Deutscher Luft- und Raumfahrtkongress 2011
DocumentID: 241468

LOCAL PLANNING FOR A FIXED-WING UAV WITH A TREE-BASED PLANNER AND MOTION PRIMITIVES

M. Niendorf, Institute of Flight Mechanics and Control, University of Stuttgart, Germany
M. Gros, Institute of Flight Mechanics and Control, University of Stuttgart, Germany
A. Schöttl, MBDA Deutschland, Germany
W. Fichter, Institute of Flight Mechanics and Control, University of Stuttgart, Germany

Abstract

Motion planning for a fixed-wing UAV is a complex task, especially due to the dynamic constraints of the aircraft. The work presented here is based on a two-stage planning framework. The global planning stage consists of a probabilistic roadmap planner which computes a sequence of waypoints accounting for kinematic constraints. Those are fed into a deterministic tree-based local planner that locally refines the path using closed-loop dynamics motion primitives. In this paper we focus on the local planner which we present in detail. Tree expansion is guided by a metric based on Dubins curves. We provide simulation results comparing the metric to three alternative metrics to support our choice. The computational complexity is reduced by limiting the set of motion primitives depending on the environment. Furthermore, we redefine the waypoint proximity criterion by considering the Dubins metric. We discuss the suggested planner and thoroughly analyze its possible failure modes. The simulations are concluded by showing the efficiency of the local obstacle avoidance strategy using just the local planner and together with the global planner in a complex urban scenario. The simulation reveals promising behavior for both the local planner and the coupled planners.

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have been in development for several decades and will play an increasingly important role in various future scenarios. Typical missions for UAVs today include surveillance tasks, reconnaissance, environmental monitoring, etc. As of today, most deployed unmanned aerial systems make use of a human operator in the loop who remotely pilots the aircraft. However, it is desirable to strive for a higher degree of automation to enable the operator to give higher level commands.

Within complex obstacle-filled environments operating an UAV holds several challenges to be solved. Fixed-wing aircrafts must maintain a velocity sufficiently greater than stall speed at any time in order to provide safe operation. Furthermore, since hovering in place is not possible, heading and flight path angle changes require significant space and time. Obstacle-cluttered environments therefore require a planner that exploits the maneuverability of the underlying system.

A popular approach to solving the motion planning problem in an obstacle-cluttered environment is to decompose it into a coarse, discretized global planning problem and a finer local planning problem which accounts for the dynamic constraints.

Roadmap-based planners have been widely adopted to solve the global planning problem [1,2]. Probabilistic Roadmaps (PRM) have gained special attention due to their favorable runtime and completeness guarantees as well as their multi-query property. During the query phase, well studied graph search algorithms such as A* are employed to obtain a solution. Randomized Trees such as the RRTs have been used for path planning under kinodynamic constraints [3]. However, increasing the dimension of the search space to the state space to account for dynamic constraints increases the computational effort dramatically. Therefore, it is common to perform sampling in the configuration space with position and attitude or even work space with only position information rather than in the state space which would be preferable in terms of system dynamics.

The reduction of the state space of a system to maneuver-space by the introduction of a state machine, which interconnects steady-state trajectories and maneuver trajectories, was introduced in [4]. This allows for implicitly addressing the dynamic constraints of the system. The application of this concept to a tree-based planner, which builds a tree of motion primitives, was presented in Ref. [5].

In [6] we introduced a two-staged approach to the UAV motion planning problem for a small fixed-wing UAV, which consists of a modified PRM stage to solve the global planning problem and a deterministic online tree-based local planning stage which implicitly addresses the dynamic constraints by using closed-loop motion primitives. It refines the trajectory locally. The local planner is strictly deterministic in contrast to the RRT motivated planner strategy mentioned above. It uses a metric based on Dubins curves as heuristic. Obstacle avoidance during runtime is crucial when unmanned aerial vehicles fly through dynamic environments. Thus, it makes sense to directly embed range information such as from a laser range finder (LRF) or a radar distance sensor into the motion planner. Such an approach is reported in [7] and [8] in the framework of RRTs and potential fields, respectively.

The objective of this paper is to provide deep insight into the deterministic online local planner and to introduce details beyond the scope of the previous paper [6]. We depict the algorithm in detail. We pay special attention to the choice of an appropriate metric and provide simulation results using 4 different metrics. The switching maneuver automaton, which reduces the computational complexity by limiting the set of applicable motion primitives depending on the environment, is introduced. Furthermore, we introduce a new waypoint proximity criterion and provide simulation results concerning the efficiency of the local obstacle avoidance strategy. We discuss the local planner and thoroughly analyze its possible failure modes. The remainder of this paper is organized as follows. Section 2 provides a brief overview of the PRM global planner. The local planner is presented in detail in section 3 and results obtained from simulation are presented in section 4. A discussion of the results paying special attention to the performance of the local planner in section 5 leads to the conclusions given in section 6.

2. GLOBAL PLANNER

The global planner computes a series of waypoints which lead from start to goal offline, i.e. once before the actual flight and accounts for static and known obstacles. The Probabilistic Roadmap algorithm is used as global planner due to its well-studied advantageous properties, i.e. probabilistic completeness and multi-query capability.

Planning a path for a robot using a probabilistic roadmap consists of two phases. First, a predefined number of random nodes are sampled from the free work space, R^3 in this case. The planner then tries to connect *n* nearest neighbors to each other with straight line segments. We store the nodes in a kd-tree to allow for efficient n-nearest neighbor search. The edge candidates are checked against collision with the world model by using oriented bounding boxes (OBB) and the collision algorithm depicted in [9].

