A Rapid Incremental Motion Planner for Flexible Formation Control of Fixed-Wing UAVs

Chang Boon Low  
DSO National Laboratories, Singapore  
Email: lchangbo@dso.org.sg

Abstract—Motion planning for a formation of nonholonomic fixed-wing UAVs in environments with obstacles is a challenge especially when the operating envelope of the platform’s is considered in the planning problem. This paper presents a rapid formation motion planning algorithm to determine a formation leader trajectory between a start pose and a final pose in the presence of stationary obstacles (static no-fly-zones) based on concatenated Dubins curves. The planner encodes formation configuration (shape) and UAV operating envelope in the planning problem to ensure that the computed formation leader trajectory is flyable by the formation. The effectiveness of the planner was evaluated in simulation using our recently proposed flexible formation keeping control scheme based on six degree-of-freedom (6DOF) fixed-wing UAVs models [1].

I. INTRODUCTION

Formation control of multiple unmanned vehicles has received considerable attentions due to its numerous useful applications and technical challenges of the problem. In general, formation control approaches can be grouped into behavioral [2]-[3], leader-following [4]-[5], and virtual structure [6]-[10], where each of these approaches has its own advantages and disadvantages.

Virtual structure (V-S) approach can be further classified into: 1) rigid virtual structure [6]-[8], and 2) flexible virtual structure approaches [9]-[10]. Rigid virtual structure approach defines formation positions based on each vehicle’s assigned relative rectilinear vector with respect to the virtual structure centre point (virtual leader position). One limitation of this approach is that the rigidity of the virtual structure affects the formation’s turning performance [7], in particular when the formation is manoeuvring along a virtual leader trajectory that possesses discontinuous curvature, e.g., Dubins trajectory. On the other hand, flexible virtual structure defines a flexible virtual structure using relative curvilinear coordinates with respect to the virtual structure centre point. With this notion, the virtual structure can be curve compliant along formation leader trajectories with constant relative curvilinear coordinates. This feature allows a formation of nonholonomic vehicles to achieve better heading control and predictability when executing a turn along a planned virtual leader trajectory compared to its rigid counterpart. This flexible V-S concept has been applied to mobile robots [9]-[10], and recently, this concept was extended to formation control of fixed-wing UAVs [1].

In the literature, other related formation control problems can also be found [10], [11], [12], [14], [15]. In [14] and [15], researchers developed formation reconfiguration solutions to transform formation configuration to avoid obstacles along a given virtual leader trajectory. In contrast to formation reconfiguration, the focus of this work is to develop a formation motion planner that computes the formation leader trajectory. We called this complementary problem as formation motion planning problem [10],[11],[12]. To plan a path for a formation, one straightforward approach is to treat the considered formation as a rigid body and plan a path for the formation to travel. Planning algorithms that are based on this strategy can be found in the literature [10],[11],[12]. In these works, grid-map based formation path planners are proposed to determine path between two specified points for a formation of mobile robots to navigate in static environments. One advantage of this strategy is that it provides a direct means to generate a path for the formation to travel. However, one major drawback of this strategy is that it ignores the kinematic motion limitations imposed by the formation configuration and the platform’s operating envelope. As a result, this strategy does not ensure that the vehicles’ reference velocities required to maintain the desired formation do not exceed the vehicles’ operating envelopes during the formation maneuvers, in particular during turning manoeuvres. Another limitation of these works is that grid-map based planners may not be suitable for fast computation of executable trajectories planning for nonholonomic vehicles, in particular in a large environment. To generate an executable virtual leader trajectory for nonholonomic vehicles, we need to ensure that the reference velocity of each vehicle of the formation does not exceed the vehicle’s operating envelope, and at the same time satisfies the nonholonomic kinematic constraint. These considerations are particularly critical for fixed-wing UAVs since the vehicle is nonholonomic and operating near the UAVs stalled speeds may result in a catastrophe.

In this paper, an attempt is made to exploit some insights gained in our recent work [1] to develop a rapid incremental formation motion planning algorithm to compute executable formation leader trajectory between two specified poses for a formation of fixed-wing UAVs to travel, assuming the formation configuration remains constant. This computed nonholonomic feasible leader trajectory avoids violation of the UAVs operating envelopes during the formation flight, and at the same time avoiding static obstacles (or no-fly-zone) in the environment.

