Stage-Optimal Dubins Tour Planning for Fixed-Wing UAV Using Hybrid A* Search

1st Aufa Rienaldifaza Ahmad
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
aufa.rienaldifaza.a@gmail.com

***Abstract*—This paper presents a two-stage path planning pipeline for fixed-wing Unmanned Aerial Vehicles (UAVs) executing multi-target visitation missions in obstacle-dense environments. In the first stage, kinematically feasible and collision-free pairwise path costs between target configurations are computed using the Hybrid A* algorithm, which leverages the analytical Dubins path length as a heuristic. In the second stage, the Held-Karp dynamic programming algorithm is employed to solve the Travelling Salesman Problem, determining the optimal target visitation order that minimizes the overall flight distance. The results demonstrate that while the hybrid search successfully computes safe, kinematically valid paths that avoid all physical obstacles, control loop latency, and vehicle dynamic limits in the physical simulator introduce path-following deviations such as corner-cutting.**

***Index Terms*—A* Algorithm, Dubin's Tour, Swarm, UAV**

I. Introduction

Unmanned Aerial Vehicle (UAV) has gained prominence in its utilization in military missions. In its autonomous operations, UAVs face multiple challenges, for example determining the optimal path given a set of targets with dynamic obstacles and kinematic constraints. Path planning algorithms for autonomous UAV missions have been developed with generalized kinematic models, such as the Dubins vehicle model, which captures the turning radius constraint imposed by fixed-wing platforms operating at constant speed [1]. Although reliable in obstacle-free environments, this model must be extended to handle real-world missions where terrain, structures, or restricted zones must be avoided.

When a single fixed-wing UAV must visit a set of targets in a mission, the order in which those targets are visited determines the total path length. This is because the Dubins path cost between two configurations depends on both position and heading, so the arrival heading at one target directly affects the departure cost toward the next. The problem of finding the minimum-cost tour over all target orderings under this heading-dependent cost metric is known as the Dubins Vehicle Problem (DVP) [1], a variant of the Traveling Salesman Problem (TSP) that remains NP-hard.

A two-stage pipeline for kinematically feasible tour planning of a single fixed-wing UAV over a set of target locations is proposed here. In the first stage, pairwise path costs between all target configurations are computed using Hybrid A* search [4], which produces paths that are both kinematically feasible and obstacle-free. The Dubins-to-goal distance serves as the admissible heuristic, so the search is guaranteed to find the shortest feasible path within the discretized state space. In the second stage, the optimal visitation ordering over all targets is determined using the Held-Karp dynamic programming algorithm [5], which solves the TSP exactly in $O(2^N N^2)$ time, a significant improvement over the $O(N!)$ complexity of exhaustive enumeration.

II. Theoretical Foundation

A. Dubins Paths

A Dubins path is the shortest kinematically feasible curve connecting two vehicle configurations $q_1 = (x_1, y_1, \theta_1)$ and $q_2 = (x_2, y_2, \theta_2)$ under the turn-rate constraint $|\dot{\theta}| \leq \Omega$, assuming an obstacle-free environment. Dubins [1] proved that such a path consists of at most three segments, each being either a maximum-rate circular arc (C) or a straight line (L). This yields six candidate path types divided into two classes as shown in Fig. 1:

- **C-L-C class**: LSL, RSR, LSR, RSL, a circular arc followed by a straight segment followed by a circular arc. These paths arise when the separation between q_1 and q_2 is large relative to the minimum turning radius $r_{\min}$.
- **C-C-C class**: RLR, LRL, three consecutive circular arcs. These paths arise when the separation is small relative to $r_{\min}$, causing the construction circles of the C-L-C paths to intersect.

The Dubins path $d_D(q_1, q_2)$ is the shortest among all feasible candidates. While this formulation efficiently captures the kinematic constraints of fixed-wing platforms, it assumes the environment is free of obstacles. This assumption limits its direct applicability to real-world missions where terrain, structures, or threat zones must be avoided.