We impose additional constraints on the slope as well as on the length of the edges to pay respect to the kinematic constraints of the UAV [6].

The resulting graph represents the free space and can be used to solve multiple queries in a second step. Start and goal are connected to the roadmap and then a modified version of the A* algorithm is used to solve the query. Modifications account for flight path constraints, e.g. the heading change between two consecutive edges is limited to be smaller than a threshold.

The a priori known obstacles are modeled using axis aligned bounding boxes. The notion of the Dubins aircraft is exploited to develop a metric. Even though, this metric does not underestimate the distance between two points as required by the A* algorithm to be admissible, we achieved good results in practice, especially when combining the global and the local planner. The 3D Dubins heuristic will be discussed in detail in the following section.

The found path is smoothed in a post-processing step, which skips nodes by connecting non-neighbor nodes with each other, if the resulting edges are collision-free and satisfy the previously mentioned constraints.

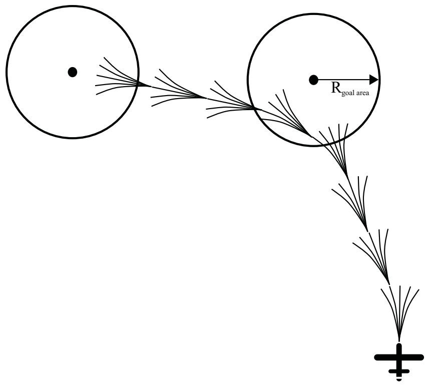


FIGURE 1. Exemplary Tree Expansion in 2D

3. LOCAL PLANNER

So far, the global planner has been presented, which generates a series of waypoints accounting for kinematical constraints. However, as the a priori known world model might be incomplete, i.e. unknown obstacles might exist and to make use of the full capabilities of the aircraft, an online local planner is executed during the flight. We chose an approach which builds a tree of motion primitives to directly address the nonholonomic constraints of the aircraft. Figure 1 illustrates the exemplary use of a tree of discrete maneuvers, which can be connected smoothly to each other, to find a path from an initial position to a goal position passing by a waypoint, with a goal area represented by a circle in R^2 with radius R_{goal}. The local planner is meant to refine the global path locally with a set of densely sampled motion primitives. We will now briefly cover the simulation model as well as the motion primitive generation and then focus on selected details.

3.1. Simulation Model

To generate the set of motion primitives, a model of the plant is necessary. In this work we use a nonlinear 6 degrees of freedom model of a fixed-wing aircraft. We simulate the closed-loop model due to the stability advantages. The control loops were designed in order to give flight path variables as reference inputs. This leads to the following closed-loop system

(1) $\dot{x} = f(x(t), r(t)), x(0) = x_0, r = (\phi_{com}, \gamma_{com}, V_{com}),$

where ϕ_{com} is the commanded roll angle. For small γ, a small angle of attack α and sideslip angle β, ϕ is equivalent to the bank angle μ. Therefore, ϕ is chosen as reference signal of the lateral motion. The commanded flight path angle γ_{com} and the commanded absolute speed V_{com} are chosen to be the reference signals of the longitudinal motion. The reference input vector is therefore defined as $r = (\phi_{com}, \gamma_{com}, V_{com})$. Refer to [6] for the full feedback policy and further details.

3.2. Motion Primitive Generation

To obtain the set of motion primitives (MPs), we suggested a straight-forward sampling method. The response of the nonlinear system in the interval $t \in [t_s, t_g]$ with initial state x_s to the input vector r_s is recorded as a motion primitive. By sequentially applying this method to all combinations of x_s and r_s we obtain a set of motion

Algorithm 1 Dubins Metric I

```
1: $L_{h,Dubins}\leftarrow$ compute horizontal Dubins distance $(q_i, q_g)$
2: $h \leftarrow L_{h,Dubins}$
3: while $\Delta z/h \geq \sin(\gamma_{max})$ do $h \leftarrow h + 2\cdot\pi\cdot R_{min}$ end
4: $L_{v,Dubins} \leftarrow h + \Delta z$
5: $L_{3D,Dubins} \leftarrow L_{h,Dubins} + L_{v,Dubins}$
```

ALGORITHM 1. Dubins Metric I

primitives, which can be stored in a maneuver automaton. We apply each input vector for a time interval that satisfies $t_g \geq t_{set}$, where t_{set} is the settling time of the underlying controller loops. Thus, we call these motion primitives steady-state motion primitives. Furthermore, a second class called non-steady state motion primitives was introduced to enable small heading changes in the horizontal plane, which could not be realized with the steady-state motion primitives due to the magnitude of t_{set}. For a more rigorous explanation of the motion primitive generation process as well as a detailed description of the non-steady state primitives, refer to [6].

3.3. Metric

In motion planning the appropriate choice of a metric measuring proximity between configurations, i.e. position and attitude, is crucial. When dealing with a system that is subject to nonholonomic constraints such as a car or a fixed-wing aircraft, both constrained with a minimum turning radius $R_{\min}$, the Euclidean distance between two points can be a poor estimate of the achievable path length. To find the shortest path between two configurations in a 2d environment for a nonholonomic car, Dubins presented a sufficient set of 6 path types, each consisting of at maximum 3 segments, which can either be circular arcs or straight line segments [10].

A significant speed gain in computation can be achieved, if the final orientation is left arbitrary such that the shortest path consists of at most 2 segments. An analytical solution to this problem was given in [11]. We use this result in the horizontal plane to compute the length of the shortest path from an initial configuration $q_i = (x_i, y_i, \chi_i)$, where χ_i is the initial heading angle, to a goal position $q_g = (x_g, y_g)$.

