Article

Path Planning Using Concatenated Analytically-Defined Trajectories for Quadrotor UAVs †

Jonathan Jamieson * and James Biggs

Mechanical and Aerospace Engineering, University of Strathclyde, 75 Montrose Street, Glasgow G1 lXJ, UK; E-Mail: james.biggs@strath.ac.uk

† This paper is an extended version of our paper published in ECC 15, Linz, Austria.

* Author to whom correspondence should be addressed; E-Mail: jonathan.jamieson@strath.ac.uk; Tel.: +44-141-574-5036.

Academic Editor: Konstantinos Kontis

Abstract: This paper presents a semi-analytical trajectory planning method for quadrotor UAVs. These trajectories are analytically defined, are constant in speed and sub-optimal with respect to a weighted quadratic cost function of the translational and angular velocities. A technique for concatenating the trajectories into multi-segment paths is demonstrated. These paths are smooth to the first derivative of the translational position and pass through defined waypoints. A method for detecting potential collisions by discretizing the path into a coarse mesh before using a numerical optimiser to determine the point of the path closest to the obstacle is presented. This hybrid method reduces the computation time when compared to discretizing the trajectory into a fine mesh and calculating the minimum distance. A tracking controller is defined and used to show that the paths are dynamically feasible and the typical magnitudes of the controller inputs required to fly them.

Keywords: UAV; trajectory planning; quadrotor; obstacle avoidance; sub-Riemannian curves

---

1. Introduction

Unmanned aerial system (UAS) applications often require the vehicle to fly in areas with many obstacles. With this desire to fly in restricted space, such as urban environments, there is increasing need for better path and trajectory planning. Typical UAS civilian applications, such as data collection,

environmental monitoring and security, require autonomous navigation to avoid collisions without the assistance of a pilot.

There are a variety of different UAS types, including fixed-wing planes, helicopters and multi-rotors. For agile navigation in small areas with tight space constraints or a dense field of obstacles, quadrotors are most suitable. Their ability to hover in a stationary position is also beneficial when awaiting instructions or collecting data using on-board sensors. In contrast, fixed-wing vehicles can efficiently cover greater distances and have faster maximum speeds. However, they cannot hover (barring experimental vehicles, such as [1]), which limits their use in certain scenarios. Additionally, their dynamic constraints restrict their manoeuvrability, and they must maintain a minimum airspeed to produce sufficient lift. Finally, helicopters offer similar performance and agility to multi-rotors, but their mechanical complexity is not justified at the scale of standard UAS. This paper will focus on trajectory generation for multi-rotors (specifically, quadrotors), because they have the least dynamic constraints; however, future work will investigate applying an extension of the methods to other vehicles.

The differential flatness of quadrotor dynamics can be exploited to allow for easier planning using the decoupled translational axes [2]. A yaw angle also needs to be defined; in the literature, this is normally fixed at zero. One technique for trajectory planning is to use an analytical function to describe the position over time. The shape of the function is altered by varying parameters. After defining an error function for the final state, a numerical minimization is performed to calculate the parameters that satisfy the boundary conditions. The derivatives of the state can be found by differentiating the analytical function. Functions that have already been used for quadrotor trajectory planning include splines [3], polynomials [4] and Bézier curves [5]. In this paper, we use the maximum principle of optimal control to define curves using standard functions that are sub-optimal with respect to a weighted quadratic cost function. The term sub-optimal is used as the analytical curves are the projection of optimal curves on $SE(3)$ onto R^3, and the rotational component of the optimal motion is not used in the motion planning.

A typical method of defining obstacles is to use p-norms [6] and is applicable in both two- and three-dimensional planning. This method can also be extended to polygonal shapes [7]. However, for the purpose of demonstrating the obstacle detection algorithm, the obstacles in this paper are defined as spheres. Obstacle avoidance using Dubins curves and sampling-based planning is considered in [8]. Methods for avoiding other moving vehicles and obstacles are given in [9,10].

This paper extends our conference paper [11] to develop concatenation of trajectories to form multi-segment paths and the obstacle detection algorithm. The format of this papers is as follows. The quadrotor dynamics and kinematics are presented in Section 2, with the analytical function defining the curves given in Section 3.1. The method for repositioning the generated curves is described in Section 3.2, and forming a single path from curves patched together is demonstrated in Section 5.2. An obstacle detection algorithm is presented in Section 4 and evaluated in Section 5.3. A tracking controller is defined (Section 5.1) and used to demonstrate the feasibility of the path in Section 5.2. Finally, the findings are summarised and future work discussed in Section 6.

