \gamma_t) \mid \text{all within safe limits}\} \tag{13}$$

For a given flight envelope, $\mathbb{A}$, s_0, and s_N, the search space per Equation (7) can be identified using a reachability method. In this study, $\mathcal{S}$ is an infinite, continuous, and non-Euclidean space where the distance between arbitrary states is a geodesic distance. Feasible actions for transition from a parent state to successor states are given by $a(s) = \{a^j(s) \mid j \in \mathbb{Z}_{>0}\}$, where $\lceil j \rceil$ is the branching factor. Each action, a^j, is composed of a course angle change, $\Delta\chi$, a path segment length, which is the great-circle distance

between the parent and successor state, ℓ, and optimal straight gliding and turning flight path angles along segment γ and γ_t, given a known steady wind.

$$a^j(s) = (\Delta\chi, \ell, \gamma, \gamma_t \mid \Delta\chi \in \mathcal{X}, \ell \in \mathbb{R}^+, \gamma \in \mathbb{R}^+, \gamma_t \in \mathbb{R}^+) \tag{14}$$

where $\mathcal{X} \subseteq [0, 2\pi]$ is a set of feasible course angle changes. The set of all feasible actions, $\mathcal{A}$, is per Equation (15), where $|\mathcal{A}| < \infty$.

$$\mathcal{A} = \{a(s) \mid s \in \mathcal{S}\} \tag{15}$$

For a given aircraft flight envelope, $\mathbb{A}$, set of actions, $\mathcal{A}$, initial and goal states s_0 and s_N, state cost function $g(s)$, and steady wind with magnitude v_w and direction χ_w, the 4D search-based path planning problem $\mathcal{T}$ is formulated as in Equation (16), where $\mathcal{P}_s$ is a non-ascending emergency landing path solution.

$$\mathcal{P}_s = \{s \mid s = \mathcal{T}(\mathbb{A}, \mathcal{A}, g(s), s_0, s_N, v_w, \chi_w)\} \tag{16}$$

4.3. Feasible Actions

The action set must consist only of feasible actions, considering potentially degraded flight envelope constraints. For example, while symmetrical lateral maneuverability can be used for left and right turns, considering airspeed and load factor for a single-engine UAS experiencing loss of thrust, asymmetrical course angle rate limits will be present for aircraft with deteriorated lateral controllability caused by conditions such as a control surface jam [16], structural damage [43], and uneven icing [44]. The minimum segment length for a given arbitrary set of course angle changes is further defined below.

A sketch of sequential turns in the horizontal plane is shown in Figure 3. The minimum segment length, ℓ_{min}, that can accommodate sequential turns between state s_i and s_{i+1} is expressed in Equation (17). When expanding state s_i, the next state and action are unknown; hence, $\Delta\chi_{i+1}$ can be substituted with the highest possible course change value to ensure feasibility.

$$\ell_{min} = R\left(\tan\frac{\Delta\chi_i}{2} + \tan\frac{\Delta\chi_{i+1}}{2}\right) \tag{17}$$

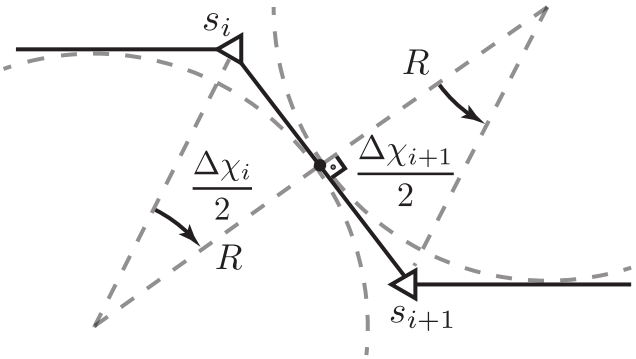

Figure 3. Planar geometry of sequential turns, the dotted lines indicate constant radius turn geometry [31].

The search algorithm uses an adaptive segment length, Equation (18), that reduces search resolution at higher altitudes where ground risk is minimal to improve real-time

performance. This function uses h_i, the altitude at s_i, $\underline{h} \in \mathbb{R}^+$, the floor altitude for the adaptive segment length , and $m \in \mathbb{R}^+$, the rate of segment length change per unit altitude.

$$\ell_i = \begin{cases} \ell_{min} + m(h_i - \underline{h}), & h_i > \underline{h} \\ \ell_{min}, & h_i \leq \underline{h} \end{cases} \tag{18}$$

Per Equation (5), γ_{opt} and $\gamma_{opt,t}$ are determined within $\mathbb{A}$ for given steady wind conditions. Therefore, a feasible action, a^j, at state s is a function of wind speed v_w and wind direction χ_w.

4.4. State Expansions

Successor states are generated by expanding parent states. A successor of parent s resulted from the j-th feasible action a^j of s is denoted as $s^j = (\varphi^j, \lambda^j, h^j, \chi^j)$. The course angle of a successor state, χ^j, is

$$\chi^j = |\chi + \Delta\chi^j|_{2\pi} \tag{19}$$

where $\chi \in s$ and $\Delta\chi^j \in a^j$. The operator $|\ .\ |_\alpha$ wraps an angle at α. The coordinates of a successor state are computed by using the World Geodetic System 1984 (WGS84) [45] model $\mathcal{W} : \mathbb{R}_{>0} \times \mathbb{R} \times \mathcal{S} \to \Phi \times \Lambda$, per Equation (20). Let $c(s) : \mathcal{S} \to \Phi \times \Lambda$ be an operator, such that $c(s) = [\varphi\ \lambda]^\intercal$. Simply, $\mathcal{W}$ is a function that returns the destination coordinates, given a reference state and action variables.

$$c(s^j) = \mathcal{W}(\ell^j, \chi^j \mid s) \tag{20}$$

The altitude of the successor state h^j Equation (21) is computed with a consideraton of turning and straight-gliding traversals where $\gamma^j_{\cdot} \in a^j$.

$$h^j = h - \ell_{turn} \tan\gamma^j_{opt,t} - (\ell - \ell_{turn}) \tan\gamma^j_{opt} \tag{21}$$

The turning traversal is denoted as ℓ_{turn} and calculated as in Equation (17). For convenience, the transition model $E : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is specified to apply an action to a state, which composes relations Equations (19)–(21). The set of successors for parent state s is

$$C(s) = \{s^j \mid s^j = E(s \mid a^j), a^j \in \mathcal{A}\}. \tag{22}$$

Reciprocally, the parent of a successor state is $s = \pi(s^j)$.

4.5. Cost Functions with a Constraint Margin Gradient Field

The path-planning cost function is constructed by four distinct terms: (1) a gradient function to prioritize state expansion with an optimal descent path angle, (2) a gradient function to guide search toward the target location, (3) a gradient function that guides search toward the desired course angle as the target location is approached, and (4) a term that discourages flight over densely populated urban areas. The total cost of a state, $g(s)$, is given in Equation (23).

$$g(s) = w_1[\rho(s)g_{d,1}(s)] + w_2[(1 - \rho(s))g_{d,2}(s)] + w_3 g_\chi(s) + w_4 g_p(s), \quad \sum_{i=1}^{4} w_i = 1 \tag{23}$$

Here, $g_{d,1}, g_{d,2}$, g_χ, and g_p are, respectively, optimal descent, direct distance, course angle, and population cost functions. Terms $w_i \in \mathbb{R}_{\geq 0}$ are weight coefficients, while $\rho(s) : \mathcal{S} \to [0, 1]$ is an adaptive weighting factor. First, common terms used in cost

functions are introduced. An approximate shortest path length to the goal is calculated based on a combination of turn and straight sections.

$$d_{min}(s) = d_{gc}(s) + R|\theta(s, s_N) - \chi|_\pi \tag{24}$$

The function $d_{gc}(s) : \mathcal{S} \to \mathbb{R}_{\geq 0}$ represents the great-circle distance from state s to s_N, where $\theta : \mathcal{S} \times \mathcal{S} \to [0, 2\pi)$ returns the bearing angle between two states. The remaining best-glide range, $d_{bg}(s) \in \mathbb{R}_{\geq 0}$, given the altitude loss from s to s_N, is defined as follows:

$$d_{bg}(s) = \frac{h - h_N}{\tan \gamma_{bg}} \tag{25}$$

States with $d_{bg}(s) < d_{min}(s)$ indicate unreachability and are pruned from further expansion. d_{opt} is defined as an approximate optimal distance to be flown to descend to goal altitude h_N from initial altitude h_0 at optimal glide slope γ_{opt}, initially estimated based on the relative wind between s_0 and s_N.

$$d_{opt} = \frac{h_0 - h_N}{\tan \gamma_{opt}} \tag{26}$$

4.5.1. Optimal Gliding Cost

Two cost functions are employed to encourage progress toward the goal state region. The first function, $g_{d,1}(s) : \mathcal{S} \to [0, 1]$, Equation (27), prioritizes the necessary altitude loss to reach the goal state. It serves as a measure of how efficiently the estimated total path length (the first two terms in the numerator) compares to the optimal descent path length. Consequently, it penalizes deviations from the optimal path length caused by deviation from the optimal flight path angle using the heuristics.

$$g_{d,1}(s) = \frac{|\ell(s) + d_{min}(s) - d_{opt}|}{d_{opt}} \tag{27}$$

Here, $\ell : \mathcal{S} \cup \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is the total path length leading s_0 to s as a function of either state s or time t to reach s, per Equation (28).

$$\ell(x) = \begin{cases} \ell_s(s), & x \in \mathbb{R}^4, \\ \ell_t(t), & x \in \mathbb{R}_{\geq 0}. \end{cases} \tag{28}$$

The cost $g_{d,1}(s)$ is proven bounded, per Lemma 1.

Lemma 1. *For a non-ascending landing path with $\gamma_{bg} \leq \gamma$, and a UAS whose optimal glide angle satisfies $\tan \gamma_{opt} \leq 2 \tan \gamma_{bg}$, the cost function $g_{d,1}(s)$ is bounded within the closed interval $[0, 1]$, $\forall s \in \mathcal{S}$.*

Proof of Lemma 1. The boundedness of $g_{d,1}(s)$ can be established by analyzing the limits of $\ell(s) + d_{min}(s)$, given that d_{opt} is a constant. Given the path slope γ, $\gamma_{bg} \leq \gamma$, a feasible landing path cannot exceed the best-glide range d_{bg}; hence, the upper bound is as follows:

$$\ell(s) + d_{min}(s) \leq \frac{h_0 - h_N}{\tan \gamma_{bg}} = d_{bg}. \tag{29}$$

Defining $\Delta h = h_0 - h_N$ and substituting d_{bg} into the expression for $g_{d,1}(s)$, one obtains the following:

$$\begin{aligned} g_{d,1}(s) &= \frac{|\ell(s) + d_{min}(s) - d_{opt}|}{d_{opt}} \\ &\leq \frac{|d_{bg} - d_{opt}|}{d_{opt}} \\ &= \frac{\Delta h / \tan \gamma_{bg} - \Delta h / \tan \gamma_{opt}}{\Delta h / \tan \gamma_{opt}} \\ &= \frac{\tan \gamma_{opt}}{\tan \gamma_{bg}} - 1. \end{aligned} \tag{30}$$

Given an aircraft flight envelope with $\tan \gamma_{opt} \leq 2 \tan \gamma_{bg}$, it follows that

$$\frac{\tan \gamma_{opt}}{\tan \gamma_{bg}} - 1 \leq 1 \tag{31}$$

Thus, $g_{d,1}(s) \leq 1$ is found. Next, the lower bound is examined. The quantity $\ell(s) + d_{min}(s)$ attains its minimum if and only if the initial and goal states coincide in latitude and longitude, i.e., $c(s_0) = c(s_N)$. Given these conditions, $\ell(s) = 0$ and $d_{min}(s) = 0$, leading to the following:

$$g_{d,1}(s) = \frac{|-d_{opt}|}{d_{opt}} = 1 \tag{32}$$

Per Equations (31) and (32), $g_{d,1}(s)$ is found to be upper-bounded by 1. With a note that $g_{d,1}(s)$ is non-negative due to the absolute value term in its definition, the following is concluded:

$$g_{d,1}(s) \in [0, 1], \quad \forall s \in \mathcal{S}. \tag{33}$$

Thus, $g_{d,1}(s)$ is bounded within the closed interval $[0, 1]$, completing the proof. □

4.5.2. Direct Distance Cost

Using only $g_{d,1}(s)$ ensures that the path follows the optimal length using heuristics, but it can lead to wasted potential energy by expanding nodes away from the goal, particularly in high-altitude emergency cases. To address this, a second function, $g_{d,2}(s) : \mathcal{S} \to [0, 1]$, a heuristic estimate for the shortest path to the goal normalized by d_{opt}, is introduced, per Equation (34). This function prioritizes states that are closer to the goal, regardless of their descent profile.

$$g_{d,2}(s) = \frac{d_{min}(s)}{d_{opt}} \tag{34}$$

Due to reachability limitations, the optimal flight path angle γ_{opt} cannot be determined per Equation (5) if $d_{min}(s_0) > d_{opt}$. In such cases, a shallower flight path angle—optimal for the specific emergency scenario—must be used to ensure reachability, such that $d_{bg}(s_0) \leq d_{min}(s_0) \leq d_{opt}$. Therefore, the condition $0 \leq d_{min}(s) \leq d_{opt} < \infty, \forall s \in \mathcal{S}$ inherently guarantees the boundedness of $g_{d,2}(s)$ on a closed interval $[0, 1]$, eliminating the need for a separate proof.