II. PROBLEM FORMULATION
A. Flexible Virtual Structure Formation Control of Fixed-Wing UAVs

In this work, we consider the formation motion planning problem for flexible formation of fixed-wing UAVs. In contrast to rigid formation schemes where the controllers aim to maintain constant rectilinear separations between the vehicles, flexible formation scheme allows the geometrical distances between the UAVs to vary slightly when the formation is executing a turning manoeuvre even though the relative curvilinear coordinates that define the formation configuration remains unchanged. Figure 1 depicts a V-shaped formation configuration when it is traveling along a nonlinear virtual leader trajectory, while Figure 2 depicts the same formation configuration when it is traveling along a linear virtual leader trajectory. Here we called the formation virtual leader trajectory as the formation group trajectory. Formation group trajectory represents the trajectory that the formation is planned to travel and it is assumed that it satisfies the nonholonomic kinematic constraint. The virtual structure centre point (virtual leader’s position) is the reference point of the desired formation where the reference position of each UAV \( U_i \) of the formation is defined by a relative curvilinear coordinate \((p_i, q_i)\) with respect to the virtual leader’s position. In this work, \((x, y, \psi)\) denotes the pose of the virtual leader (see Figure 1 and Figure 2). This definition of UAV formation position enables us to compute a curve compliant formation reference trajectory that satisfies the nonholonomic kinematic constraint of each UAV if the formation group trajectory satisfies the nonholonomic constraint [1]. When every UAV executes its respective formation reference trajectory stably, the flexible formation motion control is achieved and maintained along the planned formation trajectory.

B. Obstacles Assumptions

In this work, the obstacles or the no-fly zones (NFZ) in the environment are assumed to be static and can be modeled by obstacle circles. It will be shown in Section III that the arcs of these obstacle circles are exploited as feasible formation motion primitives to construct a feasible nonholonomic trajectory for the formation. For a polygonal obstacle \( C_n \), we can cover the obstacle with a circle \( C_n \) of suitable radius and position such that \( C_n \subset C_n \) (see Figure 3). Assuming that the polygonal obstacle \( C_n \) is represented by \( m_n \) points \( \{p_1, \ldots, p_{m_n}\} \), one simple and direct way to approximate it is to utilize an obstacle circle with a center point defined as \( p_{cp} = \frac{1}{m_n} \sum_j p_j \), and a radius of \( \max_j \|p_{cp} - p_j\| \). \( \bullet \) denotes the Euclidean norm. In the remainder of this paper, it is assumed that the set of all obstacles \( O \) considered in the environment is covered and modelled by \( m \) isolated obstacle circles, i.e., \( O \subset \cup_{l=1,\ldots,m} C_l \).

C. Problem Description

In this work, we consider a formation of \( N \) nonholonomic fixed-wing UAVs where the notations of each UAV \( U_i \) for \( i \in \{1, \ldots, N\} \) are depicted in Figure 4. Let \((x_i, y_i)\) denotes the inertial position of UAV \( U_i \), \( \psi_i \) represents the heading angle, \( X_b_i \) and \( Y_b_i \) denote the UAV body axes, \( X \) and \( Y \) denote the inertial axes, \( V_i \) denotes the airspeed, and \( \omega_i = \psi_i \) denotes the turning rate of the UAV. The motion of each UAV satisfies kinematic constraint \(-\dot{x}_i \sin \psi_i + \dot{y}_i \cos \psi_i = 0\), and has an operating airspeed envelope of \( 0 \leq V_{\min} \leq V_i \leq V_{\max} \) and a turning-rate envelope of \( |\omega_i| \leq \omega_{\max} \). \( V_{\min} \) and \( V_{\max} \) denote the minimum and maximum airspeeds, and \( \omega_{\max} \) denotes the maximum turning rate of the UAV.

The focus of this work is to develop a rapid formation motion planner to plan a flyable formation group trajectory.
to guide a flexible formation of fixed-wing UAVs between two specified poses without violating the operating envelope of each UAV. This motion planner allows the formation to respond to unmapped obstacles that are detected during the flight. The formation planning problem can be stated as follows.