Since the constraints on possible motion for the Dubins car and an aircraft are closely related, the notion of a Dubins car can be expanded to R³. The Dubins aircraft was introduced in [12]. In [5] a metric based on Dubins curves in 3d was depicted. We suggest 3 different metrics, which each use a sum of a horizontal metric and a vertical metric to compute a 3d metric. The metric in the horizontal plane is computed as the length of a two segment Dubins path between the start configuration and goal position. The 3 suggested metrics are depicted in table 1. Dubins Metric I uses the approach depicted in algorithm 1 to compute the path length, i.e. if the flight path angle γ is too large, revolutions of a circle with minimum turning radius are added until the path angle constraint is satisfied. However, other than in [5] we add Δz as we obtained better results with this modification. Dubins Metric II is the sum of the length of the Dubins

	Dubins Metric I	Dubins Metric II	Metric 3
L_I	Dubins curve	Dubins curve	Dubins curve
L_v	Helix	Dubins curve	Eq. 1

TAB 1. Metric comparison

Algorithm 2 Tree of Motion Primitives

```
1: $node_{best} \leftarrow open\_list.push() \leftarrow node_{UAV} \leftarrow q_i$; $age = 0$;
2: while $x_g$ not reached
3:    if waypoint reached
4:       update waypoint to next from PRM
5:       $open\_list$ : for all nodes: update $J$, evaluate $G$
6:       $node_{best} \leftarrow open\_list.pop()$
7:       $age\_waypoint \leftarrow age$
8:    else
9:       $age \leftarrow age + 1$
10:      $node_{best}$ for all successors: evaluate $J$ and $G$, collision/safety maneuver check
11:      for all feasible successors: $open\_list.push()$
12:      if $open\_list$ is empty then planning failure end
13:      if $age - age\_waypoint > threshold$
14:         if $waypoint = global\_goal$ then planning failure end
15:         else skip $waypoint$
16:      $node_{best} \leftarrow open\_list.pop()$
17:   if $age - age_{old} > planning\_depth$
18:      $path \leftarrow$recursively find path from find path from $current\_node$ to $node_{UAV}$
19:      $node_{UAV} \leftarrow$move UAV to the end of next trim state along $path$
20:      if $node_{UAV} = node_{best}$ then planning failure end
21:      $open\_list$: delete branches whose origin nodes possess timestamps $\leq age(node_{UAV})$
22:      $age_{old} \leftarrow age$
```

ALGORITHM 2. Local Planner

curve in the horizontal as well as in the horizontal plane. In this case the same minimum turn radius is assumed in the horizontal and the vertical plane. Metric III is the sum of the Dubins curve in the horizontal plane and the vertical distance L_{vIII} computed as:

(2) $$L_{v,III} = \left|\sqrt{\Delta x^2 + \Delta y^2}\cdot\tan(\gamma) - \Delta z\right| / \sqrt{\Delta x^2 + \Delta y^2}$$

This will penalize any flight path angle, which is not pointing towards the goal. L_{vIII} is therefore not a measure of proximity but rather a measure of matching flight path angle.

3.4. Search Tree

The local planner builds a tree of motion primitives in order to find a feasible trajectory from a start configuration to a goal position. The planner returns a sequence of control inputs to the aircraft, which then follows the generated trajectory in an open-loop fashion.

The search tree is guided by the previously found waypoints. It refines the global solution into a dynamically feasible trajectory. Depth-first behavior is desirable in uncluttered environment to avoid unnecessary branching of the tree. However, in the proximity of obstacles as well as close to the goal breadth-first behavior is preferred. A node of the local planner can be fully represented by its configuration $q = (x, y, z, \phi, \gamma)$. Tree expansion is guided by a partially greedy cost function J. For a MP that starts at q_i and ends at q_f the cost functional J is defined as

(3) $$J(q_f) = f(q_i, q_f) + g(q_f, q_{goal}) + h(q_f, q_{obs})$$

where $f(q_i, q_f)$ is the trace length of the MP and $g(q_f, q_{goal})$ is the cost-to-go computed by using the 3D Dubins heuristic between q_f and q_{goal}. The term $h(q_f, q_{obs})$ represents the value of a potential function, which will be explained in the next section. It is important to note that the motion primitives do not have the same length, as the feedback controller cannot drive the nonlinear plant from one state to any arbitrary state in constant time. Therefore, the term $f(q_i, q_f)$ is necessary to account for the differing length. A purely greedy cost-function would favor longer motion primitives over shorter motion primitives.

The pseudo code given algorithm 2 depicts the structure of the local planner. The resemblance to the A* algorithm [13] is obvious. The algorithm maintains a priority queue of nodes called Open List, which is sorted according to the cost function. However, we limit the size of the Open List, which will drive the search tree towards the goal and avoid unnecessary branching in uncluttered environment, yet still provide enough options for tree expansion in more difficult terrain. The best node $node_{best}$ is expanded by applying all possible motion primitives in line 10. All maneuvers that can be smoothly connected to the configuration at the best node are stored as new branch candidates. The new branch candidates are then checked for collision and, if feasible, added to the search tree and inserted into the Open List with the appropriate priority in line 11.

To ensure the safety of the aircraft a branch is only added to the search tree, if at the end of each motion primitive at least one out of 4 safety maneuvers can be performed, which reverse the current heading. Safety maneuvers were represented by OBB placeholders and collision checked for every motion primitive. It is assumed that a safety maneuver takes advantage of the full capabilities of the aircraft and therefore the minimum turning radius is much smaller than the one observed during nominal flight represented by the motion primitives.