B. Traveling Salesman Problem

The Traveling Salesman Problem (TSP) asks for the minimum-cost tour that visits a set of N vertices exactly once. The Traveling Salesman Optimization Problem (TSOP) seeks the minimum-cost tour:

$$\pi^* = \arg\min_{\pi \in \Pi} \sum_{j=1}^{N} c(\pi(j),\, \pi(j+1)) \tag{1}$$

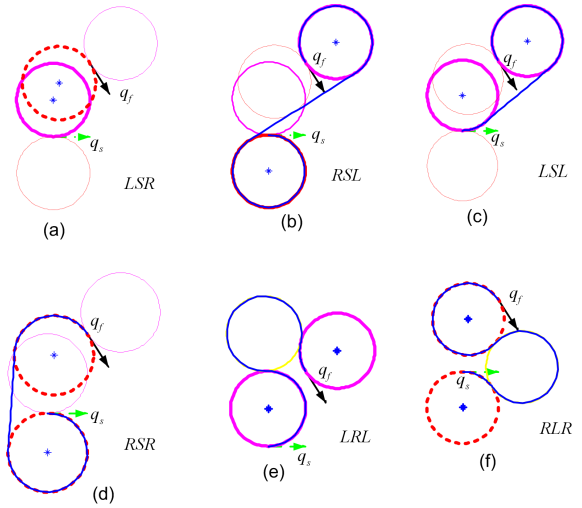

Fig. 1. Theoretical formulation of Dubins path classes (C-L-C and C-C-C) connecting two configurations.

where Π is the set of all permutations over the N vertices and $c(\cdot,\cdot)$ is the inter-vertex cost. The Traveling Salesman Decision Problem (TSDP) asks whether a tour of total cost at most B exists:

$$\exists\, \pi \in \Pi \;\text{ such that }\; \sum_{j=1}^{N} c(\pi(j),\, \pi(j+1)) \leq B \qquad (2)$$

The TSDP is NP-complete and the TSOP is NP-hard [2]. When the inter-vertex cost is defined by Dubins path lengths rather than Euclidean distances, the problem is referred to as the Dubins Vehicle Problem (DVP) [3]. The DVP remains NP-hard due to its heading-dependent cost structure: the cost of traveling from target τ_j to target τ_k depends on the arrival heading at τ_j, which is itself determined by the preceding leg. This heading coupling prevents the problem from being decomposed into independent subproblems.

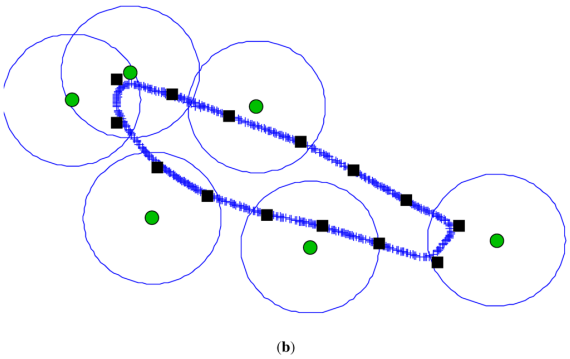

Fig. 2. Theoretical formulation of the Dubins Vehicle Problem (DVP) showing heading coupling and tour planning.

C. Hybrid A* Search

Standard A* operates over a discrete grid, producing paths that respect obstacle boundaries but ignore vehicle kinematic constraints. Hybrid A*, introduced by Dolgov et al. [4], extends this by associating each grid cell with a continuous state $q = (x, y, \theta)$, ensuring that the planned path is both obstacle-free and kinematically feasible.

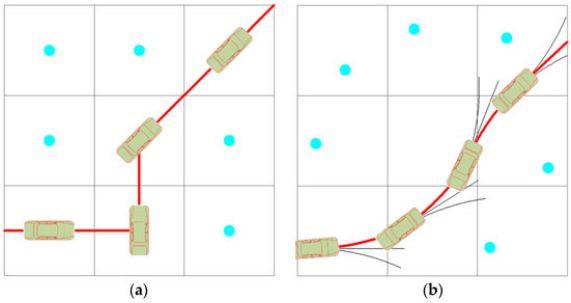

Fig. 3. Theoretical search grid expansion of the Hybrid A* algorithm incorporating non-holonomic constraints.