Assuming an environment where the obstacles are modeled by \( m \) isolated obstacle circles with radius \( R_j \) for \( j \in \{1, \ldots, m\} \), and assuming that the desired N-UAV formation \( \mathcal{F} \) is defined by \( \{p_i, q_i\} \) for \( i \in \{1, \ldots, N\} \) with a constant speed of \( V_{cp} \), determine a formation group trajectory that connects between a start pose \( Q_s \) and a final pose \( Q_f \) such that all the respective coordinated formation reference trajectories of the \( N \) fixed-wing UAVs satisfy \( 0 \leq V_{min} \leq V_{ri} \leq V_{max} \) and \( |\omega_{ri}| \leq \omega_{max} \) for \( i \in \{1, \ldots, N\} \), and do not result in any collision between a UAV of the formation and an obstacle circle during the flight.

\( V_{cp} \) denotes the desired constant velocity of the formation, where \( (V_{ri}, \omega_{ri}) \) denotes the reference velocity and reference turning rate of each UAV \( U_i \) along the formation group trajectory.

### III. Formation Motion Planner for Flexible Formation of Fixed-wing UAVs

#### A. Formation Motion Planning Strategy

In this article, we are developing a low runtime nonholonomic feasible formation group trajectory planner for a formation of nonholonomic fixed-wing UAVs to navigate among static obstacles. Here, we trade off optimality for computation speed, and at the same time we want the algorithm to achieve efficient trajectory. One strategy to achieve this aim is to exploit the concepts of delayed collision checking and incremental planning, that are commonly used in addressing path planning problems for mobile robots [16], [17] and UAVs in the presence of obstacles [18]. The strategy we proposed here first determines a reasonably good trajectory without considering the obstacles, then followed by performing collision checks on this trajectory to determine if the path collides with the obstacles. A replanning on some selected segments of the trajectory will be performed to repair the trajectory if its leads to a collision with the obstacles. This process repeats itself until a complete collision-free trajectory is determined. In this manner, low computation time would be required to obtain a collision-free trajectory for the formation to travel between two poses.

To implement this strategy to address the formation motion planning problem, one approach is to utilize Dubins curves as the motion primitives to construct the formation group trajectory incrementally. Computing a Dubins curve is computational inexpensive and the generated segments are nonholonomic kinematic feasible. Here, we exploit the concept of generalized minimum formation turning radius, \( R_{min}^F \), proposed in [1] to determine a feasible Dubins based formation group trajectory that connects the start pose \( Q_s \) and the final pose \( Q_f \). \( R_{min}^F \) is the minimum turning radius that the flexible formation is able to turn without violating the UAV airspeed and turning rate envelopes, i.e., the required reference velocity of each UAV to maintain the formation satisfies \( V_{ri} \in [V_{min}, V_{max}] \) for \( i = 1, \ldots, n \) [1]. To incorporate this turning constraint to the planning problem, we redefine the radius of some selected obstacle circles accordingly to \( R_{min}^F \). The planning algorithm can be briefly summarized as follows.

First, we determine an optimal Dubins trajectory that connects \( Q_s \) to \( Q_f \). If the trajectory does not collide with any obstacle circle in the environment, then the optimal Dubins trajectory will be chosen as the final formation group trajectory that connects between the two poses. However, if there is an intersection between the group trajectory and an obstacle circle, then the trajectory replanning is carried out to repair the selected segments to avoid the intersected obstacle circles.

Dubins trajectory planner has been applied to generate constant speed nonholonomic kinematic feasible trajectories for a single nonholonomic vehicle [13][19][20]. Essentially, a Dubins curve is a 3-segment curve that connects between two specified poses, and each segment of a Dubins curve is either a circle arc or a straight line (see Figure 5). For every pair of specified poses, there is a set of six possible Dubins curves: \{RSR, RSL, LSL, LSR, RLR, LRL\}. \( R \) denotes a right turn maneuver, \( S \) denotes a straight maneuver, and \( L \) denotes a left turn maneuver. The left and right turn maneuvers are based on two turning circles located at the left and right sides of the pose (see Figure 5). \( C_{Start}^+ \) and \( C_{Start}^- \) denote the left and right turning circles of the start pose, and \( C_{Final}^+ \) and \( C_{Final}^- \) denote the left and right turning circles of the final pose. The figure depicts a \( LSR \) Dubins curve connecting from the start pose to the final pose. To determine the optimal Dubins trajectory that connects a pair of specified poses, the algorithm selects the shortest-length curve from the set of six possible Dubins curves. Readers can refer to [19] for more details about the Dubins planner that has been applied to fixed-wing UAVs.