The online search property is realized by the introduction of the parameter “planning depth” in line 17. We limit the number of node expansions before propagating the aircraft to the next position, which is found by recursively calculating the trajectory from the current best node in the search tree to the current position of the aircraft in line 18. Parts of the tree, which become unreachable, are pruned in line 21.

We included a check for planning failure in three lines of the pseudo code in algorithm 2. In case one of the constraints is violated, a safety maneuver is executed. Obstacle avoidance techniques complementary to the collision checks countervail these cases.

3.5. Obstacle Penalty function

Typically, a free space heuristic such as the Euclidian distance or the Dubins metric is used to guide tree expansion, i.e. a possible line-of-sight to the goal is assumed. However, this might lead to the tree being dead-locked, as local minima such as the bug-trap might prevent further tree expansion. By the introduction of the term $h(q_f, q_{obs})$ into the cost function, we include environment information, resulting in a more informed heuristic. $h(q_f, q_{obs})$ is the value of a potential function that takes the position coordinates q_f and an obstacle position q_{obs} into account.

(4) $h(q_f, q_{obs}) = K \cdot 1/(|q_f - q_{obs}| - 1)$

q_{obs} is determined by the simulation of a laser range finder (LRF) similar to Ref. [7]. The simulation of the LRF assumes a beam with limited range d_{LRF}, variable pitch angle $\Delta\theta_{LRF}$ and variable azimuth angle $\Delta\psi_{LRF}$. q_{obs} is the minimum distance between q_f and any obstacle in range. Thus, motion primitives that lead too close to obstacles will be penalized heavily, which in return guides the tree towards the free space.

3.6. Switching Maneuver Automaton

The computational effort in motion planning can be reduced efficiently by the representation of the vehicle dynamics with a maneuver automaton. However, there is a trade-off between how densely the flight envelope is sampled and computational effort. It is desirable to densely sample the flight envelope, which will in return increase the computational effort significantly, because denser sampling requires a larger number of motion primitives, thus leading to a higher branching factor of the tree of motion primitives. Especially in obstacle cluttered environment or close to the initial and goal position it is favorable to have a wide set of motion primitives available to make use of the full dynamics of the aircraft. To address this dilemma a maneuver automaton can be defined, which only allows a subset of the complete set of motion primitives at a time [14]. One can classify subsets by different characteristics such as velocity, range, turn radius or by maximum roll angle. A switching function then has to be chosen, which limits the allowable states and transitions of the maneuver automaton to a specific set. The switching function can depend on time or environmental factors such as proximity to obstacles or proximity to start and goal.

To achieve a maximum of maneuverability in obstacle cluttered environment, we define a subset of motion primitives with roll angles greater than a threshold ϕ_{bound} that are applicable additionally when close to obstacles. In an obstacle free environment only MPs below ϕ_{bound} are applied; this is called limited set of MPs. We use obstacle proximity information provided by the simulated Laser Range Finder in the switching function G to provide the switching function depending on a threshold d_{bound}. Additionally, if the difference between current heading and the azimuth of an assumed line-of-sight connection to the goal is greater than an upper bound $\Delta\psi$ the MPs with $\phi \geq \phi_{bound}$ are also applicable to allow for turning with minimum turning radius.

3.7. Goal Area

During planning each waypoint can be considered a local goal for the local planner. The global goal is the last of the waypoints and is therefore planned towards by the local planner as if it was another local goal.

Due to the discrete nature of planning with motion primitives it is unlikely that goals can be reached precisely. It is therefore common practice to define a spherical goal area around the goal. The Dubins metric, which penalizes heading errors close to the goal heavily, makes the heuristic value alone an insufficient measure of proximity to a goal. Therefore the node with top priority might not be the physically closest to the goal. In general a global goal should be encountered as precisely as possible, i.e. the goal area around it should be small. In contrast, the waypoints only give “guidance” on how the local planner should approach the environment. It is therefore possible to relax the constraints on how precisely they need to be followed. Nonetheless, if the constraints are too loose, chances are high that the locally planned trajectory will deviate too far from the previously found global solution, which can then lead to planning failure, because the local planner might not succeed in guiding the aircraft back to the global solution.

ϕ	-45°	-20°	-10°	0°	10°	20°	45°
γ		-15°	-10°	0°	10°	15°	
v_k	20m						

TAB 2. Discretized Reference Inputs

Case	Dubins Metric I	Dubins Metric II	Metric 3	2-Norm
1	10	10	11	-
2	9	10	9	9
3	6	6	7	6

TAB 3. Number of node expansions for each metric

On implementation level two different ways to distinguish, whether a goal has been reached, were found to be efficient. An upper bound on the Euclidian distance R_{goal} and a looser upper bound on the Dubins metric serve as decision criteria. The looser bound on the Dubins metric R_{goal_Dubins} is possible, because heading information is directly incorporated into this metric. If one of the conditions is satisfied, the waypoint is reached and the next waypoint becomes active. For the global goal, i.e. the last waypoint, we chose that only the Euclidian distance is of interest, as this goal should be reached as precisely as possible with arbitrary heading. It is important to note, however, that the magnitude of the thresholds have to be adjusted to the specific model of the aircraft, i.e. the maneuver automaton. If the maneuver automaton included shorter or more aggressive maneuvers, one could lower the threshold and vice versa.