1) State Expansion: Rather than expanding to fixed grid neighbors, Hybrid A* generates successor states by integrating the kinematic model forward from the current state using a set of motion primitives. For a fixed-wing UAV, these primitives correspond to different curvature inputs:

$$q_{k+1} = q_k + \int_0^{\Delta t} \begin{pmatrix} V\cos\theta \\ V\sin\theta \\ u \end{pmatrix} dt, \quad u \in \{-\Omega, 0, +\Omega\} \qquad (3)$$

where u is the applied turn rate. Each resulting state is checked for collision against the obstacle map, and infeasible successors are discarded.

2) Heuristic: The search uses an admissible heuristic $h(q, q_{\text{goal}})$ computed as the maximum of two independent lower bounds:

$$h(q, q_{\text{goal}}) = \max(h_{nh}(q, q_{\text{goal}}),\; h_h(q, q_{\text{goal}})) \qquad (4)$$

where h_{nh} is the non-holonomic-without-obstacles heuristic, computed as the Dubins path length from q to q_{goal} ignoring obstacles, and h_h is the holonomic-with-obstacles heuristic, computed as the 2D A* shortest path length from the grid cell of q to q_{goal} ignoring heading. Because both bounds underestimate the true cost-to-go, their maximum remains admissible and the search is guaranteed to find the optimal path within the discretized state space.

3) Relationship to Dubins Paths: In an obstacle-free environment, the Hybrid A* path $d_H(q_i, q_j)$ converges toward the Dubins path $d_D(q_i, q_j)$ as the motion primitive resolution increases. In the presence of obstacles, Hybrid A* produces a kinematically feasible detour that Dubins cannot represent.

D. Held-Karp Algorithm

The Held-Karp algorithm is an exact dynamic programming solution for the TSP. It reduces the time complexity from $O(N!)$ for exhaustive enumeration to $O(2^N N^2)$, making it tractable for moderate values of N [5].

Let $S \subseteq T$ be a subset of targets and $\tau_j \in S$ be a target in that subset. The DP state is defined as:

$$\mathrm{dp}[S][j] \tag{5}$$

the minimum cost to depart from q, visit exactly the targets in S, and arrive at the end target.
The base case initializes the cost of reaching each target directly from the UAV start configuration:

$$\mathrm{dp}[\{\tau_j\}][j] = c(q_0,\, \tau_j), \quad \forall\, j \in \{1, \ldots, N\} \tag{6}$$

The recurrence extends the tour by one target at a time:

$$\mathrm{dp}[S][j] = \min_{l \in S \setminus \{j\}} \Big(\mathrm{dp}[S \setminus \{j\}][l] + c(\tau_l,\, \tau_j) \Big) \tag{7}$$

where $c(\tau_l, \tau_j)$ is the path cost from target τ_l to target τ_j. For Dubins-based costs, this transition cost depends on the arrival heading at τ_l inherited from the preceding leg. In this work, $c(\tau_l, \tau_j)$ is computed as the minimum Hybrid A* path cost over all feasible arrival headings at τ_j, which provides a valid lower bound on the true chained cost. The optimal tour cost is recovered as $\min_{j \in T} \mathrm{dp}[T][j]$, and the tour itself is reconstructed by backtracking through the DP table.

III. Problem Formulation

A. UAV Model

A single fixed-wing UAV is modeled as a Dubins vehicle with state $q = (x, y, \theta)$ describing its position and heading. The kinematic model is given by:

$$\dot{x} = V\cos(\theta), \quad \dot{y} = V\sin(\theta), \quad |\dot{\theta}| \le \Omega \tag{8}$$

where V is the constant forward velocity and $\Omega = V/r_{\min}$ is the maximum turn rate determined by the minimum turning radius $r_{\min}$. The UAV departs from a fixed start configuration $q_0 = (x_0, y_0, \theta_0) \in W_{\text{free}}$.