By weighting $g_{d,1}(s)$ and $g_{d,2}(s)$, the prioritization of expanding states towards the goal and the establishment of the optimal descent profile are balanced. To achieve this balance, upper- and lower-bound radii around s_N; respectively, $\bar{r}$ and $\underline{r}$, are specified to calculate $\rho(s)$ to limit outward branching. Equation (35) defines $\bar{r}$, while $\underline{r}$ is a predetermined constant.

$$\bar{r} = \frac{1}{2}(d_{opt} + d_{min}(s_0)) \tag{35}$$

The weighting factor is

$$\rho(s) = \begin{cases} 0, & d_{min}(s) \geq \bar{r} \\ \dfrac{\bar{r} - d_{min}(s)}{\bar{r} - \underline{r}}, & \underline{r} < d_{min}(s) < \bar{r} \\ 1, & d_{min}(s) \leq \underline{r} \end{cases} \tag{36}$$

Equation (36) guides state expansion toward s_N by adjusting the influence of $g_{d,1}(s)$ and $g_{d,2}(s)$, based on the state's location in the search space. Specifically, according to Equation (23), when s is farther than $\bar{r}$ from s_N, $g_{d,1}(s)$ is set to zero, preventing outward branching and conserving excess altitude. This allows the aircraft to later use its potential energy to navigate around densely populated areas. Conversely, when s is closer than $\underline{r}$ to s_N, $g_{d,2}(s)$ is eliminated, prioritizing the optimal descent path by managing altitude loss.

To facilitate further analysis, let $g_d(s)$ be as follows:

$$g_d(s) = \rho(s) g_{d,1}(s) + (1 - \rho(s)) g_{d,2}(s) \tag{37}$$

Now, the boundedness of $g_d(s)$ is established in Theorem 1.

Theorem 1. *The weighted sum of $g_{d,1}(s)$ and $g_{d,2}(s)$ is bounded within the closed interval $[0, 1]$, $\forall s \in \mathcal{S}$.*

Proof of Theorem 1. Per Lemma 1, $g_{d,1}(s)$ is bounded within $[0, 1]$. Additionally, by definition, both $g_{d,2}(s)$ and $\rho(s)$ are constrained within $[0, 1]$. Since $g_d(s)$ is a convex combination of these bounded functions, it follows that $g_d(s)$ remains within $[0, 1]$, $\forall s \in \mathcal{S}$. The constant weights w_1 and w_2 do not alter the boundedness of the linear combination. □

Figure 4 demonstrates the evolution of $g_d(s)$ during path planning. The vector field visualizes the gradient of $g_d(s)$, $\nabla g_d(s) = [\partial g_d(s)/\partial \lambda \;\; \partial g_d(s)/\partial \varphi]^\intercal$, indicating the preferred direction of state expansion.

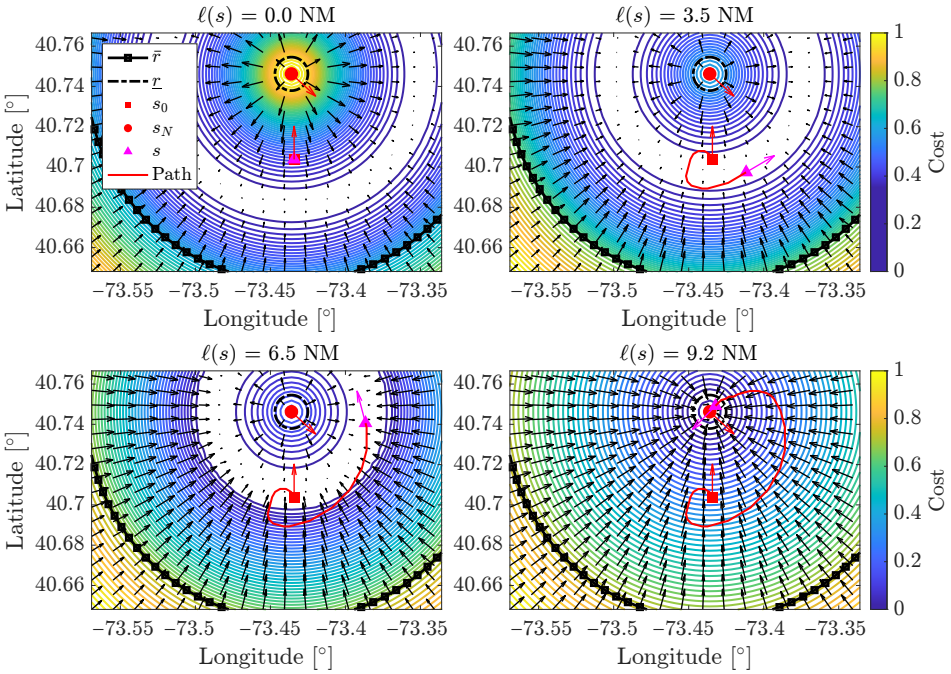

Figure 4. An evolution of $g_d(s)$ with a vector field around the goal state. Red and magenta arrows show the course angles. Black arrows indicate the gradient vector field.

Initially, in the upper left figure, where $\ell(s) = 0$ NM, the vector field around state s points away from the goal state. This outward direction of state expansion occurs as $g_d(s)$ forms a repellent, high-cost region around goal state s_N due to $d_{opt} > d_{min}(s_0)$. As the path planning progresses, the search captures and follows the optimal descent path, as in the upper right and lower left corners. Finally, it converges toward s_N by following the minimum cost isoline, as depicted in the lower right figure. If the initial altitude were set lower, the minimum cost region would lie between s_0 and s_N due to $d_{opt} \approx d_{bg}$, leading state expansion immediately toward s_N.

4.5.3. Course Angle Cost

The costs $g_{d_1}(s)$ and $g_{d_2}(s)$ enable a search to reach the goal region, yet they do not reward the desired course angle. It is challenging to match the exact target course with a discrete search. Explicitly incorporating the course angle difference in a cost function would eventually enforce states to only expand parallel to the goal course, which impedes necessary turns to match the goal location. Thus, a course angle cost that represents the preferred direction for approaching the goal state in lieu of exactly matching the landing runway course angle is proposed.

A unit normal vector, $\vec{n}_{\chi_N}$, represented in a north–east–down (NED) coordinate system and aligned with the target course χ_N, is used to partition the cost map into regions of higher and lower cost, per Equation (38), where $\mathcal{R}^{NED}_{(\varphi,\lambda)} : \Phi \times \Lambda \to \mathbb{R} \times \mathbb{R}$ is the transformation operator from a geodetic system to the NED coordinate system.

$$\tilde{g}_\chi(s) = \vec{n}_{\chi_N} \cdot \left[\mathcal{R}^{NED}_{(\varphi,\lambda)}(c(s)) - \mathcal{R}^{NED}_{(\varphi,\lambda)}(c(s_N)) \right] \tag{38}$$

It should be noted that the vector operations in Equation (38) must be in a local coordinate system to eliminate the asymmetrical gradient field with respect to χ_N due to the Earth's curvature. The scalar product in Equation (38) evaluates the projection of the vector from an arbitrary state, s to s_N, onto $\vec{n}_{\chi_N}$. A $\tilde{g}_\chi(s)$ value of zero indicates that s lies on the line perpendicular to $\vec{n}_{\chi_N}$ that passes through $c(s_N)$, bisecting the search map with a vertical plane. Such states require a 90° turn to align with the final course, similar to the common practice in an airport traffic pattern. States with $\tilde{g}_\chi(s) < 0$ are located in the region behind the bisecting line and require a course change of less than 90° to reach the goal. Conversely, states with $\tilde{g}_\chi(s) > 0$ are penalized, as they necessitate a course angle change greater than 90° to align with the landing site. Equation (38) is exploited to further define the course angle cost $g_\chi(s)$, per Equation (39), where $\kappa_\chi > 0$ is a normalization constant to ensure the cost is bounded on the closed interval $[0, 1]$. The scalar product term is multiplied by distance terms to ensure that distant states can be explored in any direction, while nearby states are restricted to the region behind the bisecting line. The $\max(.,.)$ operator is introduced to zero out the course angle cost of states with $\tilde{g}_\chi(s) < 0$.

$$g_\chi(s) = \frac{1}{\kappa_\chi} \max\left(0, \tilde{g}_\chi(s) \frac{d_{opt}}{d_{gc}(s)}\right) \tag{39}$$

The normalizer κ_χ can easily be determined based on the maximum course cost value in the search space $\mathcal{S}$, per Equation (40).

$$\kappa_\chi = \sup_{s \in \mathcal{S}} \tilde{g}_\chi(s) \frac{d_{opt}}{d_{gc}(s)} \tag{40}$$

Overall, the $\max(.)$ operator ensures a lower bound of zero, while κ_χ normalizes the upper bound to one, simplifying the need for a boundedness proof for $g_\chi(s)$ on the closed interval $[0, 1]$.

An illustration of Equation (39) for $\chi_N = 132^\circ$ is shown in Figure 5. The arrows point in the course angle directions, and the vector field of $\partial g_\chi = [\partial g_\chi/\partial\lambda \;\; \partial g_\chi/\partial\varphi]^\intercal$ encourages the search algorithm to expand states toward regions where $g_\chi(s) = 0$, enabling a feasible final approach.

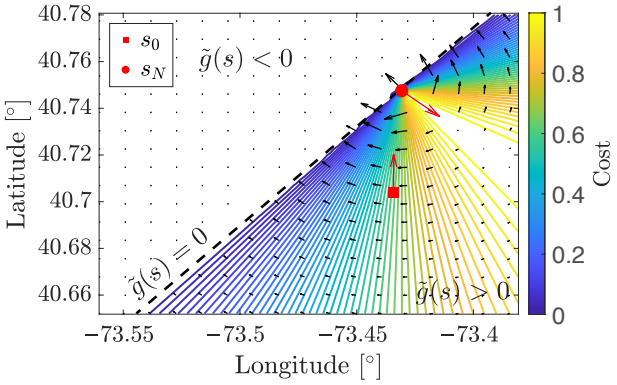

Figure 5. A map of the course angle cost. The black dashed line illustrates the state-bisecting plane. Red arrows show the course angles. Black arrows indicate the gradient vector field.

The function $g_\chi(s)$ alone is insufficient to guide the search to the solution, as it only provides guidance for one half of the horizontal cost map. However, stacking $g_\chi(s)$ on top of $g_{d,1}(s)$ and $g_{d,2}(s)$ alters the gradient field direction in the vicinity of the goal state, allowing state expansion in the right direction.

The combined $g_{d,1}(s)$, $g_{d,2}(s)$, and $g_\chi(s)$ are depicted in Figure 6. In contrast to the solution shown in Figure 4, the state expansion crosses the bisecting line from $\tilde{g}_\chi(s) > 0$ to $\tilde{g}_\chi(s) < 0$ while simultaneously capturing the optimal descent path.