4. SIMULATION RESULTS

The following parameters have been set if not otherwise stated to conduct the presented simulations. Motion primitives for a set of $n=35$ reference vectors r shown in table 2 were generated. Additionally, to allow for small heading changes a set of non steady-state MPs has been added. In total we obtain a set of 1087 motion primitives. The limited set of motion primitives was chosen to be the Motion Primitives with a roll angle $\phi \geq 25^\circ$. The thresholds in the switching function G were set to $d_{bound} = 50m$ and $\Delta\psi = 15^\circ$. Parameters for the global planner are set to $\gamma_{\max} = 10^\circ$, $\Delta\chi_{\max} = 60^\circ$, $R_{\min} = 38m$ and the LRF parameters were set to $d_{LRF} = 80m$, $\Delta\theta = [-20°, 20°]$, $\Delta\psi = [-45°, 45°]$. The default metric is Dubins metric I. The radius of the sphere around waypoints is set to $R_{goal} = 15m$ and the threshold for proximity when evaluating Dubins metric I to $R_{goal\_Dubins} = 30m$.

4.1. A Deeper Insight Into the Local Planner

By definition the global planner returns an obstacle free path, i.e. intervisibility between waypoints is guaranteed. Therefore, any case in which intervisibility between the start position and goal position is not given, the obstacle blocking the line-of-sight can be considered unknown and will be referred to as unknown obstacle.

4.1.1 Dubins Metric Performance

First, we present a comparison of the 3 suggested metrics. Simulation results obtained when using the Euclidian distance serve as reference. Table 3 shows the number of node expansions necessary to reach the goal using each one of the metrics.

Case	Distance	n_{exp}	$n_{limited}$	$n_{limited} / n_{exp}$
1	-	10	0	0%
2	374m	9	5	55.56%
3	274.4m	6	5	83.33%
4	254m	30	11	36.67%
5	196.5m	6	0	0%

TAB 4. Applicability of the limited set of MPs

Heading information is necessary in test case 1 (figure 2) for the planner to find a solution, because the goal is located elevated and behind the aircraft's initial position. The Euclidean Distance does not provide such information and therefore the algorithm is not able to determine which branch to expand, since none of the candidates is the best evaluating J. Thus, when the condition in line 13 of algorithm 2 is violated, the planner aborts the planning process. The usage of Dubins curves in the horizontal plane incorporates heading information into the other 3 tested metrics. They perform similarly well. Using metric 3, the planner expands one more node compared to Dubins Metric I and Dubins Metric II.

Test case 2 (fig. 3) is trivial, but reveals the core message of the metric comparison. In terms of node expansions all metrics perform almost similarly well. However, a visual comparison reveals a very wiggly path for the Euclidean Metric compared to Dubins Metric I+II caused by lack of heading information in the Euclidean Metric.

Figure 4 demonstrates the difference in behavior of Dubins Metric I and III, if an unknown wall is introduced. The planner is able to encounter the goal using all 4 metrics. Dubins metric I, Dubins metric II and the Euclidean distance result in the same path, whereas metric 3 produces a slightly longer path. The property of metric to maintain the trajectory headed directly towards the goal 3 in the x-z plane can be observed.

4.1.1. Reducing Computational Effort in Obstacle Free Environments

To evaluate the performance of the Switching Maneuver Automaton, table 4 gives the number of node expansions n_{exp} and the number of times the limited set of motion primitives is used in test cases 1 through 5. The application of the limited set reduces the computational effort since fewer motion primitives have to be evaluated in each step. In test cases 2, 3 and 4 the limited set has been applied in at least one third of the node expansions. Difficult start-goal configurations or unknown obstacles need to make use of the full flight envelope of the UAV, therefore requesting all motion primitives. Such cases are test cases 1 and 5.

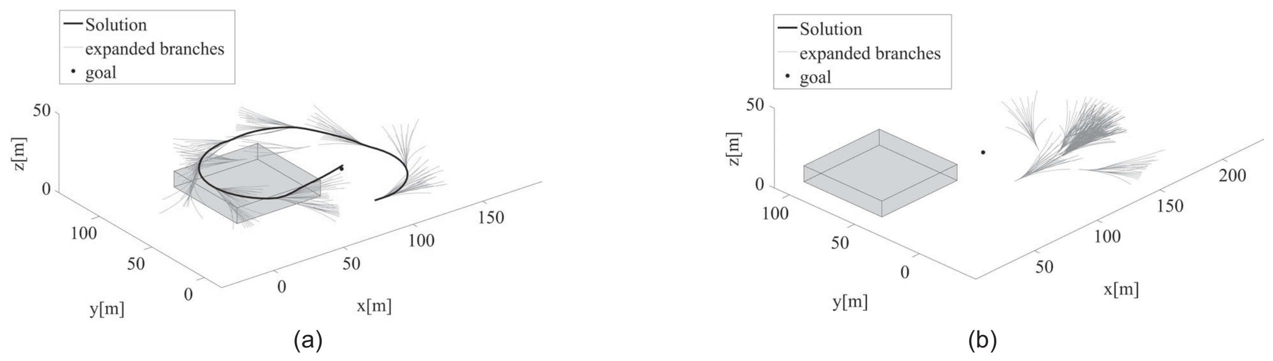


FIGURE 2. Test Case 1 shows the necessity to incorporate heading information. In (a) Dubins Metric I comes to a feasible solution, the Euclidean Metric in (b) fails to reach the goal with the given parameters.