![Optimal Dubins Trajectory](image-url)

To replan the Dubins-based trajectory, we exploit segments of circle arcs and lines to repair the initial optimal trajectory since these segments can be easily concatenated using Dubins segments to form a new nonholonomic kinematic feasible trajectory. Here, the circular arc of the obstacle circle is used to provide the intermediate turning segment that allows the formation to traverse while avoiding the obstacle. This concept is illustrated with a simple example depicted in Figure 6. In the figure, the optimal Dubins...
Fig. 6. Two-step local trajectory replan strategy. Step 1 (Left): determines a partial 2-segment Dubins curve, Step 2 (Right): determines a 3-segment modified Dubins curve and combines the two curves.

trajectory (see Figure 5) intersects the obstacle circle and it can be repaired by a two-step implementation to detour the intersected obstacle circle. At Step 1, the planner determines a 2-segment Dubins curve which connects the start pose to the left tangent point of the selected obstacle circle to avoid the collision with the circle. The left of Figure 6 illustrates this step. Once a collision-free 2-segment Dubins curve is determined, the planner determines the remaining segments that connects from the left tangent point to the final pose. Here, the planner plans the remaining trajectory to the final pose using a 3-segment Dubins curve. This connection is depicted on the right of Figure 6. The first segment of this 3-segment Dubins curve is constrained to a $R$ turn (clockwise turn) with the radius of the intersected obstacle circle that the formation is trying to go around. This constraint limits the possible Dubins maneuvers to a set of 3 possible maneuvers $\{RLR, RSR, RSL\}$. The shortest-length curve of this reduced set will be selected as the remaining trajectory to the final pose.

The EXPAND procedure implements the obstacle expansion concept by increasing the radius of each obstacle circle, so that the formation can travel around them without having a collision between a UAV and the obstacle circles $\mathcal{O}$. The inputs of the EXPAND procedure are the unexpanded obstacle circles, the minimum formation turning radius of formation $\mathcal{F}$, the maximum width of the formation $q_{\text{max}}$, and the error buffer $\epsilon > 0$. The output of this procedure is the expanded set of obstacle circles $\mathcal{O}$. The radius $R_j$ denotes the radius of the expanded circle $\mathcal{C}_j$. $\epsilon$ is the additional buffer that is used to cater for other errors such as control errors or obstacles modeling errors. In this way, the UAVs will not collide with the obstacle circles when the formation is traveling along the boundaries of the expanded circles.

B. Pseudo Code of Formation Motion Planning Algorithm

The remainder of this section presents the pseudo code of the formation motion planning algorithm that implements the strategy that is briefly discussed in Section III-A. One assumption we assumed in this work is as follows.

Assumption 1: There is no intersection between an expanded obstacle circle $\mathcal{C} \in \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$ and a turning circle $\mathcal{C} \in \{\mathcal{C}_{\text{Start}}, \mathcal{C}_{\text{Start}}, \mathcal{C}_{\text{Final}}, \mathcal{C}_{\text{Final}}\}$.

Assumption 1 is imposed to ensure there exists a Dubins-based trajectory between the start and final poses in the presence of expanded obstacle circles $\mathcal{O}$. With this assumption in mind, the formation motion planning algorithm is presented as follows.