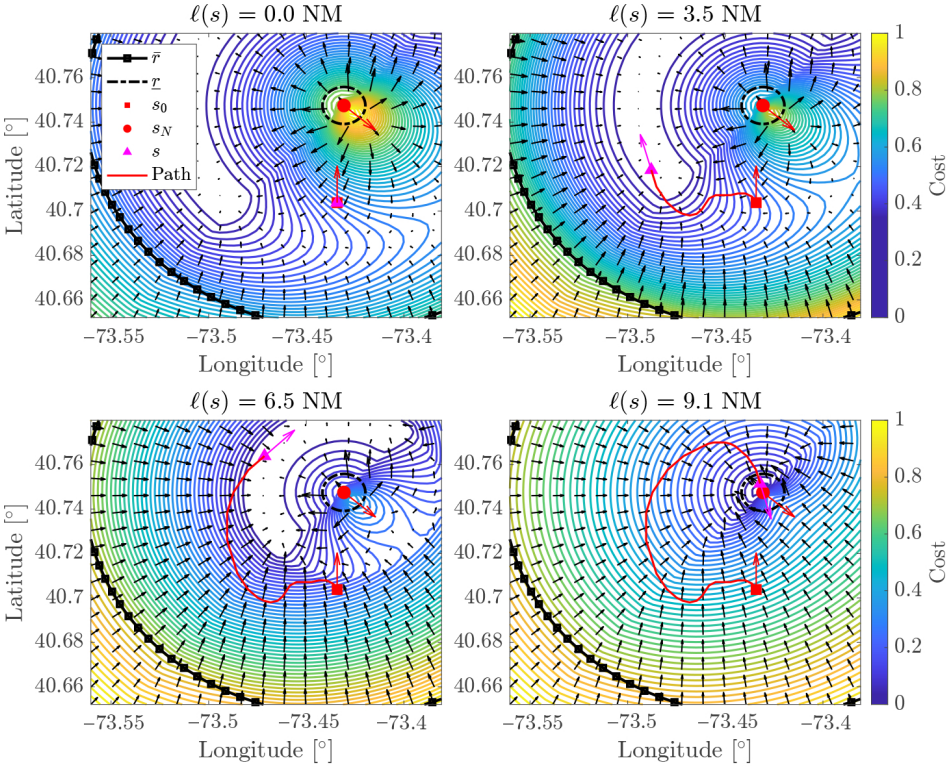

Figure 6. Evolution of the total distance and course angle costs with a vector field around a goal state. Red and magenta arrows show the course angles. Black arrows indicate the gradient vector field.

4.5.4. Population Cost

The population cost is computed by integrating the overflown population density over time. Let the population density function be $\tilde{\eta} : \Phi \times \Lambda \to \mathbb{R}_{\geq 0}$. The function $\tilde{\eta}$ is reformulated as a function of time for a known state, per Equation (41), where $\circ$ is the composition operator. In this paper, the population and land area dataset shared by the U.S. Census Bureau [46] is utilized to obtain population density as a function of time $\eta(t \mid s, v)$, which will be denoted as $\eta(t)$ for convenience. For population risk estimation, the wind effect is ignored; thus, $v = v_a$.

$$\eta(t \mid s, v) = \tilde{\eta} \circ \mathcal{W}(vt, \chi \mid s) \ \text{ persons/m}^2 \tag{41}$$

To reflect the true population risk, a weighting function, $w_{p,1}(s)$, Equation (42), is introduced to penalize η under a lower altitude path.

$$w_{p,1}(t) = \begin{cases} 1 - \dfrac{h(t)}{\min(h_0, h_c)}, & h(t) \leq h_c \\ 0, & h(t) > h_c \end{cases} \tag{42}$$

Here, the crossover altitude h_c is used for the ceiling altitude for population risk consideration. Therefore, population risk is not taken into account for path planning above h_c.

In the case of an emergency, the population residing under a given destination is not a factor, as the aircraft is supposed to be approaching and landing at an airport runway in this work. Thus, the impact of population risk is reduced proportionally to the remaining traversal using a quadratic cost, $w_{p,2}(t)$, Equation (43).

$$w_{p,2}(t) = \left(1 - \frac{\ell(t)}{d_{opt}}\right)^2 \tag{43}$$

Time t is the input argument to Equation (28) for integration purposes.

Let the cumulative overflown population risk to reach s^j from s in flight time t be the integration of weighted Equation (41) with respect to time, per Equation (44).

$$g_p(s^j) = \frac{1}{\kappa_p} \int_0^t w_{p,1}(t) w_{p,2}(t) \eta(t) dt \tag{44}$$

Here, the flight time t is estimated using ℓ^j / v. The boundedness of $g_p(s)$ depends on the proper scaling by $\kappa_p \in \mathbb{R}_{\geq 0}$ (45).

$$\kappa_p = \kappa_w \kappa_\eta t \tag{45}$$

Per Equation (45), the population risk is time-averaged and normalized by the upper bound of the weight product $\kappa_w = \sup w_{p,1}(t) w_{p,1}(t)$ and the maximum population density κ_η within the search space $\mathcal{S}$ (46).

$$\kappa_\eta = \sup_{s \in \mathcal{S}} \tilde{\eta}(c(s)) \tag{46}$$

The magnitudes of each cost term must be normalized and/or appropriately weighted in path planning for an unbiased solution. To prove the boundedness of $g_p(s)$ in numerical implementations, two lemmas are provided.

Lemma 2. *The population density function $\eta(t)$ is Riemann-integrable on $[0, T]$.*

Proof of Lemma 2. A function is Riemann-integrable on a closed interval, $[0, T]$, if and only if it is bounded and the lower and upper integrals are equal to each other [47].

First, the boundedness of $\eta(t)$ is established. Since $\eta(t)$ represents population density, a physical quantity, it is inherently bounded over the finite time interval $[0, T]$. Next, the discontinuities of $\eta(t)$ are taken into account. By its nature, $\eta(t)$ is piecewise continuous. Consider a partition, $P = \{t_0, t_1, \cdots t_n\}$ on $[0, T]$, such that $0 = t_0 < t_1 < \cdots < t_n = T$. The norm of the partition P is

$$||P|| = \max_{0<i<n} \{(t_i - t_{i-1})\} \tag{47}$$

Let the lower sum $\underline{S}(P, \eta(t))$ and upper sum $\bar{S}(P, \eta(t))$ of $\eta(t)$, given the partition P, be

$$\begin{aligned} \underline{S}(P, \eta(t)) &= \sum_{i=1}^{n} \inf\{\eta(t) \mid t_{i-1} < t < t_i\}(t_i - t_{i-1}) \\ \bar{S}(P, \eta(t)) &= \sum_{i=1}^{n} \sup\{\eta(t) \mid t_{i-1} < t < t_i\}(t_i - t_{i-1}) \end{aligned} \tag{48}$$

The function $\eta(t)$ has a finite number of discontinuities corresponding to discrete transitions (e.g., entering or leaving census blocks). Each discontinuity has zero width. Therefore, it is deduced that

$$\lim_{\|P\|\to 0} \underline{S}(P, \eta(t)) = \lim_{\|P\|\to 0} \bar{S}(P, \eta(t)) = \int_0^T \eta(t) \tag{49}$$

$\eta(t)$ is Riemann-integrable on $[0, T]$. □

Lemma 3. *There exists a point, $\ell^*(t) \in (0, d_{opt})$, at which the product $w_{p,1}(t)w_{p,2}(t)$ attains its maximum value, denoted as $\kappa_w \in \mathbb{R}_{\geq 0}$, for $h_0 \leq h_c$.*

Proof of Lemma 3. Substituting $h(t) = h_0 - \ell(t)\tan\gamma$ into $w_{p,1}(t)w_{p,2}(t)$, the product is expressed in terms of $\ell(t)$. To find its maximum, one can take the derivative and equate it to zero:

$$0 = \frac{\partial}{\partial \ell(t)} w_{p,1}(t)w_{p,2}(t) = \left(1 - \frac{\ell(t)}{d_{opt}}\right)\left(\frac{\tan\gamma}{h_0}\right)\left(\frac{d_{opt} - 3\ell(t)}{d_{opt}}\right) \tag{50}$$

Solving for $\ell^*(t)$,

$$\ell^*(t) = \frac{d_{opt}}{3} \tag{51}$$

Substituting $\ell^*(t)$ into $w_{p,1}(t)w_{p,2}(t)$, the upper bound κ_w is obtained:

$$\kappa_w = \frac{4}{27}\frac{d_{opt}\tan\gamma}{h_0} \tag{52}$$

Thus, $w_{p,1}(t)w_{p,2}(t)$ is bounded by κ_w, completing the proof. □

Theorem 2. *The population cost $g_p(s)$ is bounded on the closed interval $[0, 1]$, $\forall s \in \mathcal{S}$.*

Proof of Theorem 2. Since $w_{p,1}(t)$, $w_{p,2}(t)$, and $\eta(t)$ are nonnegative, it is concluded that $g_p(s) \geq 0$. To find the upper bound, consider the magnitude of (44).

$$|g_p(s)| = \left|\frac{1}{\kappa_p}\int_0^t w_{p,1}(t)w_{p,2}(t)\eta(t)dt\right| \tag{53}$$

The function $g_p(s)$ is Riemann-integrable, per Lemma 2; thus, the triangular inequality for integrals can be applied to Equation (53) [48]:

$$|g_p(s)| \leq \frac{1}{\kappa_p} \int_0^t |w_{p,1}(t) w_{p,2}(t) \eta(t)| dt \tag{54}$$

Using the Cauchy–Schwarz inequality,

$$|g_p(s)| \leq \frac{1}{\kappa_p} \int_0^t |w_{p,1}(t) w_{p,2}(t)||\eta(t)| dt \tag{55}$$

Per Lemma 3 and Equation (46),

$$|g_p(s)| \leq \frac{1}{\kappa_p} \int_0^t \kappa_w \kappa_\eta dt = \frac{1}{\kappa_p} \kappa_w \kappa_\eta t = 1 \tag{56}$$

□

This study implements the R*-Tree [49], a spatial indexing algorithm, to efficiently query population density in real time. The process begins by assigning a unique identification number (ID) to each polygon (i.e., census block) in the dataset. These polygons are then organized hierarchically in an R*-tree. The intermediate nodes store bounding boxes that enclose multiple polygons, allowing for efficient spatial pruning, while the leaf nodes contain the smallest bounding boxes, along with the corresponding polygon IDs.

To perform a query, the R*-Tree first retrieves the smallest bounding box that contains the query coordinate, quickly eliminating large portions of the dataset. Then, a ray-casting algorithm is used to precisely determine which polygon contains the query point among the remaining candidates. Finally, once the correct polygon is identified, its population and land area values are retrieved from a lookup table using the polygon's ID. The normalized density values are then used in trapezoidal integration. This approach significantly reduces the computational cost of population density queries, making real-time overflown population risk estimation feasible.

4.6. Feasible Solution Identification for Discrete Search

In a 4D discrete search space, reaching an exact goal state, s_N, cannot be guaranteed. Therefore, there are two parts of the proposed search algorithm: (1) discrete search to a feasible region around a goal state to identify s'_N, and (2) a Dubins solver that can always find a geometric solution from the end of the search path to finalize the landing path. Let $\mathcal{Z} \subseteq \mathcal{S}$ be the solution identification set, defining the set of feasible final search states. A state, s, is considered final, s'_N, if and only if the following conditions are met:

1. First, $d_{gc}(s)$ must fall within a defined annulus, per Equation (57).

$$R_i \leq d_{gc}(s) \leq R_o \quad \text{s.t.} \quad 2R_o \geq \ell \tag{57}$$

   This annulus, centered on s_N, is independent of altitude. Ensuring that the outer diameter $2R_o$ is at least the length of one discrete search segment, ℓ, guarantees state expansion within the outer circle. The inner radius R_i ensures the feasibility of a Dubins path connecting the search solution to the final approach.
2. Second, the flight path angle of the remaining traversal γ_{rem} from s'_N to s_N should adhere to the constraints specified in Equation (58).

$$\gamma_{bg} \leq \tan^{-1} \frac{h_N - h'_N}{d_{gc}(s)} \leq \gamma_{max} \tag{58}$$

This constraint introduces altitude bounds to $\mathcal{Z}$.