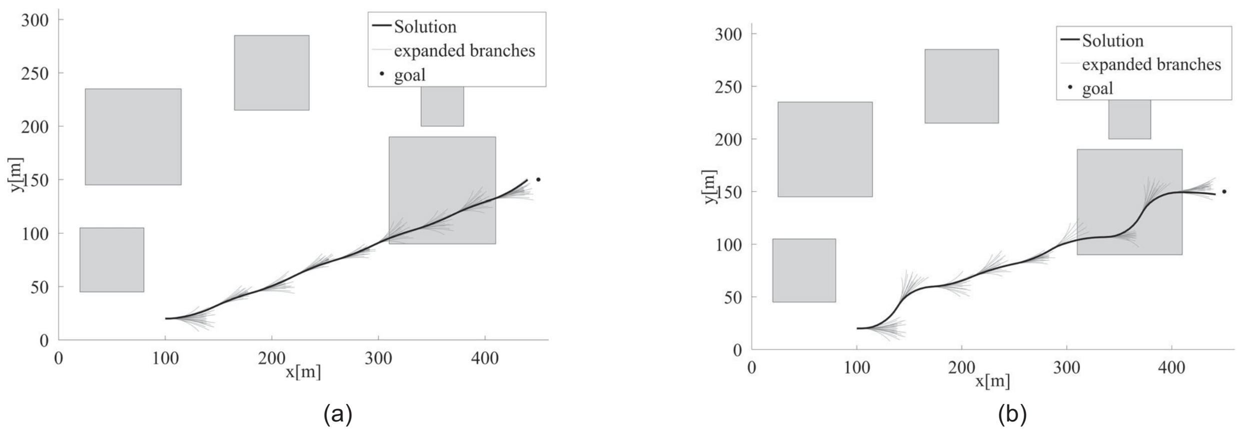


FIGURE 3. Test Case 2: For long obstacle free sections the comparison of Dubins Metric I (a) with the Euclidean Distance (b) reveals a meandering path for the latter which is in general not favorable.

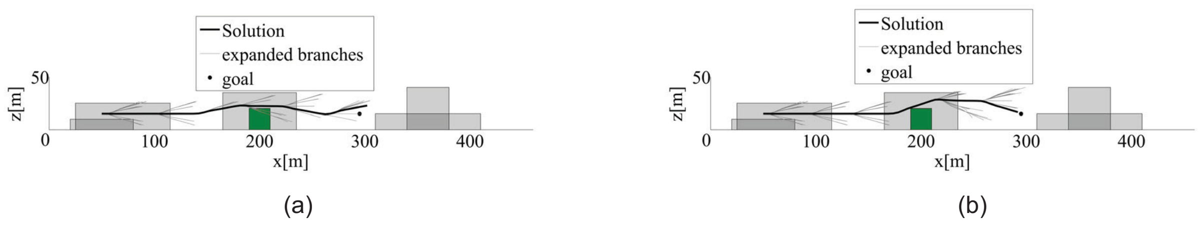
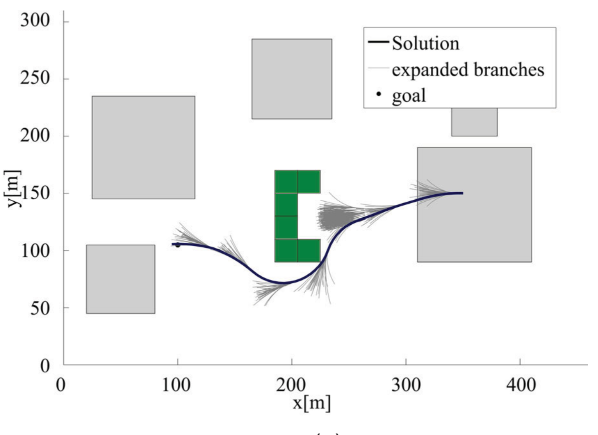


FIGURE 4. Test Case 3: Passing the unknown obstacle (green) from the x-y perspective is shown here. Figure (a) depicts a trajectory created with Dubins Metric I. The planning process is terminated successfully as the trajectory enters the goal region with no direction preference. The result of the additional incorporation of x-y plane direction information towards the goal in Dubins Metric III can be seen in (b).



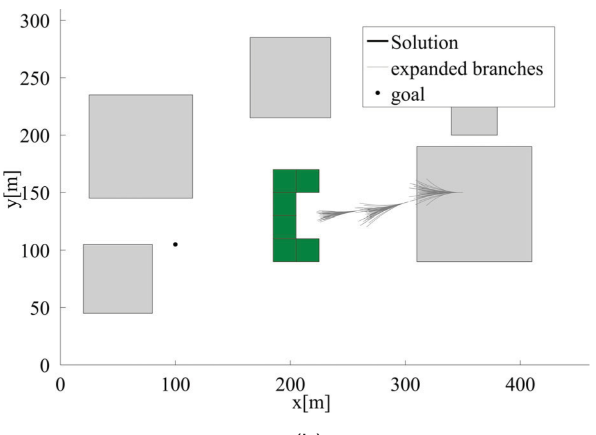


FIGURE 5. Test Case 4: A scenario with an unknown obstacle (green) with penalty function (a) and without penalty function (b) can be seen.

4.1.2. Obstacle Avoidance

We already showed basic unknown obstacle avoidance in test case 3. However, it is interesting to compare the performance of the planner in a more difficult scenario with the penalty function disabled. U-shaped obstacles present a special challenge to planning algorithms as the search tree can become dead-locked. In test case 4, depicted in figure 5, we introduce a large U-shaped obstacle with height $h_{obs} = 50m$. If the penalty function is disabled, the Open List is empty after 22 node expansion, leaving the planner with no more branches to explore, thus aborting planning. If the penalty function is used, it successfully biases the tree away from the obstacle.