TABLE I
UAV Model and Planner Specifications

Parameter	Value	Unit
UAV Model / Airframe	Mini Talon V-Tail	-
Cruising Airspeed (V)	12.5	m/s
Roll Limit (ϕ)	30.0	deg (°)
Minimum Turning Radius ($r_{\min}$)	27.59	m
Grid Scaling Factor (SCALE)	50.0	m/unit
Takeoff Runway Safety Offset	200.0	m
Obstacle Cylinder Radius	10.0	m
Obstacle Cylinder Height	120.0	m
Planner Inflation Radius ($R_{\text{inflation}}$)	25.0	m
Home Latitude	-35.363262	deg (°)
Home Longitude	149.165237	deg (°)
Target Cruise Altitude	60.0	m

B. Target Set

A set of N targets is defined as $T = \{\tau_1, \tau_2, \ldots, \tau_N\}$, where each target τ_j is located at a fixed position $p_j \in W_{\text{free}}$. The arrival heading at each target is a free variable, minimized as part of the path cost computation.

C. Path Cost

The cost of traveling between two configurations is given by the Hybrid A* path cost $d_H(q_i, q_j)$, defined as the length of the shortest kinematically feasible path from q_i to q_j that remains entirely within W_{free}. In obstacle-free environments, d_H reduces to the Dubins path cost d_D. This substitution preserves kinematic feasibility while extending the planner to environments with obstacles.

D. Tour Cost

A tour is a permutation π over the N targets defining the visitation sequence. The total tour cost is:

$$C(\pi) = d_H\big(q_0,\, q_{\pi(1)}\big) + \sum_{j=1}^{N-1} d_H\big(q_{\pi(j)},\, q_{\pi(j+1)}\big) \tag{9}$$

where $q_{\pi(j)}$ denotes the configuration of the UAV upon arrival at the j-th target in the tour. The departure heading at each intermediate target is determined by the arrival heading from the preceding leg, so consecutive Dubins costs are heading-coupled and cannot be minimized independently.

E. Optimization Problem

The goal is to find the permutation π^* that minimizes the total tour cost:

$$\pi^* = \arg\min_{\pi \in \Pi} C(\pi) \tag{10}$$

where Π is the set of all $N!$ permutations over T. This constitutes the Dubins Vehicle Problem (DVP), which is NP-hard. Using the Held Karp algorithm, the time complexity is reduced to $O(2^N N^2)$ by exploiting the optimal substructure of partial tours. Path costs between target pairs are precomputed using Hybrid A* and stored in a cost matrix prior to the DP, so the tour ordering stage operates entirely on the precomputed matrix without further path queries.

IV. Methodology

A. System Overview

The proposed pipeline solves the Dubins Vehicle Problem for a single fixed-wing UAV visiting N targets through two sequential stages. In the first stage, pairwise path costs between all target configurations are precomputed and stored in a cost matrix. In the second stage, the Held-Karp dynamic programming algorithm operates on this matrix to find the optimal visitation ordering. Both stages are implemented in Python using NumPy for numerical computation and Plotly for result visualization. The pipeline is evaluated under three test conditions of increasing complexity, described in Section IV-D.

B. Stage 1: Cost Matrix Construction

Given the UAV start configuration q_0, the target set T, and an optional obstacle map O, this stage produces two outputs: the entry cost vector $\mathbf{c}_0 \in \mathbb{R}^N$, where $\mathbf{c}_0[j]$ is the minimum path cost from q_0 to target τ_j, and the inter-target cost matrix $\mathbf{C} \in \mathbb{R}^{N \times N}$, where $\mathbf{C}[j][k]$ is the minimum path cost from target τ_j to target τ_k.

Arrival and departure headings at each target are discretized into K equally spaced values over $[0, 2\pi)$. For each pair of configurations, the minimum cost over all heading combinations is computed using Hybrid A* when O is non-empty, and using the shortest Dubins path otherwise. Minimizing over all heading combinations produces a valid lower bound on the true heading-coupled cost, as fixing headings can only increase the Dubins path length.

C. Stage 2: Tour Ordering via Held-Karp

Given $\mathbf{c}_0$ and $\mathbf{C}$, the Held-Karp algorithm finds the permutation π^* minimizing $C(\pi)$ as defined in Equation (9). Subsets of T are represented as integer bitmasks over $\{0, \ldots, N-1\}$, where bit j is set if and only if τ_j belongs to the subset. This permits subset enumeration and set operations to be performed using bitwise arithmetic in constant time. The DP table has $2^N \cdot N$ entries, and the algorithm runs in $O(2^N N^2)$ time with $O(2^N N)$ space. The tour is recovered by backtracking through the parent pointer table.