3. Third, s must lie behind s_N, such that $g_\chi(s) = 0$. This effectively excludes states requiring a significant course angle change to join the final approach.
4. Four, a Dubins path must be feasible from s'_N to s_t.

Formally, the solution identification volume $\mathcal{Z}$ is per Equation (59).

$$\mathcal{Z} = \{s \mid s \in \mathcal{S}, R_i \leq d_{gc}(s) \leq R_o, \gamma_{bg} \leq \gamma_{rem} \leq \gamma_{max}, g_\chi(s) = 0, \mathcal{D}(s, s_t) \neq \varnothing\} \tag{59}$$

The first and third conditions are treated as hard constraints, while modifications to the flight paths are permitted if the second and fourth conditions are not met. If the flight path angle constraints are satisfied, the optimal flight path angle remains unchanged. Otherwise, the gliding angles of both the discrete search and Dubins solutions must be adjusted. However, any such adjustments must ensure that the resulting flight path angle remains within the aircraft's flight envelope for all segments of the flight. An example of a solution identification set is demonstrated in Figure 7. The volume between steepest-glide and best-glide surfaces defines the states of $\mathcal{Z}$, which holds the potential final search state s'_N.

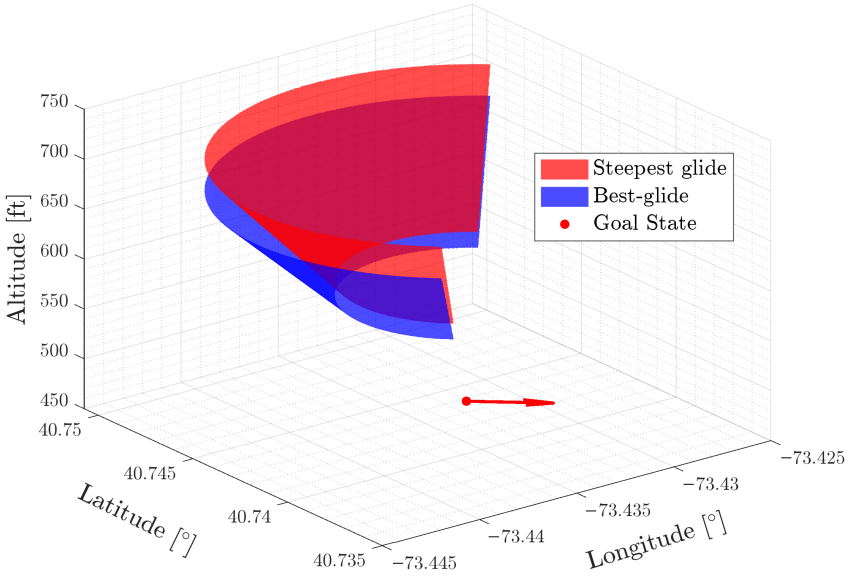

Figure 7. A representative solution identification set. Red arrow shows the course angle of the goal state.

4.7. Minimum-Risk Holding Pattern Identification

Optimal holding point identification for UAS emergencies is based on leveraging census population data and geospatial analysis. This process also ensures compliance with safety constraints, such as the avoidance of no-fly zones within search space $\mathcal{N} \subset \mathcal{S}$ and a sufficient distance from the goal to enable maneuvering. The developed approach involves offline and online parts. The offline part begins by defining the circular area around the goal state, $\mathcal{H}$, Equation (60), to identify candidate holding patterns, considering optimal glide performance.

$$\mathcal{H} = \{\Delta p \mid \Delta p = \bigcup_{i=1}^{n} e_i, e_i = \{(\varphi_i, \lambda_i), (\varphi_j, \lambda_j)\}, A_{\Delta p} > \pi R^2, \Delta p \cap \mathcal{N} = \varnothing\} \tag{60}$$

Here, $\mathcal{H}$ is a set of polygons, Δp, where e is an edge of a polygon with n edges. Each edge is defined by two vertices in latitude and longitude. The algorithm merges small, scattered polygons into larger polygons with area $A_{\Delta p}$, ensuring that the merged areas meet

a minimum threshold defined by the turn radius so that the holding pattern is completely situated inside the polygon. It is worth mentioning that simply filtering out polygons with the minimum area constraint does not necessarily ensure that the remaining polygons can accommodate a holding pattern with a minimum radius of R due to their arbitrary shapes. However, this heuristic step prunes polygons that are unlikely to fit the required holding pattern, thereby reducing online computational overhead. A significant aspect of $\mathcal{H}$ is that it can be processed offline and retrieved in real-time from the flight computer.

The online steps of holding point identification include a real-time optimization to find holding pattern center coordinates and utility calculation over candidate holding points. Let a circular left-turn holding pattern be identified with centering state $s_h \in \Delta p$ and $\Delta p \in \mathcal{H}$. A holding pattern is evaluated with a utility function, per Equation (61). The maximum values of these metrics are found over the set $\mathcal{H}$.

$$U(s_h) = \frac{\eta_{max} - \tilde{\eta}(c(s_h))}{\eta_{max}} + \frac{d_{max} - d_{gc}(s_h)}{d_{max}} + \frac{r_h(s_h)}{r_{max}} + \frac{g_{p,max} - g_p(s_h)}{g_{p,max}} \tag{61}$$

The coordinates of s_h, $c(s_h)$, are determined per Equation (62), which is a real-time optimization solved using [50] in this work.

$$c(s_h) = \arg\max \sum_{i=0}^{n} \|c(s_h) - e_i\| \tag{62}$$

The radius r_h refers to the largest inscribed circle in Δp Equation (63).

$$r_h(s_h) = \min_{e_i \in \Delta p} \|c(s_h) - e_i\| \tag{63}$$

The estimated overflown population risk from s_h to s_N on a straight path is specified by $g_p(s_h)$, per Equation (44). The best holding pattern, s_h^*, is identified per Equation (64).

$$s_h^* = \underset{s_h \in \Delta p, \Delta p \in \mathcal{H}}{\arg\max}\ U \tag{64}$$

This approach ensures an efficient and robust selection of low-risk holding points for emergency scenarios over urban areas. The holding inbound state is specified in Equation (65).

$$\begin{gathered} \chi_{h\leftarrow} = \theta(s_0, s_h^*), \quad c(s_{h\leftarrow}) = \mathcal{W}(R, |\chi_{h\leftarrow} + 0.5\pi|_{2\pi} \mid s_h^*) \\ h_{h\leftarrow} = h_0 - \delta h(\mathcal{P}_D(s_0, s_{h\leftarrow})) \end{gathered} \tag{65}$$

The function $\delta h : \mathbb{P} \to \mathbb{R}_{>0}$ yields the altitude loss along $\mathcal{P} \in \mathbb{P}$. Figure 8 visualizes the hold inbound state determination.

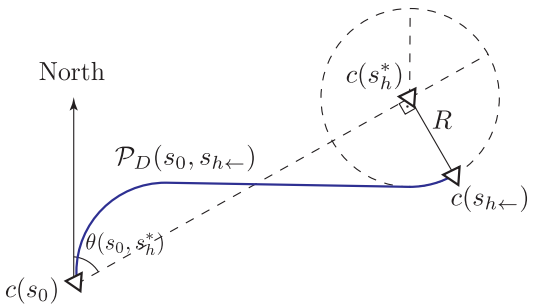

Figure 8. Holding inbound state determination.

First, the holding outbound state $s_{h\rightarrow}$ is built by determining its altitude, which is set while accounting for the maximum obstacle height $\bar{h}_o$ within $\mathcal{S}$, the optimal glide distance from s_h^* to s_N, the holding pattern floor altitude $\underline{h}_{h\rightarrow}$, and a buffer, h_{buffer}. The obstacle height data can be found from aeronautical charts.

$$h_{h\rightarrow} = \max\left(\bar{h}_o, d_{gc}(s_h^*)\tan\gamma_{opt}, \underline{h}_{h\rightarrow}\right) + h_{buffer} \tag{66}$$

The number of turns n_{360} and the net course change $\Delta\chi_{net}$ during holding are given in Equation (67), with $\Delta h_{360} = 2\pi R\tan\gamma_{opt,t}$ being the altitude loss in a 360° turn.

$$n_{360} = \frac{h_{h\leftarrow} - h_{h\rightarrow}}{\Delta h_{360}}, \quad \Delta\chi_{net} = 2\pi(n_{360} - \lfloor n_{360}\rfloor) \tag{67}$$

The holding outbound coordinate for a left-hand hold is

$$c(s_{h\rightarrow}) = \mathcal{W}(R, |\theta(s_h^*, s_{h\leftarrow}) - \Delta\chi_{net}|_{2\pi} \mid s_h^*) \tag{68}$$

The outbound course angle, the last element of $s_{h\rightarrow}$, is found from $|\chi_{h\leftarrow} + \Delta\chi_{net}|_{2\pi}$.

5. Real-Time Contingency Landing Planning Algorithms

This section presents search space discretization and contingency landing planning algorithms.

5.1. Search Space Discretization

Search space $\mathcal{S}$ is a continuous infinite set, rendering the search problem $\mathcal{T}$ computationally incomplete. A search over a continuous infinite set is not guaranteed to return a solution, $\mathcal{P}_s$. To address this issue, $\mathcal{S}$ can be discretized into $\mathcal{S}_d$ using uniform cells and reduced to a discrete finite set of states. Approaches for state-based discretization for search can be found in [51,52].

In this work, discretization is performed in the horizontal plane, altitude, and course angle dimensions. Hexagonal prism cells are used to represent the 3D position discrete state-space $s_d \in \mathcal{S}_d$, as illustrated in Figure 9. This choice ensures the efficient tiling of the space, guaranteeing the full coverage of $\mathcal{S}$ while providing an upper bound on the size of the discrete search space. However, it is important to note that each hexagonal prism does not correspond to a single state; rather, a single prism represents multiple states, with each associated with a different discretized course angle. Each continuous state, s, within a discrete cell is mapped to a single unique discrete state, s_d, which is defined as the centroid of the cell.

An upper bound to the total number of states can be estimated by considering an optimal packing of hexagons into a circle, an approximate footprint. The number of layers of hexagonal cells required to cover the footprint with a radius of $r_{\mathcal{F}}$, per Equation (69), where the circumradius of cells is $\ell/2$.

$$l = \left\lceil \frac{r_{\mathcal{F}} - \ell\sqrt{3}/4}{\ell\sqrt{3}/2} \right\rceil \tag{69}$$

The total number of latitude–longitude pairs needed to completely cover the footprint is as follows:

$$n_{(\varphi,\lambda)} = 1 + 3l(l+1) \tag{70}$$

The number of discrete course angles and altitude ranges is

$$n_\psi = \left\lceil \frac{2\pi}{\Delta\psi} \right\rceil, \quad n_h = \left\lceil \frac{h_0 - h_N}{\Delta h_{360}} \right\rceil \tag{71}$$

Therefore, an upper bound to the total number of states is given per Equation (72).

$$n_s = n_{(\varphi,\lambda)} \times n_\psi \times n_h \tag{72}$$

As a finite search space is obtained, a formal proof for guaranteed convergence can be given.

Theorem 3. *Given a finite discrete search space, $\mathcal{S}_d$, a defined solution identification set, $\mathcal{Z} \subset S_d$, and assured reachability between arbitrary states $s_i \in \mathcal{S}_d$ and $s_j \in \mathcal{S}_d$, the discrete gradient-guided search from s_i to s_j is guaranteed to converge to a final state, $s'_N \in \mathcal{Z}$.*

Proof of Theorem 3. The state-space $\mathcal{S}$ is discretized into a finite set of states with $|\mathcal{S}_d| = n_s < \infty$, ensuring that the search process operates over a well-defined and bounded space. A solution from s_i to s_j exists, given the reachability assurance.

$$\exists \mathcal{P}(s_i, s_j) \subset \mathcal{S}_d \tag{73}$$

Given that the search space is finite and a feasible solution exists, the algorithm is guaranteed to terminate in a discrete feasible state within $\mathcal{Z}$.

$$\lim_{k \to n_s} s_k = s'_N, \quad s'_N \in \mathcal{Z} \tag{74}$$

Thus, the discrete gradient-guided search is complete, given the conditions. □

Figure 9 illustrates an example of $\mathcal{S}_d$ for underestimated best-glide ranges with $n_h = 3$ and $n_\psi = 1$. These values are intentionally chosen for visualization purposes and do not reflect the actual space discretization used in the algorithm. The red, green, and blue cells represent increasing altitudes, with $n_{(\varphi,\lambda)}$ values being 3, 4, and 5, respectively.