Another case, which needs special attention, is a scenario with narrow passages, because the dynamic constraints of the aircraft make it difficult to enter canyons. Furthermore, obstacle avoidance techniques should not be too conservative, because otherwise goal positions, which are only reachable through canyons, will not be attained. Test case 5 depicted in figure 6 shows such a case. The local planner performs well and finds the connection through the canyon even if no line-of-sight connection between start and goal exists.

4.2. Integrated Planner

Test case 6 depicted in figure 7 presents a solution of the two-stage planning approach. The roadmap of the global planning stage consisted of 10000 nodes and the 500 nearest neighbors of each node were considered during roadmap construction. The red circles represent the waypoints, which were computed using the global planner. The blue solid line depicts the final solution of the local planner and the green U-shaped obstacle with height $h_{obs} = 35m$ was unknown during global planning. It intersects the edge between waypoint 5 and the goal rendering the global solution infeasible.

The initial heading of the aircraft points away from the goal. A U-turn towards waypoint 1 leads the aircraft towards the global path. At the first tree-expansion step the Dubins metric ensures, that the branch of the tree, which is the closest to waypoint 1 under the minimum turning radius constraint is expanded. As soon as the waypoint proximity condition (line 3 of algorithm 2) is fulfilled, waypoint 2 is set as next local goal. Then the nodes on the Open List are reevaluated (lines 5-6) and the best node with respect to waypoint 2 is expanded. It can be seen that this is not necessarily the node closest to waypoint 1.

As already shown in the previous section, the local planner is able to avoid the unknown obstacle using the obstacle avoidance techniques implemented. A feasible 3D trajectory, which avoids previously known as well as unknown obstacles by flying over them as well as around them, was found in this test case using our two-stage approach in an urban scenario.

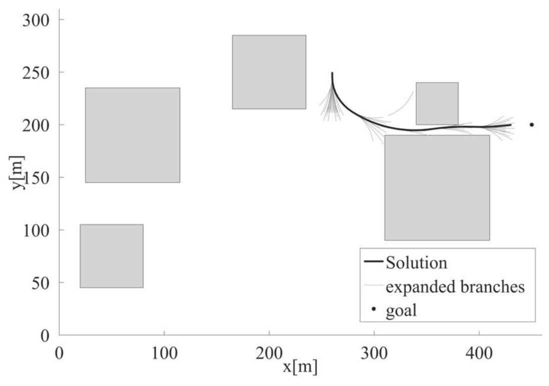


FIGURE 6.Test Case 5: Narrow passage

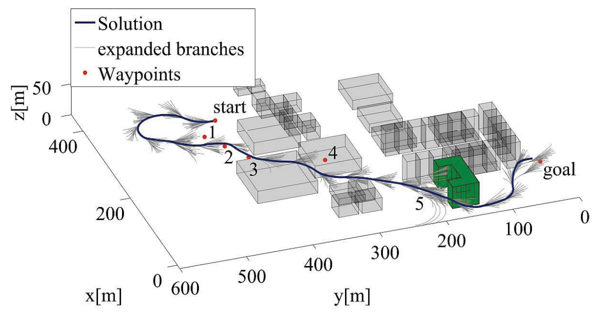


FIGURE 7.Test Case 6: Global + Local planning

5. DISCUSSION

5.1. Local Planner - Features and Failure Cases

The presented local planning algorithm works fast and returns smooth trajectories for a variety of cases. The switching maneuver automaton provides a feasible representation of the aircraft's dynamics and reasonably reduces the computational effort due to the switching property.

The choice of an appropriate metric was identified as a key component. A metric should be computationally cheap but also robust in a sense, that it does not lead to failure in unforeseen situations, such as pop-up obstacles or goals, which are positioned directly behind the aircraft. The Euclidean Distance is therefore not suitable. Even if the advantages of the Dubins metric are not of purely quantitative nature, as can be seen in table 2, its use results in paths of higher quality i.e. less unnecessary turning. To provide comparability throughout testing of the properties of the suggested planner we chose to implement Dubins Metric I as default metric.

The penalty function is computationally cheap but improves the performance in obstacle rich environment. We chose a completely deterministic approach for the local planner, due to the advantages of predictability and traceability. However, there is a trade-off between these advantages and completeness. Other than in the case of planners with probabilistic completeness guarantees the presented local planner is not guaranteed to find a solution if one exists. It is therefore necessary to carefully analyze the reasons for aborted planning.

Three failure modes exist, which cause the local planner to abort the planning process depicted in algorithm 2.

1) Empty Open List failure: If the last node on the Open List cannot be expanded successfully, planning is aborted in line 12. This failure mode occurs, if the search-tree gets entrapped by the environment. It can be countervailed by increasing the size of the Open List. Furthermore, the aforementioned penalty term lowers the probability of this failure mode.
    2) Expansion Failure: The threshold in line 13 is set to 50. Given that the global planner returns a set of waypoints, which are spaced by the length of at least 1 but on average 4 to 5 motion primitives in the given test environment, 50 tree expansion steps are sufficient to plan a trajectory between two consecutive waypoints. Thus, a violation of this limit can have two reasons:
    a. The sequence of waypoints produced by the global planner is not kinematically feasible. This can occur, when planning in maze-like environments with sharply cornered canyons. This is the most common reason for failure, because the sequence of waypoints cannot be tracked by the local planner.
    b. Unknown obstacles are discovered blocking the globally planned path. In this case the local planner plans around the obstacle. However, unfavorable obstacle geometries such as U-shaped obstacles or wedges, which drive the tree away from the waypoint, can result in this failure mode. The penalty term countervails the occurrence of this failure mode.