D. Experimental Validation

1) Test Case 1: Nonholonomic Benchmark: The first test validates the correctness of both the Dubins path computation and the Held-Karp tour ordering against established benchmarks from the literature. Target configurations are drawn from the circular arrangement used in Kenefic [3], in which N targets are placed uniformly on a circle of radius R and the UAV departs from the center with a fixed initial heading. This arrangement admits a known tour structure that serves as a reference for the computed solution.

2) Test Case 2: Obstacle Avoidance: The second test evaluates the obstacle-aware extension using Hybrid A*. Synthetic 2D environments are constructed with rectangular and circular obstacles placed such that direct Dubins paths between several target pairs are blocked. The cost matrix is recomputed using Hybrid A* with the composite admissible heuristic from Equation (4), and the Held-Karp algorithm is then applied to the resulting matrix.

Each path in the computed tour is verified for two properties. Collision freedom is checked by confirming that no sampled point along the path lies within any obstacle boundary. Kinematic feasibility is verified by confirming that the curvature at every sampled point does not exceed $1/r_{\min}$. The total tour cost is compared against a Dubins-only baseline computed on the same target configuration without obstacles, to quantify the overhead introduced by obstacle avoidance.

3) Test Case 3: Flight Simulation: The third test validates the complete pipeline in a physics-based flight simulation environment. The software stack is composed of the following components:

- **Gazebo**: physics and world simulation environment providing obstacle geometry, aerodynamic ground effects, and sensor output.
- **ArduPilot SITL (ArduPlane)**: software-in-the-loop flight controller simulating the fixed-wing autopilot stack, including attitude control, navigation, and mission execution.
- **URDF model**: robot description of the fixed-wing UAV loaded into Gazebo via the ArduPilot Gazebo plugin, which relays simulated sensor data to the SITL instance and applies actuator commands back to the physics engine.
- **MAVLink**: communication protocol used to upload the computed waypoint mission to the ArduPlane SITL instance and to monitor flight state during execution.
- **ROS2**: middleware layer hosting the planning nodes and the MAVROS2 bridge, which translates between MAVLink messages and ROS2 topics.

The waypoints produced by Stage 2 are expressed in local Cartesian coordinates relative to the simulation origin. Before upload, these are converted to GPS coordinates using a flat-Earth projection centered at the designated home position. The converted mission is then sent to ArduPlane via the MAVROS2 `/mavros/mission/push` service, after which the UAV is armed and placed in AUTO mode to execute the mission.

V. Results and Discussion

The performance of the path planning pipeline was evaluated under both obstacle-free and obstacle-constrained environments. A particular focus was placed on comparing the continuous Dubins path planner with the discrete Hybrid A* search.

A. Nonholonomic Benchmark Test

In an obstacle-free benchmark consisting of $N = 10$ targets arranged on a circle of radius $R = 2.0$, both algorithms were executed to compute the minimum-cost tour. While the Dubins path solver outputs the exact mathematical minimum, the Hybrid A* algorithm operates over a discretized state space with $K = 8$ heading angles and a step size of 0.2.

As shown in Fig. 4, Hybrid A* successfully converges to the overall path structure of the Dubins tour. However, the Hybrid A* path creates a slightly jagged behavior due to the discrete motion primitives. This discretization error results in a total tour cost difference of $+6.35\%$. A summary of these results and the physical justification for this behavior is detailed in Table II.

B. Resolution Optimality in Constrained Environments

While the discretization of Hybrid A* introduces a minor length penalty compared to the continuous Dubins baseline, it provides a critical advantage in environments containing

TABLE II
COMPARISON OF PATH PLANNING RESULTS AND DISCRETIZATION JUSTIFICATIONS

Planner	Total Cost	Rel. Error	Path Characteristics & Justification
Dubins (Continuous)	12.4121	Baseline	Smooth, continuous, mathematically optimal in obstacle-free space.
Hybrid A* (K=8, step=0.2)	13.2000	+6.35%	Resolution-optimal, slightly jagged due to discrete step length and heading constraints.