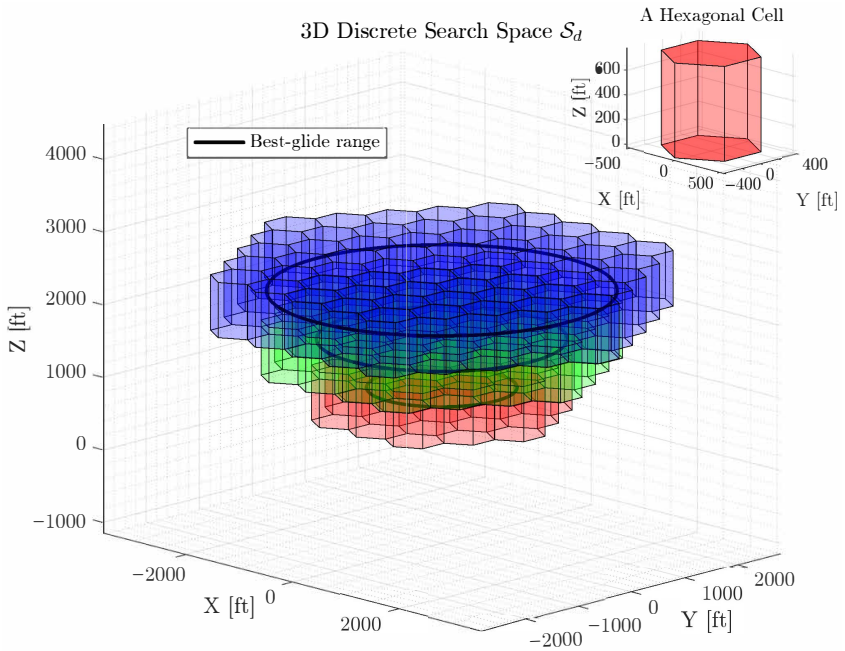

Figure 9. An illustration of a discrete search space in 3D.

Table 1 presents upper bounds on the search space size for an engine-out Cessna 182, considering various altitude differences and segment lengths. As shown, increasing the altitude difference expands the search space, while the longer segment lengths reduce the number of discrete cells.

Table 1. Upper bounds of the discrete search space size for the Cessna 182 experiencing loss of thrust.

$h_0 - h_N$ [ft]	n_s for Different Circumradii in ft		
	500	**1000**	**1500**
500	2.88×10^4	7.39×10^4	2.85×10^3
2500	3.30×10^6	8.56×10^5	3.60×10^5
5000	26.2×10^6	8.40×10^6	4.11×10^6

It should be noted that this structured grid is not directly used for search expansions. The search algorithm expands states based on actions defined in the previous section, and the resulting child states do not necessarily align with the hexagonal cell centers. Instead, each expanded state is assigned as the centroid of a newly generated hexagonal cell at an arbitrary position. Once a cell is explored, its corresponding centroid is stored in a closed list to track visited states and prevent redundant expansions using a point-in-polygon algorithm.

5.2. Contingency Landing Planner

This section describes a contingency landing planning framework consisting of three sequential stages. The first stage constructs an initial set of feasible landing paths using a 3D Dubins path planner formulated with Equation (9) to provide a geometric solution and holding pattern identification governed by Equations (60)–(68) to ensure safe altitude management. This first stage also performs a gradient-guided search based on standard tree search [53,54] but with a key distinction: it detects repeated discrete states using a geometrical point-in-polygon algorithm, rather than state hashing.

In the second stage, Algorithm 1 implements the handling of different altitude cases by conditionally utilizing the first-stage algorithms. Contingent on the initial emergency altitude, a direct search-based path, $\mathcal{P}_s$, is computed, followed by a Dubins connection to the landing site. If the UAS has sufficient altitude, a holding pattern is first executed to dissipate the altitude before transitioning to the final approach descent via successive search- and Dubins-based trajectories.

The final stage executes the contingency landing planner in Algorithm 2, which runs two planners in parallel: (1) the altitude-dependent search-based path planner to define the risk-aware path and (2) a Dubins-based shortest path solver, $\mathcal{P}_D$. Since Dubins is a geometrical solution, it enables a real-time response with onboard computing resources for execution as a fallback solution in the case where the search-based path cannot be computed within time limit $T_{max} \in \mathbb{R}_{>0}$.

As a result, this framework guarantees that an emergency landing solution can be computed within a prescribed time limit. As described below, the search-based planner typically identifies a solution in time that improves safety using our cost metrics relative to Dubins.

Algorithm 1 Altitude-dependentpath planning.

Require: $\mathbb{A}, \mathcal{A}, g(s), s_0, s_N, s_t, v_w, \chi_w$
Ensure: $\mathcal{P}$
  if $h_0 \leq h_c$ **then**
    $\mathcal{P}_s(s_0, s'_N) \leftarrow \mathcal{T}(\mathbb{A}, \mathcal{A}, g(s), s_0, s_N, v_w, \chi_w)$
    $\mathcal{P}_D(s'_N, s_t) \leftarrow \arg\min_{\mathcal{P} \in \mathcal{D}(s'_N, s_t)} \mathcal{L}(\mathcal{P})$
    $\mathcal{P} = \mathcal{P}_s \cup \mathcal{P}_D$
  else
    $s_h^* \leftarrow \arg\max_{s_h \in \mathcal{H}} U$
    $s_{h\leftarrow} \leftarrow$ Equation (65)
    $\mathcal{P}_D(s_0, s_{h\leftarrow}) \leftarrow \arg\min_{\mathcal{P} \in \mathcal{D}(s_0, s_{h\leftarrow})} \mathcal{L}(\mathcal{P})$
    if $h_{h\leftarrow} > h_c$ **then**
      $s_{h\rightarrow} \leftarrow$ Equations (66)–(68)
      $\mathcal{P}_D(s'_N, s_t) \leftarrow \arg\min_{\mathcal{P} \in \mathcal{D}(s'_N, s_t)} \mathcal{L}(\mathcal{P})$
      $\mathcal{P}_s(s_{h\rightarrow}, s'_N) \leftarrow \mathcal{T}(\mathbb{A}, \mathcal{A}, g(s), s_{h\rightarrow}, s_N, v_w, \chi_w)$
      $\mathcal{P}_D(s'_N, s_t) \leftarrow \arg\min_{\mathcal{P} \in \mathcal{D}(s'_N, s_t)} \mathcal{L}(\mathcal{P})$
      $\mathcal{P} \leftarrow \mathcal{P}_D(s_0, s_{h\leftarrow}) \cup \mathcal{P}_s(s_{h\rightarrow}, s'_N) \cup \mathcal{P}_D(s'_N, s_t)$
    else
      $s_c \leftarrow \arg\min_{s_c \in \mathcal{P}_D(s_0, s_{h\leftarrow})} |h - h_c|$
      $\mathcal{P}_s(s_c, s'_N) \leftarrow \mathcal{T}(\mathbb{A}, \mathcal{A}, g(s), s_c, s_N, v_w, \chi_w)$
      $\mathcal{P}_D(s'_N, s_t) \leftarrow \arg\min_{\mathcal{P} \in \mathcal{D}(s'_N, s_t)} \mathcal{L}(\mathcal{P})$
      $\mathcal{P} \leftarrow \mathcal{P}_D(s_0, s_c) \cup \mathcal{P}_s(s_c, s'_N) \cup \mathcal{P}_D(s'_N, s_t)$
    end if
  end if
  return $\mathcal{P}$

Algorithm 2 Contingency landing path planner.

Require: $\mathbb{A}, \mathcal{A}, g(s), s_0, s_N, s_t, v_w, \chi_w, T_{\max}$
Ensure: $\mathcal{P}$ or fallback path P_D
Initialize runtime $t \leftarrow 0$
Initialize path set $\mathcal{P} \leftarrow \varnothing$
Initialize status flag flag $\leftarrow$ False
while $t \leq T_{\max}$ **and** $\neg$flag **do**
  Parallel Execution:
    Run Algorithm 1 on Processor-1
    Solve Equations (8) and (9) for $P_D(s_0, s_N)$ on Processor-2
  if $\mathcal{P} \neq \varnothing$ **then**
    flag $\leftarrow$ True
  end if
  $t \leftarrow t + \Delta t$
end while
if flag **then**
  return $\mathcal{P}$ ▷ Return the minimum-risk path
else
  return $\mathcal{P}_D(s_0, s_N)$ ▷ Fallback to default path
end if

6. Use Cases and Algorithm Benchmarking

This section presents use cases demonstrating the capabilities of the contingency landing planner under varying altitude conditions and steady wind. Descriptive statistics are also provided on the performance of the proposed method in different initial emergency states over a uniform grid. A populated residential area, Farmingdale, Long Island, NY, is chosen to simulate emergency landing scenarios for a large cargo drone operating in urban delivery missions. Therefore, all use cases and tests share the same goal state: a 1 NM

final approach fix at 514 ft mean sea level (MSL) altitude to Runway 14 at Republic Airport (KFRG) in Long Island, New York. Table 2 lists user-defined parameters for path planning.

Table 2. Parameters for altitude-dependent path planning.

Parameter	**Value**	**Unit**
R	3000	ft
h_c	5000	ft
$\mathcal{X}$	$\{-30, -15, 0, 15, 30\}$	°
$\underline{h}, m$	2000, 0.1	ft, -
$\underline{r}$	0.5	NM
R_i, R_o	3000, 6000	ft, ft
$\underline{h}_{h\rightarrow}$	1000	ft
h_{buffer}	2000	ft
$\Delta\psi$	5	°
Weight coefficients as a function of g_χ		
Condition on g_χ	Cost Weights	
$g_\chi \geq 0.5$	$(w_1, w_2, w_3, w_4) = (0.2, 0.3, 0.2, 0.3)$	
$10^{-4} < g_\chi < 0.5$	$(w_1, w_2, w_3, w_4) = (0.2, 0.3, 0.1, 0.4)$	
$g_\chi \leq 10^{-4}$	$(w_1, w_2, w_3, w_4) = (0.2, 0.4, 0.0, 0.4)$	

Weight coefficients are used to compute $g(s)$ Equation (23) as a function of $g_\chi(s)$. A state with a high g_χ requires a final approach turn greater than 90°. To mitigate this, state expansion toward regions where the final course change remains below 90° is prioritized. As g_χ decreases, w_3 is gradually reduced, thereby shifting the cost emphasis toward the overflown population and distance to the goal. These coefficients are manually fine-tuned through simulations, while a machine learning algorithm can be leveraged for a more streamlined approach.

6.1. Altitude-Dependent Path Planning

The impact of initial emergency altitude on contingency landing solutions is illustrated in Figure 10, where an emergency scenario is simulated at a randomly selected location within the footprint, heading to the east, at altitudes of 4000 ft, 6000 ft, and 10,000 ft MSL.

At a low emergency onset altitude of 4000 ft MSL, Algorithm 1 directly computes the search-based path $\mathcal{P}_s(s_0, s'_N)$ and a 3D Dubins path for the final approach to Runway 14 at KFRG, as the initial altitude h_0 is below the crossover altitude h_c. The trajectory effectively avoids residential areas by following roads, leveraging the available altitude for risk mitigation. This path is computed with as few as 3709 explored states, dramatically below the upper bound of the search space size of 8.44×10^5 for this case.