Algorithm Formation Motion Planning Algorithm

Input:
\{$Q_s, Q_f$\}: start and final poses
\{$p_i, q_i$\}: relative coordinates for $i \in \{1, \ldots, N\}$
$O$: obstacle circles

Output:
$\tau$: Formation Group Trajectory

1: $\tau \leftarrow \text{Set Empty}$
2: $\mathcal{O} \leftarrow \text{EXPAND}(\mathcal{O})$
3: Flag $\leftarrow 0$

PROCEDURE EXPAND

Input:
$\mathcal{O} = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$: obstacle circles

$q_{\text{max}}$: maximum width of formation $\mathcal{F}$
$R_{\text{min}}$: formation turning radius
$\epsilon$: error buffer

Output:
$\mathcal{O} = \{\mathcal{C}_1, \ldots, \mathcal{C}_m\}$

1: for each $j \in \{1, \ldots, m\}$ do
2: $\bar{R}_j \leftarrow R_j + q_{\text{max}} + \epsilon$
3: if $R_{\text{min}} \leq \bar{R}_j$
4: $\bar{R}_j \leftarrow R_j$
6: else
7: $\bar{R}_j \leftarrow R_{\text{min}}$
7: end if
8: end for
9: return
4: \( \tau \leftarrow \text{DUBINS}(Q_s, Q_f, R_{F \text{min}}^F) \)
5: \( \text{if} \ \text{TESTSEG}(O, \tau) == \text{false} \)
6: \( \text{Flag} \leftarrow 1 \)
7: \( \text{return} \ \tau \)
8: \( \text{else} \)
9: \( Q_{st} \leftarrow \text{STARTPOSE}(\tau) \)
10: \( \tau_p \leftarrow \tau \)
11: \( \tau \leftarrow \text{Set Empty} \)
12: \( \text{while} \ \text{Flag}==0 \)
13: \( \text{repeat} \)
14: \( k \leftarrow \text{OBS.ID}(\bar{O}, \tau_p) \)
15: \( \tau_p \leftarrow \text{PDUBINS}(C_k, Q_{st}) \)
16: \( \text{until} \ \text{TESTSEG}(O, \tau_p) == \text{false} \)
17: \( \text{ADDSEG}(\tau, \tau_p) \)
18: \( Q_{st} \leftarrow \text{ENDPOSE}(\tau_p) \)
19: \( \tau_m \leftarrow \text{MDUBINS}(Q_{st}, Q_f, C_k) \)
20: \( \text{if} \ \text{TESTSEG}(\bar{O}, \tau_m) == \text{false} \)
21: \( \text{Flag} \leftarrow 1 \)
22: \( \text{ADDSEG}(\tau, \tau_m) \)
23: \( \text{return} \ \tau \)
24: \( \text{end if} \)
25: \( \tau_p \leftarrow \tau_m \)
26: \( \text{end while} \)
27: \( \text{return} \ \tau \)
28: \( \text{end if} \)

Procedure \text{DUBINS} computes the initial optimal Dubins trajectory based on the minimum formation turning radius \( R_{F \text{min}}^F \). Procedure \text{TESTSEG} performs a simple 2D line-circle intersection test to check whether the linear segment of \( \tau \) collides with any expanded obstacle circle. If a collision is detected, then the procedure returns a \text{true}, otherwise, it returns a \text{false}. Procedure \text{OBS.ID} identifies and returns the index \( k \) of the nearest intersecting expanded obstacle circle from the beginning pose of the linear segment of \( \tau_p \). Procedure \text{ADDSEG}(\tau, \tau_m) \) augments the newly computed Dubins segments \( \tau_m \) incrementally to the trajectory \( \tau \). Procedure \text{STARTPOSE} and procedure \text{ENDPOSE} take a Dubins curve as its argument and return the start and end poses of the curve. Procedure \text{EXPAND} essentially implements the obstacle circles expansion strategy presented in Section III-A. Procedure \text{PDUBINS} implements Step 1 of the obstacle avoiding strategy to seek a 2-segment Dubins curve \( \tau_p \) that connects from \( Q_{st} \) to the left tangent point of the expanded obstacle circle \( C_k \) (see left of Figure 6). Procedure \text{MDUBINS} implements Step 2 of the obstacles avoiding strategy by computing the incremental formation trajectory \( \tau_m \) from \( Q_{st} \) to final pose \( Q_f \).