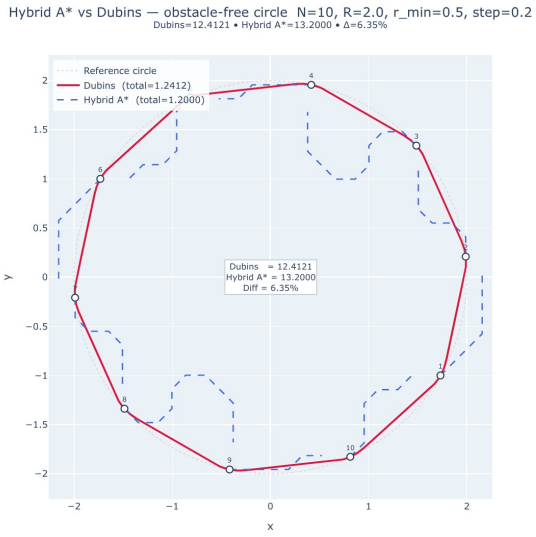

Fig. 4. Overlay of Hybrid A* and Dubins path solutions demonstrating convergence and discretization effects.

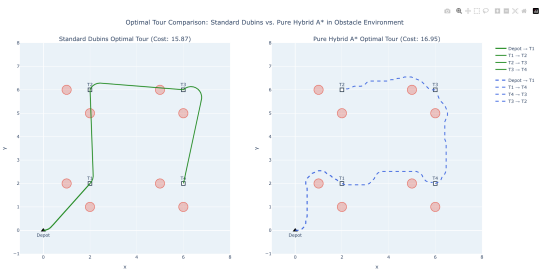

Fig. 5. Visual comparison of standard Dubins paths (colliding) and pure Hybrid A* obstacle-avoiding paths in a constrained environment with 4 targets and 8 obstacles.

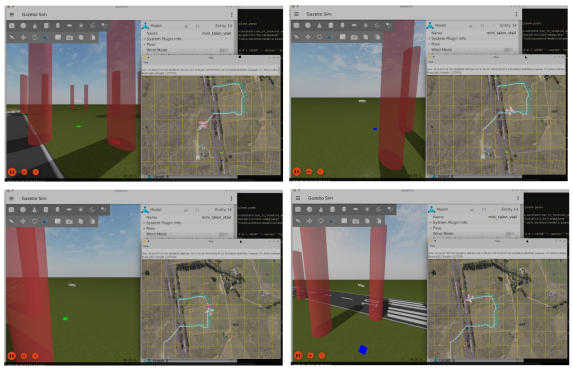

Fig. 6. Planned Hybrid A* path successfully avoiding all 8 obstacles while finding the optimal tour to visit each of the 4 targets.

obstacles. Standard Dubins paths cannot represent or avoid obstacle boundaries dynamically. In contrast, Hybrid A* is resolution-optimal, guaranteeing that it finds the shortest kinematically feasible and collision-free path within the limits of the discretized grid and heading resolution.

To evaluate this performance, a test case with 4 target nodes and 8 circular obstacles (2 obstacles surrounding each target) was set up. As illustrated in Fig. 5, standard Dubins path calculations (left plot) completely fail to account for the obstacles, leading to direct collisions. Meanwhile, the pure Hybrid A* search (right plot) successfully navigates around the obstacles.

Crucially, even in this pure search mode, the Hybrid A* algorithm uses the analytical Dubins path distance as its non-holonomic-without-obstacles heuristic. This allows the search space expansion to remain informed of the UAV's minimum turning radius constraint at every step, significantly reducing the search time while guaranteeing collision avoidance.

The full planned tour computed by our pipeline in this environment is shown in Fig. 6. The Hybrid A* path planning algorithm successfully constructs an optimal, collision-free tour that routes the aircraft through all target nodes while navigating around the obstacle clearances. In practice, the resulting trajectory can serve as an initialization (warm start) for continuous path smoothing.

C. Discrepancies in Real-Life Physics Simulation

To evaluate the practical applicability of the planned waypoints, the generated mission was loaded into the ArduPilot SITL simulator using a Mini Talon V-Tail model in a Gazebo virtual environment. As shown in Fig. 7, the actual flight path taken by the UAV deviates from the reference path planned by the Hybrid A* algorithm.