At a high emergency altitude of 10,000 ft MSL, altitude dissipation is necessary before the approach. A holding pattern is established over a sparsely populated region while maintaining a 0.5 NM no-fly zone around Runway 14 to prevent conflicts with the airport traffic pattern. A 3D Dubins path is then generated from s_0 to $s_{h\leftarrow}$ without considering population risk, as exposure risk at high altitude is minimal due to $h_0 > h_c$. The distressed UAS descends from $h_{h\leftarrow} = 7923$ ft to $h_{h\rightarrow} = 3000$ ft over 3.28 full left-hand holding turns. Despite this descent, excess altitude remains as the aircraft approaches s_N. The gradient-guided search subsequently determines a descent path to s'_N that avoids densely populated areas, exploring a total of 320 states. Once the search-based path is generated, a 3D Dubins path is planned from s'_N to s_t.

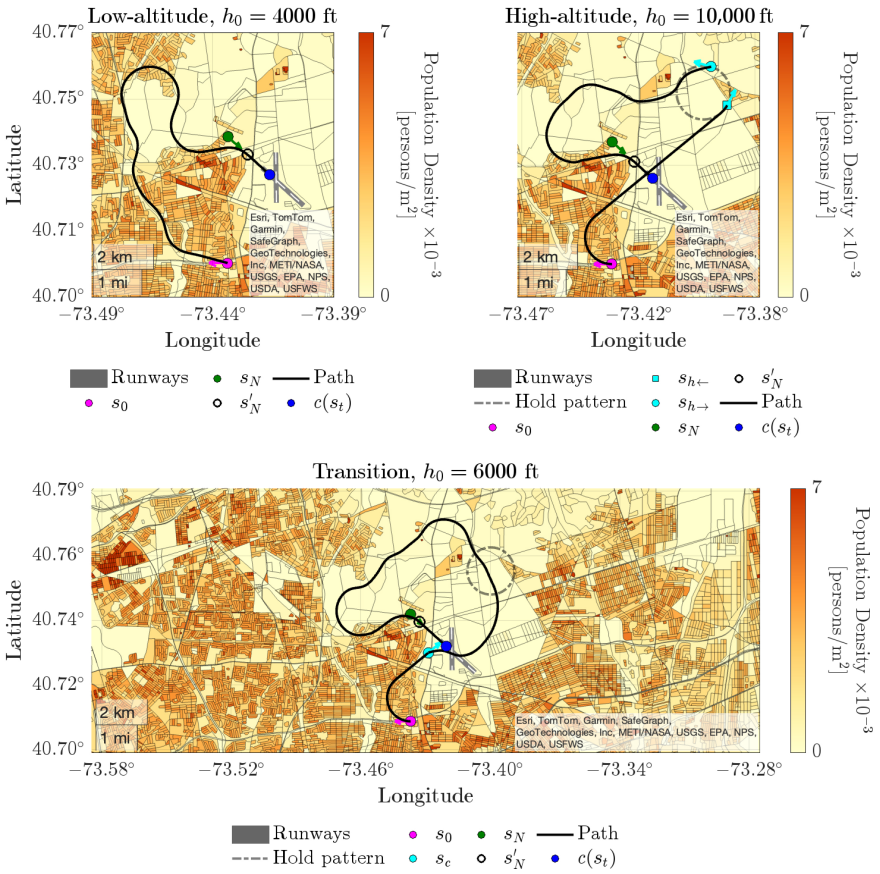

Figure 10. Contingency landing solutions for varying initial altitudes. Arrows show the course angles.

At an initial emergency altitude of 6000 ft, a holding pattern is also identified. However, unlike the 10,000 ft case, the landing trajectory deviates from the initial Dubins path upon reaching the crossover altitude h_c at s_c, as the hold inbound cannot be reached above this altitude. From the crossover state s_c, gradient-guided search is initialized. Similar to the example in Figure 4, $g_{d,1}(s)$ leads the minimum cost states away from s_N due to the aircraft's higher altitude relative to the remaining horizontal traversal, causing state expansion away from s_N. As altitude decreases, $g_{d,1}(s)$ shifts the minimum-cost states closer to s_N, gradually directing state expansion toward s_N. Compared to the low- and high-altitude cases, the landing path found via the gradient-guided search is significantly longer due to the higher crossover altitude; however, $\mathcal{P}s(s_c, s'_N)$ is computed with a total of 177 explored states, whereas $n_s = 1.2 \times 10^6$. As with the previous cases, the final segment is completed with a 3D Dubins path to s_t. The complete 3D contingency landing solution, along with the explored states for the transition case, is shown in Figure 11. The low number of state expansions demonstrates the efficiency of the presented search-based landing planner.

The dynamic feasibility of the low-altitude emergency landing plan is validated using a six-degree-of-freedom engine-out Cessna 182 model [17]. The simulation environment employs a conventional proportional–integral–derivative control system, a fly-by waypoint following for executing the search-based path, and a Dubins guidance manager to achieve final runway alignment. As shown in Figure 12, the lateral tracking error between the planned and simulated trajectories remains minimal, ensuring close alignment between the planned overflown population risk versus simulated overflown population risk. Figure 13 presents the time histories of the flight path angle and airspeed, highlighting constraints in red. Both quantities remain well within the prescribed operational limits. In addition,

the maximum and minimum load factors are recorded below 1.1 G and above 0.85 G, respectively, confirming safe flight conditions. Since the simulated landing trajectory encompasses all five action types used in the planning framework, the feasibility of all search-generated sequences is confirmed.

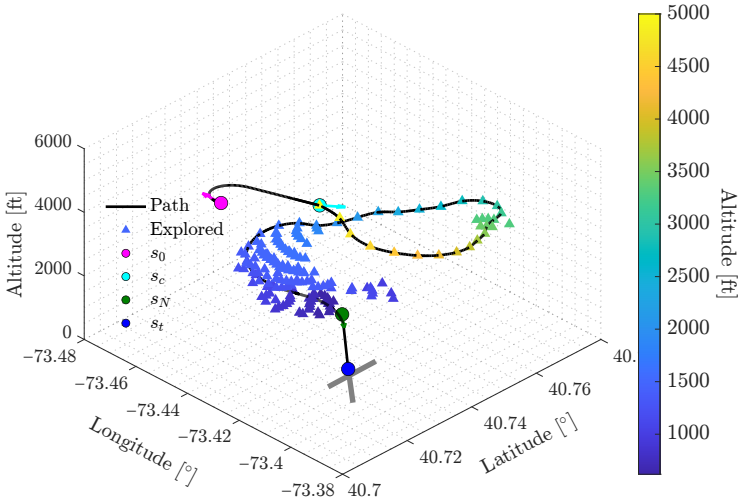

Figure 11. The 3D contingency landing trajectory and explored state coordinates for the transition case, initially at 6000 ft MSL. Arrows indicate the course angles.

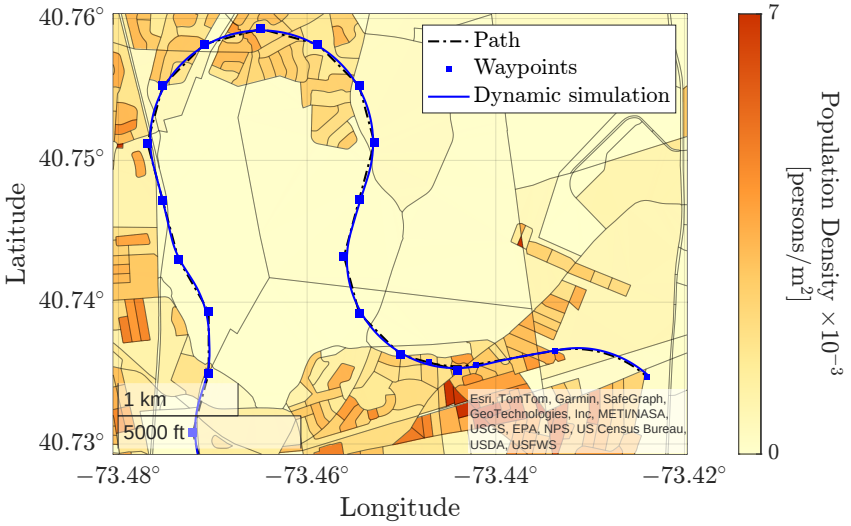

Figure 12. Dynamic simulation: a magnified view comparing the plan and simulation trajectory of the low-altitude case.

6.2. Contingency Landing Planning Under Steady Wind

With the recollection of Equations (5) and (14), a feasible action depends on wind conditions. A nonzero wind vector alters both the direction and magnitude of the ground speed vector, which in turn affects the required flight path angle to maintain a given airspeed. For path planning, the flight path angle for each segment is selected from a dataset derived from Equations (1)–(4), taking into account wind speed and relative direction.

To illustrate the effect of wind on contingency landing planning, two trajectories are compared in Figure 14.

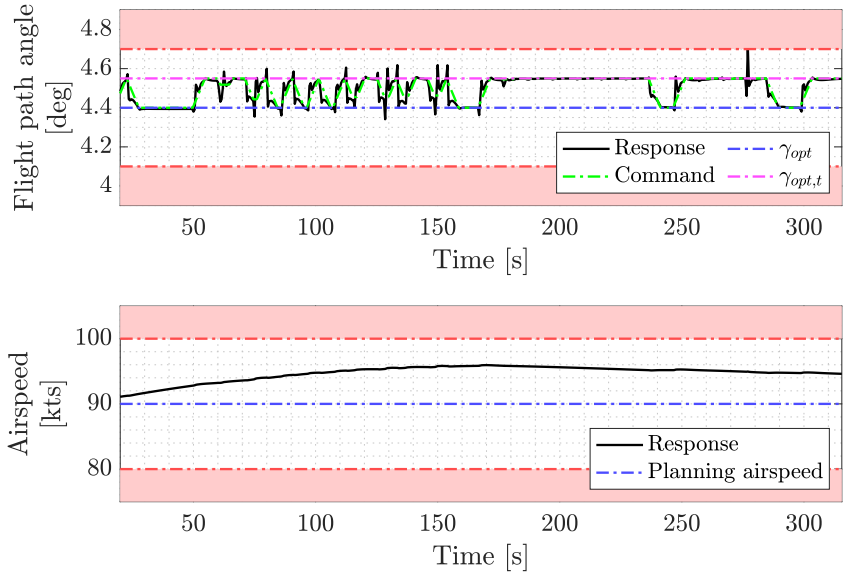

Figure 13. Dynamic simulation: flight path angle and airspeed of an engine-out Cessna 182 in time for the low-altitude case. Shaded areas indicate regions outside the degraded flight envelope.

In this emergency scenario, the aircraft begins at 4000 ft MSL with a 10 knots south wind ($v_w = 10$ knots, $\chi_w = 180°$). In windless conditions, the planned trajectory spans 7.33 NM at a constant optimal flight path angle of 4.4°. However, with a partial tailwind, the feasible flight path angle range becomes shallower with an average optimal flight path angle of 4.18°, leading to an extended trajectory of 7.73 NM.

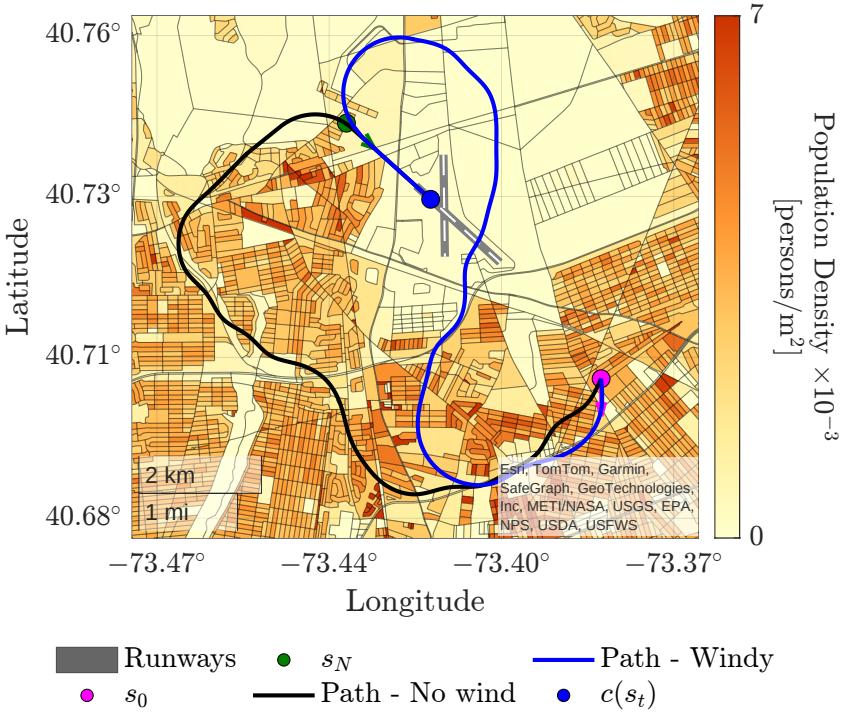

Figure 14. The impact of tailwind in the contingency landing solution. Arrows show the course angles.