IV. SIMULATION

In this implementation, the proposed formation motion planning algorithm is implemented using MATLAB on a computer that is equipped with a 2.66GHz CPU. The desired formation speed is \( V_{cp} = 25 \text{msec}^{-1} \), and each fixed-wing UAV we considered here has an operating envelope of \( V_{min} = 18 \text{msec}^{-1} \), \( V_{max} = 34 \text{msec}^{-1} \), and \( \omega_{max} = 0.05 \text{rad/sec}^{-1} \). The configuration of the desired formation is depicted in Figure 8. The formation has a maximum formation width of \( q_{max} = 25 \text{ m} \), and thus the formation has a minimum turning radius of \( R_{F \text{min}}^F = 500 \text{ m} \). We let \( \epsilon = 20 \text{ m} \) and consider five obstacle circles in the environment. The five circles \( \{C_1, C_2, C_3, C_4, C_5\} \) were located at \( X(m)-Y(m) \) coordinates \( \{(-1, -1), (-3, 0), (0, 0.7), (-2, 2), (2, 1)\} \times 10^3 \), and have the radii of \( \{R_1, R_2, R_3, R_4, R_5\} = \{0.8 \ 0.5 \ 0.7 \ 0.8 \ 0.4\} \times 10^3 \text{ m}. \) The radii of the expanded obstacle circles are \( \{R_1 \ R_2 \ R_3 \ R_4 \ R_5\} = \{0.845 \ 0.545 \ 0.545 \ 0.7450 \ 0.5\} \times 10^3 \text{ m}. \) These expanded circles are represented in dashed lines shown in Figure 9. The start pose of the formation is \( Q_s = (-3000 \text{ m}, -3000 \text{ m}, 0 \text{ rad}) \), and the final pose is \( Q_f = (4000 \text{ m}, 4000 \text{ m}, 0 \text{ rad}) \). The mean computation time taken to compute the formation group trajectory is 10 msec, and the computed trajectory is represented by the solid line in Figure 9.

The computed formation group trajectory was evaluated using our recently proposed flexible V-S formation keeping control scheme, together with nonlinear 6DOF fixed-wing models implemented in this simulation [1]. Figure 10 depicts the formation tracking control errors of the 4 UAVs when it is travelling along the computed trajectory. The initial longitudinal, lateral, and heading control errors set in this
simulation were \((x_{ei}, y_{ei}, \psi_{ei}) = (-10 \text{m}, 10 \text{m}, 0.1 \text{rad})\) for \(i = 1, 2, 3, 4\). The result shows that the formation keeping control scheme stabilized the initial errors and maintained in the desired formation stably throughout the flight. The steady-state errors are bounded by \(|x_{ei}(\infty)| = 1.7 \text{m}, |y_{ei}(\infty)| = 13 \text{m},\) and \(|\psi_{ei}(\infty)| = 0.1 \text{rad}\) for \(i = 1, 2, 3, 4\). The control errors that occurred at \(t = 120\text{sec}\) and \(t = 270\text{sec}\) are due to the discontinuities in curvature at the connecting points between any two consecutive Dubins segments. This curvature discontinuity between two Dubins segments is a feature of Dubins curve. The simulation results suggest that the planner generates feasible formation group trajectory for the 4-UAV formation to execute stably without violating the UAVs airspeed and turning rate envelopes. The results also depict stable heading control responses despite the formation group trajectory possesses discontinuous curvature. In this simulation, an obstacles collision detection was implemented to check for collision and there was no collision reported between the UAVs and the obstacles during the simulated flight.

V. CONCLUSIONS

In this paper, a formation motion planning algorithm was presented to provide a means to compute a flyable formation trajectory between two specified poses to navigate a formation of fixed-wing UAVs in an environment with static obstacles. This algorithm incorporates the UAVs operating envelopes and formation configuration to determine a collision-free and nonholonomic kinematic feasible formation group trajectory. It was shown in the simulation that the computation time required to determine a formation group trajectory is relatively low, and thus suggesting that the algorithm can be applied in near real-time applications where the formation needs to replan a feasible formation trajectory rapidly to react to detected obstacles that are not known a priori. In the simulation, we implemented our flexible virtual structure formation control scheme to execute the computed formation group trajectory. The simulation results suggest that the proposed formation motion planner generates flyable formation group trajectory to guide a formation of fixed-wing UAVs between two specific poses in the presence of static obstacles.

REFERENCES