    If the waypoint, which cannot be reached within the limits, is not the global goal, the planner tries to skip this waypoint in line 15.
3) Propagation failure: If the node, the aircraft is supposed to move to during the propagation procedure in line 20, is equivalent to the current best node, the aircraft would move to a node, which has not been successfully expanded yet. We choose to abort planning in this case, as no information exists if this node can be expanded successfully. This failure mode is caused by large unknown obstacles such as walls preventing the aircraft from reaching a waypoint. This failure mode is closely related to the failure mode presented in 2.(b)

Considering the failure modes, it can be observed that failure mode 2.(a) is caused by problems concerning the combination of both planners. Further research will address this issue. The other failure modes, however, are due to unknown obstacles or very difficult obstacle configurations. This might be tolerable, because the aircraft's safety can be guaranteed through the safety maneuvers. Failure mode 2.(a) also does not compromise the safety of the aircraft, as the safe execution of one of the safety maneuvers is always guaranteed.

5.2. Integration of Global and Local Planner

The integration of the global planner and the local planner into a two-stage motion planning framework has been outlined and a simulation in a difficult scenario has been performed. The overall performance is good, i.e. feasible trajectories between random start and goal configurations can be generated reliably for a wide variety of environments, start and goal configurations.

We found it difficult to carefully account for the kinematic constraints in the global planner due to the random sampling approach used by the PRM. However, the results we obtained were satisfying. Deterministic sampling during roadmap generation [16] seems promising to directly address kinematic feasibility constraints and improve the performance of the global planner. However, a kinematically unfeasible path does not necessarily lead to failure of the integrated planner if the environment allows for additional turns of the aircraft. Overall, the integrated planning approach performs well and is able to successfully address narrow passages as well as unknown obstacles. It also exhibited promising runtime behavior in the test runs we conducted.

6. CONCLUSION

We presented a deterministic tree-based local planner for a fixed-wing UAV in detail. It is part of the two-stage planning approach presented in [1]. The Dubins metric was shown to outperform the Euclidian metric in terms of visible path quality. Furthermore, augmenting the heuristic with environment information such as obstacle proximity increased the success rate of the planner significantly. We conducted simulations of the integrated planner in difficult terrain and obtained promising results. The presented local planner as well as the framework is a step towards a computationally inexpensive online planner for small UAVs which are able to function safely in a varying urban environment. Further details can be found in Ref. [15].

7. REFERENCES

[1] Kavraki, L. E., Svestka, P., Latombe, J. C., et al.: “Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces”, IEEE Transactions on Robotics and Automation, Vol. 12, No. 4, August 1996

[2] Andert, F., Adolf, F.-M.: „Online World Modelling and Path Planning for an Unmanned Helicopter“, Autonomous Robots(27), 2009, Springer Netherlands, pp. 147-164

[3] LaValle, S.M., Kuffner, J.J.: “Randomized Kinodynamic Planning”, Proceedings of the 1999 IEEE Int. Conf. on Robotics and Automation, IEEE, 1999

[4] Frazzoli, E., Dahleh, M. A., Feron, E.: “Real-Time Motion Planning for Agile Autonomous Vehicles”, Journal of Guidance, Control and Dynamics, Vol. 25, No. 1, 2002, pp. 116-129

[5] Hwangbo, M., Kuffner, J., Kanade, T.: „Efficient Two-Phase 3D Motion Planning for Smal Fixed-Wing UAVs“, Proceedings of the 2007 IEEE Int. Conf. on Robotics & Automation, IEEE, 2007

[6] Gros, M., Niendorf, M., Schöttl, A., Fichter, W.: „Motion Planning for a Fixed-Wing MAV Incorporating Closed-Loop Dynamics Motion Primitives and Safety Maneuvers“, Advances in Aerospace Guidance, Navigation and Control, Springer Verlag, Berlin, 2011, pp. 241-260

[7] Saunders, J., Call, B., Curtis, A. et al.: “Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles”, AIAA Infotech@Aerospace, Arlington VA, pp. 2005-6950, 2005

[8] Scherer, S., Singh, S., Chamberlain, L. et al.: “Flying Fast and Low Among Obstacles”, International Conference on Robotics and Automation (ICRA), pp. 2023-2029, 2007

[9] Gottschalk, S., Lin, M., Manocha, D.: “OBBTree: A Hierarchical Structure for Rapid Interference Detection”, Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, pp. 171-180, ACM New York, NY, USA, 1996

[10] Dubins, L.: “On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents,” American Journal of Mathematics, vol. 79, no. 3, pp. 497–516, 1957

[11] Enright, J., Frazzoli, E., Savla, K., Bullo, F.: “On Multiple UAV Routing with Stochastic Targets: Performance Bounds and Algorithms”, Proceedings of the AIAA Conference on Guidance, Navigation and Control, 2005

[12] Chitsaz, H., LaValle, S. M.: „Time-optimal paths for a Dubins airplane“, Proceedings IEEE Conference Decision and Control, 2007

[13] Hart, P., Nilsson, N. J., Raphael, B.: “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”, IEEE transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100-107, 1968

[14] Kehoe, J., Watkins, A., Lind, R.: “A Time-Varying Hybrid Model for Dynamic Motion Planning of an Unmanned Air Vehicle” AIAA Guidance, Navigation and Control Conference, Keystone, CO, August 2006

[15] Niendorf, M.: ”Two-Stage Motion Planning for a Fixed-Wing UAV Incorporating A Tree-Based Local Planner With Motion Primitives”, Thesis, Institute of Flight Mechanics and Control Universitaet Stuttgart, 2010