6.3. Algorithm Benchmarking on a Uniform Grid

A benchmark study is conducted to evaluate the developed algorithm's performance in terms of overflown population risk, optimal flight path angle, and computational overhead. The flight time comparison is refrained, as it conflicts with the optimal descent. For contingency landing solutions, the implemented framework is compared against 3D and

S-turn Dubins paths. To determine the best Dubins-based solution for low-altitude cases, all possible Dubins path types are evaluated, and the one that minimizes the overflown population risk is selected. Specifically, for lower altitudes, four types of 3D Dubins paths are generated. If additional altitude dissipation is required, S-turn Dubins paths, extending in both directions, are computed, and the one with minimum risk is chosen for comparison.

Test scenarios are distributed across a uniform grid with nine coordinates, three altitude levels, and four cardinal directions, totaling 108 unique cases. All initial states are ensured to be reachable from the goal. Algorithm 1 is compared against the minimum-risk Dubins solutions, with results summarized in Table 3, including descriptive statistics on overflown population risk.

Table 3. Risk and gliding constraint comparison of the gradient-guided search and the minimum-risk 3D Dubins solutions.

h_0 [ft]	Method	Population Risks				
		Min.	**Max.**	**Median**	μ	σ
4000	Search	0.0043	0.1658	0.0223	0.0412	0.0393
	Dubins	0.0074	0.1624	0.0446	0.0593	0.0454
6000	Search	0.0058	0.0390	0.0153	0.0162	0.0077
	Dubins	0.0058	0.1360	0.0433	0.0491	0.0283
10,000	Search	0.0085	0.0594	0.0239	0.0254	0.0124
	Dubins	0.0050	0.0835	0.0244	0.0278	0.0164

The search-based method consistently outperforms the Dubins-based approach in minimizing overflown population risk across altitude levels. This is expected as the construction of Dubins paths does not offer explicit risk mitigation. Additionally, at sufficiently high initial altitudes, both methods strictly follow the optimal flight path angle. In contrast, at lower altitudes, search solutions maintain this angle without deviation, whereas flight path angles for the four Dubins solutions are flattened (closer to level flight) by an average of 0.21°. However, these shallower paths are considered, as they offer minimum-risk solutions.

The framework, along with the benchmarking Dubins path solvers, is programmed in C, with the exception of R*-tree, whose source codes are written in C++ [55]. Table 4 provides a detailed runtime analysis benchmarking search-based path planning against the minimum-risk Dubins solutions. For higher-altitude scenarios, the total elapsed time for optimal holding point identification and the subsequent Dubins path solution to the hold pattern, labeled as hold planning, are also tabulated. Despite being slower than Dubins solutions, as would be anticipated, the developed contingency landing planner exhibits strong potential for real-time implementation. In addition, the separate computation times, $g_p(s)$, Equation (44), are provided to highlight its significant contribution to the total computational overhead. As is evident, a high percentage of the total runtime is attributed to the overflown population risk estimation despite an efficient R*-tree implementation. This overhead could be reduced, albeit with a trade-off in accuracy due to interpolation, by reducing the integration time step or employing data projection methods onto uniform grids.

Table 4. Runtime comparison of the gradient-guided search and the minimum-risk 3D Dubins solutions with a C language code implementation.

h_0 [ft]	Method	Total Runtime [ms]					Risk (g_p) Computation Runtime [ms]				
		Min.	Max.	Median	μ	σ	Min.	Max.	Median	μ	σ
4000	Search	78.4	4037.7	466.3	718.9	839.9	77.0	3239.8	428.8	631.1	680.7
	Dubins	33.3	100.4	53.3	57.9	17.4	4.9	49.4	32.9	33.1	10.5
6000	Hold Planning	83.6	92.7	87.4	87.3	1.8	1.3	6.2	5.1	4.4	1.3
	Search	99.3	3891.3	598.3	852.3	869.9	94.0	3173.2	547.9	741.6	719.2
	Dubins	46.2	92.9	66.0	65.3	10.5	6.8	64.3	53.3	48.7	12.8
10,000	Hold Planning	83.6	98.9	88.5	88.5	2.6	1.3	6.1	5.2	4.5	1.3
	Search	49.4	1933.1	567.6	747.2	541.5	44.2	1727.5	498.5	662.7	473.4
	Dubins	16.6	96.7	50.7	48.5	21.5	3.4	21.0	10.3	9.8	4.2

All experiments were conducted on a single performance core of anApple M2 processor (Apple Inc., Cupertino, CA, USA) (ARM64, 3.49 GHz) using Apple Clang 17.0.0 (clang-1700.0.13.5) as the backend compiler.

Table 5 presents the solution availability of Algorithm 2 across 108 test cases for different solver time limits. The planner provides search-based minimum-risk landing solutions in 75 out of 108 scenarios within the solver time limit of $T_{max} = 1$ s. When the solver time limit is increased to $T_{max} = 2$ s, search-based landing solutions are found in 102 out of 108 cases. Only four scenarios require fallback Dubins solution when the maximum time limit, $T_{\max} = 3$ s is applied.

Table 5. Algorithm 2 solution availability for different time limits.

$T_{\max}$ [s]	h_0 [ft]			Number of Solutions	
	4000	6000	10,000	Search-Based ($\mathcal{P}$)	Fallback Dubins ($\mathcal{P}_D$)
1	28	22	25	75	33
2	34	33	35	102	6
3	34	34	36	104	4

7. Discussion

This paper has presented a contingency landing planning framework for a large cargo UAS with wing-lift capability. The proposed approach is adaptable to other emergency scenarios, including but not limited to control surface failures and structural damage by defining the set of feasible actions based on the aircraft's degraded performance envelope. In addition, the turn radius and flight path angle constraints are applicable to any aircraft capable of wing-lift flight, including hybrid wing-lift configurations. Even multicopters operating with reduced but non-zero thrust could benefit from the proposed approach for risk avoidance. Therefore, the presented method is transferable and can be applied to a wide variety of UAS and crewed aircraft configurations with distinct aerodynamic characteristics.

To estimate the overflown population risk, publicly available population census data was utilized. To improve risk quantification, an interquartile rule for outlier detection was applied. These values were then adjusted to the third-quartile population density, ensuring a more representative risk assessment. Alternative population models might be created and updated with real-time data, e.g., from cellular network call data. Moreover, incorporating obstacle avoidance, real-time weather updates, and air traffic separation is crucial for holistic risk estimation and avoidance.

In general, the number of explored states in the presented use cases remains low. However, it is observed that, when the initial emergency state is located in a densely populated area with a high initial course angle cost, state expansion increases as the

algorithm searches for a low-cost trajectory, leading to longer solution times. Additionally, a greater altitude difference between the initial and goal states requires a longer landing path, further increasing state expansion. The gradient-guided search utilizes course angle changes to navigate through narrow, low-risk passages. This is not a concern for UASs with attitude control authority and modern autopilot systems. However, these solutions can only reasonably be shared with unmanned aircraft traffic management systems, air traffic control, and other nearby air traffic via datalink to assure coordinated separation.

8. Conclusions

This paper has presented a real-time emergency landing planner that combines a novel gradient-guided discrete search with 3D Dubins paths to generate safe and risk-aware landing trajectories for wing-lift drone. To further minimize high-altitude risk, a minimum-risk holding pattern identification algorithm was also introduced. A series of 108 Long Island, New York, case studies demonstrated that the presented method significantly reduces risk compared to standard minimum-risk Dubins path solvers. While computationally more demanding, the search-based planner achieved real-time performance, generating minimum-risk landing trajectories within three seconds in 104 out of 108 cases. Future work must consider other air traffic in contingency flight planning solutions to minimize the risk associated with disrupting or losing separation within the low-altitude urban airspace. Such solutions will require awareness of commercial and Advanced Air Mobility airspace corridors and reserved geofence volumes for UAS traffic. To address this, airspace and air traffic avoidance risks will be captured in contingency landing planning by introducing new cost terms within the gradient-guided search framework.

Author Contributions: Conceptualization, H.E.T. and E.M.A.; methodology, H.E.T. and E.M.A.; software, H.E.T.; validation, E.M.A.; formal analysis, H.E.T. and E.M.A.; investigation, H.E.T. and E.M.A.; resources, E.M.A.; data curation, H.E.T. and E.M.A.; writing—original draft preparation, H.E.T. and E.M.A.; writing—review and editing, E.M.A.; visualization, H.E.T. and E.M.A.; supervision, E.M.A.; project administration, E.M.A.; funding acquisition, E.M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Langley Research Center grant number 80LARC23DA003.

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Nomenclature

$\mathbb{A}$	Aircraft flight envelope
$\mathcal{A}$	Feasible action set
$\mathcal{D}$	Dubins path solution set
D	Drag force
L, L_{max}	Lift force and maximum lift force
$\mathcal{P}$	Landing path
$\mathcal{P}_D$, $\mathcal{P}_s$	Shortest Dubins and search-based paths
R	Aircraft turn radius
R_i, R_o	Annulus radii of the solution identification set
$\mathcal{S}$	Aircraft state set
$\underline{S}$, $\bar{S}$	Lower and upper sums in the context of integration
T	Thrust
W	Total weight
$\mathcal{Z}$	Solution identification set

a	Feasible action
d_{bg}, d_{opt}	Best-glide and optimal gliding traversals
g	Gravitational acceleration
h	Altitude in mean sea level
h_0, h_N, h_c	Initial, goal, and risk crossover altitudes
$\bar{h}_o, h_{buffer}$	Maximum vertical obstacle height and a buffer altitude
$\underline{h}$	Floor altitude for adaptive segment length
$\underline{h}_{h\rightarrow}$	Holding pattern floor altitude
ℓ	Segment length
ℓ_{min}, ℓ_{turn}	Minimum feasible segment length and effective arc length
m	Segment length change rate per altitude
n_{360}	Number of full turns in a holding pattern
n_s	Upper bound to the total number of states
$\vec{n}_{\chi_N}$	Unit normal vector of the goal state
$\underline{r}, \bar{r}$	Lower and upper bound radii around the goal state
s	Aircraft state
s_0, s_N, s_t	Initial, goal, and touchdown states
s'_N	Final search state of a converged path
s^*_h, $s_{h\leftarrow}, s_{h\rightarrow}$	Optimal hold, hold inbound and outbound states
t	Time
$\mathbf{u}$	Aircraft control input vector
v_a, v_g	Airspeed and ground speed
v_{FE}	Maximum flap extended speed
v_w	Wind speed
w_i	Cost function weighting coefficients
$\mathbf{x}$, $\dot{\mathbf{x}}$	Aircraft dynamic state vector and its derivative
χ	Course angle
χ_w	Wind direction
$\Delta\chi$	Course angle change
$\gamma_{bg}, \gamma_{bg,t}$	Best-glide flight path angle for forward and turning flight
$\gamma_{max}, \gamma_{max,t}$	Steepest flight path angle for forward and turning flight
$\gamma_{opt}, \gamma_{opt,t}$	Optimal flight path angle for forward and turning flight
$\kappa_\chi, \kappa_w, \kappa_\eta$	Cost normalizers
λ	Longitude
Λ	Valid longitude set
μ	Bank angle
Φ	Valid latitude set
φ	Latitude