Several physical and control factors explain why the planned path does not translate identically to the real-life simulation trajectory:

1) **Kinematic vs. Dynamic Modeling:** The path planner assumes a simplified kinematic model (Dubins vehicle) operating at a constant speed with instantaneous attitude transitions. In contrast, the simulator incorporates full flight dynamics (mass, inertia, lift, drag, sideslip) and actuator physical limits, preventing instantaneous roll and turn rate adjustments.
2) **Autopilot Guidance and Look-Ahead:** The ArduPlane autopilot uses navigation guidance controllers (such as

Fig. 7. Comparison between the planned Hybrid A* path and the actual flight trajectory executed by the UAV in the Gazebo physics simulation.

the L1 controller) that track the path by aiming at a dynamic look-ahead point. To maintain speed and flight stability while transitioning between waypoints, the autopilot cuts corners, resulting in smoother but wider turns than the discrete geometric transitions in the plan.

3) **Takeoff and Altitude Dynamics:** The simulator starts with the UAV on the runway at zero velocity. The UAV must first execute a straight-line takeoff and climb to the target altitude of 60.0 meters. This vertical transition overlaps with the horizontal path-following controller, causing initial tracking deviations.
4) **Aerodynamic Disturbances:** Factors like wind, ground effect during takeoff/landing, and sideslip introduce drift that the feed-forward path generator cannot predict, requiring the feedback control loop to constantly adjust, creating minor oscillations around the planned trajectory.

VI. Conclusion and Future Work

In this work, a stage-optimal Dubins tour planning pipeline utilizing Hybrid A* search and the Held-Karp dynamic programming algorithm for a fixed-wing UAV in obstacle-rich environments is developed. The algorithm successfully solved the target sequencing problem and generated collision-free path segments respecting the kinematic constraints of the Mini Talon V-Tail model. However, flight evaluations using the ArduPilot SITL controller and Gazebo physics simulator highlighted a discrepancy between the planned kinematic trajectory and the actual flight path. These deviations arise from full aircraft flight dynamics, actuator limitations, and autopilot corner-cutting during waypoint navigation. While applying an obstacle inflation buffer in the path planner effectively mitigated collision risks, achieving precise tracking requires incorporating a more sophisticated aerodynamic and dynamic model directly into the state expansion and heuristic computation of the path planning phase.

Acknowledgment

The author would like to express sincere gratitude to Dr. Ir. Rinaldi, M.T., lecturer of IF2211 Algorithmic Strategy course for his guidance throughout this semester. The author would also like to thank Aksantara ITB and his colleagues for introducing him to robotics motion planning that ultimately inspired this paper.

Appendix

The implementation source code for this project is available on GitHub at: https://github.com/aufafaza/13524027_makalah_stima. A video demonstration of the flight simulation and pipeline results is available on YouTube at: https://youtu.be/j_H6oRjkXHE.

References

[1] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, pp. 497–516, 1957.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[3] R. J. Kenefic, "Finding good tours for the Dubins traveling salesman problem," in *Proc. IEEE Aerospace Conf.*, Big Sky, MT, USA, 2008, pp. 1–6.

[4] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," in *Proc. 1st AAAI Workshop Intell. Transp. Syst.*, Chicago, IL, USA, 2008, pp. 1–6.

[5] M. Held and R. M. Karp, "A dynamic programming approach to sequencing problems," *J. Soc. Ind. Appl. Math.*, vol. 10, no. 1, pp. 196–210, 1962.

[6] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Sci. Robot.*, vol. 7, no. 66, p. eabm6074, May 2022.

[7] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sendai, Japan, 2004, pp. 2149–2154.

[8] ArduPilot Development Team, "ArduPilot open source autopilot," 2024. [Online]. Available: https://ardupilot.org

[9] L. Meier *et al.*, "MAVLink: Micro air vehicle message marshalling library," 2009. [Online]. Available: https://mavlink.io

[10] V. Ermakov *et al.*, "MAVROS: MAVLink extendable communication node for ROS," 2014. [Online]. Available: https://github.com/mavlink/mavros

Statement

Hereby, I declare that this paper I have written is my own work, not a reproduction or translation of someone else's paper, and not plagiarized.

Bandung, 19 Juni 2026

Aufa Rienaldifaza Ahmad [13524027]