References

1. Bacchini, A.; Cestino, E.; Van Magill, B.; Verstraete, D. Impact of lift propeller drag on the performance of eVTOL lift+ cruise aircraft. *Aerosp. Sci. Technol.* **2021**, *109*, 106429. [CrossRef]
2. Mathur, A.; Atkins, E. Multi-Mode Flight Simulation and Energy-Aware Coverage Path Planning for a Lift+Cruise QuadPlane. *Drones* **2025**, *9*, 287. [CrossRef]
3. Kim, J.; Atkins, E. Airspace Geofencing and Flight Planning for Low-Altitude, Urban, Small Unmanned Aircraft Systems. *Appl. Sci.* **2022**, *12*, 576. [CrossRef]
4. Russell, L.; Goubran, R.; Kwamena, F. Emerging Urban Challenge: RPAS/UAVs in Cities. In Proceedings of the 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini Island, Greece, 29–31 May 2019; pp. 546–553. [CrossRef]
5. Wu, Y.; Low, K.H. An Adaptive Path Replanning Method for Coordinated Operations of Drone in Dynamic Urban Environments. *IEEE Syst. J.* **2021**, *15*, 4600–4611. [CrossRef]
6. Dubins, L.E. On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents. *Am. J. Math.* **1957**, *79*, 497–516. [CrossRef]

7. Yomchinda, T.; Horn, J.F.; Langelaan, J.W. Modified Dubins parameterization for aircraft emergency trajectory planning. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2017**, *231*, 374–393. [CrossRef]
8. Chitsaz, H.; LaValle, S.M. Time-optimal paths for a Dubins airplane. In Proceedings of the IEEE CDC, New Orleans, LA, USA, 12–14 December 2007; pp. 2379–2384. [CrossRef]
9. Ma, D.; Hao, S.; Ma, W.; Zheng, H.; Xu, X. An optimal control-based path planning method for unmanned surface vehicles in complex environments. *Ocean. Eng.* **2022**, *245*, 110532. [CrossRef]
10. Bergman, K.; Ljungqvist, O.; Axehill, D. Improved Path Planning by Tightly Combining Lattice-Based Path Planning and Optimal Control. *IEEE Trans. Intell. Veh.* **2021**, *6*, 57–66. [CrossRef]
11. Liu, J.; Han, W.; Liu, C.; Peng, H. A New Method for the Optimal Control Problem of Path Planning for Unmanned Ground Systems. *IEEE Access* **2018**, *6*, 33251–33260. [CrossRef]
12. Zhang, K.; Sprinkle, J.; Sanfelice, R.G. A hybrid model predictive controller for path planning and path following. In Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems, Seattle, WA, USA, 13–16 April 2015; pp. 139–148. [CrossRef]
13. Shen, C.; Shi, Y.; Buckham, B. Model predictive control for an AUV with dynamic path planning. In Proceedings of the 2015 54th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Hangzhou, China, 28–30 July 2015; pp. 475–480. [CrossRef]
14. Akametalu, A.K.; Tomlin, C.J.; Chen, M. Reachability-Based Forced Landing System. *J. Guid. Control. Dyn.* **2018**, *41*, 2529–2542. [CrossRef]
15. Berksetas, D.P. *Dynamic Programming and Optimal Control*, 3rd ed.; Athena Scientific, Belmont, MA, USA, 2005; Volume 1.
16. Donato, P. Toward Autonomous Aircraft Emergency Landing Planning. Ph.D. Thesis, University of Michigan, Ann Arbor, MI, USA, 2017.
17. Tekaslan, H.E.; Atkins, E.M. Vehicle-to-Vehicle Approach to Assured Aircraft Emergency Road Landings. *J. Guid. Control. Dyn.* **2025**, *48*, 1800–1817. [CrossRef]
18. Howlett, J.K.; McLain, T.W.; Goodrich, M.A. Learning Real-Time A* Path Planner for Unmanned Air Vehicle Target Sensing. *J. Aerosp. Comput. Inf. Commun.* **2006**, *3*, 108–122. [CrossRef]
19. Qian, Y.; Sheng, K.; Ma, C.; Li, J.; Ding, M.; Hassan, M. Path Planning for the Dynamic UAV-Aided Wireless Systems Using Monte Carlo Tree Search. *IEEE Trans. Veh. Technol.* **2022**, *71*, 6716–6721. [CrossRef]
20. Chour, K.; Pradeep, P.; Munishkin, A.A.; Kalyanam, K.M. Aerial Vehicle Routing and Scheduling for UAS Traffic Management: A Hybrid Monte Carlo Tree Search Approach. In Proceedings of the 2023 IEEE/AIAA Conference on Digital Avionics Systems, Barcelona, Spain, 1–5 October 2023; pp. 1–9. [CrossRef]
21. Guo, Y.; Liu, X.; Jia, Q.; Liu, X.; Zhang, W. HPO-RRT*: A sampling-based algorithm for UAV real-time path planning in a dynamic environment. *Complex Intell. Syst.* **2023**, *9*, 7133–7153. [CrossRef]
22. Xu, D.; Qian, H.; Zhang, S. An Improved RRT*-Based Real-Time Path Planning Algorithm for UAV. In Proceedings of the IEEE International Conference on High Performance Computing & Communications, Haikou, China, 20–22 December 2021; pp. 883–888. [CrossRef]
23. Kothari, M.; Postlethwaite, I.; Gu, D.W. Multi-UAV path planning in obstacle rich environments using Rapidly-exploring Random Trees. In Proceedings of the IEEE Conference on Decision and Control, Shanghai, China, 15–18 December 2009; pp. 3069–3074. [CrossRef]
24. Sláma, J.; Herynek, J.; Faigl, J. Risk-Aware Emergency Landing Planning for Gliding Aircraft Model in Urban Environments. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems, Detroit, MI, USA, 1–5 October 2023; pp. 4820–4826. [CrossRef]
25. Meuleau, N.; Plaunt, C.; Smith, D.; Smith, T. A POMDP for Optimal Motion Planning with Uncertain Dynamics. In Proceedings of the ICAPS-10: POMDP Practitioners Workshop, Toronto, ON, Canada, 12 May 2010; ICAPS: CA, USA, 2010. Available online: https://www.icaps-conference.org/icaps-executive-council/ (accessed on 24 April 2025).
26. Sharma, P.; Kraske, B.; Kim, J.; Laouar, Z.; Sunberg, Z.; Atkins, E. Risk-Aware Markov Decision Process Contingency Management Autonomy for Uncrewed Aircraft Systems. *AIAA J. Aerosp. Inf. Syst.* **2024**, *21*, 234–248. [CrossRef]
27. Castagno, J.; Atkins, E. Map-based planning for small unmanned aircraft rooftop landing. In *Handbook of Reinforcement Learning and Control*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 613–646.
28. Daniel, K.; Nash, A.; Koenig, S.; Felner, A. Theta*: Any-Angle Path Planning on Grids. *J. Artifical Intell. Res.* **2010**, *39*, 533–579. [CrossRef]
29. Akinola, A.A.; Atkins, E.M. Markov Decision Process Contingency Landing Management Autonomy for Advanced Air Mobility. In Proceedings of the AIAA Aviation Forum and Ascend, Las Vegas, NV, USA, 22–24 July 2025; p. 3399.
30. Beard, R.W.; McLain, T.W. *Small Unmanned Aircraft: Theory and Practice*; Princeton University Press: Princeton, NJ, USA, 2012.
31. Tekaslan, H.E.; Atkins, E.M. Gradient Guided Search for Aircraft Contingency Landing Planning. In Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA), Atlanta, GA, USA, 19–23 May 2025; pp. 526–532. [CrossRef]

32. Raymer, D. *Aircraft Design: A Conceptual Approach*, 5th ed.; American Institute of Aeronautics and Astronautics, Inc.: Reston, VA, USA, 2012.
33. Napolitano, M.R. *Aircraft Dynamics: From Modeling to Simulation*; John Wiley: Hoboken, NJ, USA, 2011.
34. Coombes, M.; Chen, W.H.; Render, P. Reachability Analysis of Landing Sites for Forced Landing of a UAS. *J. Intell. Robot. Syst.* **2013**, *73*, 635–653. [CrossRef]
35. Matthew Coombes, W.H.C.; Render, P. Landing Site Reachability in a Forced Landing of Unmanned Aircraft in Wind. *AIAA J. Aircr.* **2017**, *54*, 1415–1427. [CrossRef]
36. Arslantaş, Y.E.; Oehlschlägel, T.; Sagliano, M. Safe landing area determination for a Moon lander by reachability analysis. *Acta Astronaut.* **2016**, *128*, 607–615. [CrossRef]
37. Kirchner, M.R.; Ball, E.; Hoffler, J.; Gaublomme, D. Reachability as a Unifying Framework for Computing Helicopter Safe Operating Conditions and Autonomous Emergency Landing. *IFAC-PapersOnLine* **2020**, *53*, 9282–9287. [CrossRef]
38. Chen, M.; Tomlin, C.J. Hamilton–Jacobi Reachability: Some Recent Theoretical Advances and Applications in Unmanned Airspace Management. *Annu. Rev. Control. Robot. Auton. Syst.* **2018**, *1*, 333–358. [CrossRef]
39. Shanmugavel, M.; Tsourdos, A.; White, B.; Zbikowski, R. 3D Dubins Sets Based Coordinated Path Planning for Swarm of UAVs. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Keystone, CO, USA, 21–24 August 2006.
40. Shanmugavel, M.; Tsourdos, A.; White, B.; Żbikowski, R. Cooperative path planning of multiple UAVs using Dubins paths with clothoid arcs. *Control Eng. Pract.* **2010**, *18*, 1084–1092. [CrossRef]
41. Atkins, E.M.; Portillo, I.A.; Strube, M.J. Emergency Flight Planning Applied to Total Loss of Thrust. *AIAA J. Aircr.* **2006**, *43*, 1205–1216. [CrossRef]
42. Fallast, A.; Messnarz, B. Automated trajectory generation and airport selection for an emergency landing procedure of a CS23 aircraft. *CEAS Aeronaut. J.* **2017**, *8*, 481–492. [CrossRef]
43. Bacon, B.; Gregory, I. General Equations of Motion for a Damaged Asymmetric Aircraft. In Proceedings of the AIAA Atmospheric Flight Mechanics Conference and Exhibit, Hilton Head, SC, USA, 20–23 August 2007. [CrossRef]
44. Lampton, A.; Valasek, J. Prediction of Icing Effects on the Lateral/Directional Stability and Control of Light Airplanes. In Proceedings of the AIAA Atmospheric Flight Mechanics Conference and Exhibit, Keystone, CO, USA, 21–24 August 2006. [CrossRef]
45. National Imagery and Mapping Agency. *Department of Defense World Geodetic System 1984, Its Definition and Relationships with Local Geodetic Systems*; Technical Report TR8350.2; National Imagery and Mapping Agency: Bethesda, MD, USA, 2000.
46. U.S. Census Bureau. *RACE*; U.S. Census Bureau: Suitland, MD, USA, 2020.
47. Rana, I.K. *An Introduction to Measure and Integration*, 2nd ed.; American Mathematical Society: Providence, RI, USA, 2002.
48. Chae, S.B. *Lebesgue Integration*, 2nd ed.; Springer: New York, NY, USA 1995.
49. Beckmann, N.; Kriegel, H.P.; Schneider, R.; Seeger, B. The R*-tree: An efficient and robust access method for points and rectangles. *SIGMOD Rec.* **1990**, *19*, 322–331. [CrossRef]
50. Johnson, S.G. The NLopt Nonlinear-Optimization Package. 2007. Available online: https://github.com/stevengj/nlopt (accessed on 24 April 2025.).
51. Bandi, S.; Thalmann, D. Space discretization for efficient human navigation. *Comput. Graph. Forum* **1998**, *17*, 195–206. [CrossRef]
52. Henrich, D.; Wurll, C.; Worn, H. Online path planning with optimal C-space discretization. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Victoria, BC, Canada, 17 October 1998; Volume 3, pp. 1479–1484. [CrossRef]
53. Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Prentice Hall Series; Pearson: Upper Saddle River, NJ, USA, 2010.
54. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
55. Libspatialindex Contributors. Libspatialindex: A General Framework for Spatial Indexing. 2025. Available online: https://libspatialindex.org/en/latest/ (accessed on 15 February 2025).