2. Quadrotor Dynamics and Kinematics

A standard quadrotor has four motors and propellers arranged in a square formation. It has an inertial translational position, velocity and acceleration $x = [x_1, x_2, x_3]^T$, $\dot{x} = [\dot{x}_1, \dot{x}_2, \dot{x}_3]^T$ and $\ddot{x} = [\ddot{x}_1, \ddot{x}_2, \ddot{x}_3]^T$, respectively. The angular velocities about the body axes are $\Omega = [\Omega_1, \Omega_2, \Omega_3]^T$, and the body frame velocities are $v = [v_1, v_2, v_3]^T$. The forces generated by the propellers are $f = [f_1, f_2, f_3, f_4]^T$, and the moments induced in the body-fixed frame are denoted as $M = [M_1, M_2, M_3]^T$. The total thrust is defined as $F = \sum_{i=1}^{4} f_i$. The mass of the quadrotor is denoted by m and the inertia matrix by $J \in R^{3\times3}$. The body-frame thrust vector is $e_3 = [0, 0, 1]^T$. In matrix form, the total thrust and moments are:

$$\begin{bmatrix} F \\ M_1 \\ M_2 \\ M_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -d & 0 & d \\ d & 0 & -d & 0 \\ -c_\tau f & c_\tau f & -c_\tau f & c_\tau f \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \tag{1}$$

where d is the distance between the centre of the quadrotor and $c_\tau f$ is a constant that relates thrust to induced yaw. In this paper, the quadrotor is treated as a rigid body with a constant gravitational acceleration g_a. To simplify the procedure of the motion planning and controller design, typical aerodynamic disturbances on the moments and forces, such as drag, ground effect and rotor dynamics, are neglected. The simulated quadrotor dynamics follow the derivations in [12–14]. The equations of motion are as described in [15]:

$$\dot{x} = Rv \tag{2}$$

$$m\ddot{x} = mg_a e_3 - FRe_3 \tag{3}$$

$$\dot{R} = R\hat{\Omega} \tag{4}$$

$$J\dot{\Omega} + \Omega \times J\Omega = M \tag{5}$$

with $R \in SO(3)$, where:

$$\mathrm{SO(3)} \triangleq \{R \in \mathbb{R}^{3\times3} : \ R^T R = I \quad \text{and} \quad \det(R) = 1\} \tag{6}$$

3. Trajectory Planning

The first subsection defines the curves and the basis behind them. The second demonstrates the process to reposition the curves from starting at the origin to a specified translational position and orientation in the inertial frame.

3.1. Sub-Riemannian Curves for Quadrotor Trajectory Planning

The kinematics, Equations (2) and (4), can be expressed equivalently on the Euclidean group of motions $SE(3)$:

$$\frac{dg}{dt} = g(\sum_{n=1}^{3} B_i v_i + \sum_{n=1}^{3} A_i \Omega_i) \tag{7}$$

where A_1, A_2, A_3, B_1, B_2 and B_3 are the basis elements of the Lie algebra of the Lie group $SE(3)$ [16–19] with $g \in SE(3)$:

$$g = \begin{pmatrix} 1 & 0 & 0 & 0 \\ x & & R & \end{pmatrix} \tag{8}$$

Body velocities for the planned motion were specified as $v_1 = v_2 = \Omega_3 = 0$ to simplify the process of deriving analytical expressions for the reference curves. It is convenient to choose a single translational body velocity, so that the translational speed in the inertial frame is equal to the magnitude of that body velocity. The translational speed will also be constant, since after applying the maximum principle, it is shown that the translational body velocity is time invariant. Additionally, having a single non-zero body velocity facilitates the joining of trajectory segments using the method described in Section 3.2. Future work will consider the inclusion of the other body velocities. With the given restrictions on the body velocities, Equation (7) can be reduced to:

$$\frac{dg}{dt} = g(v_3 B_3 + A_1\Omega_1 + A_2\Omega_2) \tag{9}$$

We choose to define a set of curves for motion planning that minimizes the cost function:

$$J = \frac{1}{2}\int_0^T v_3^2 + c(\Omega_1^2 + \Omega_2^2)dt \tag{10}$$

subject to the nonholonomic constraint Equation (9) and given boundary conditions $g(0) = g_0$ and $g(T) = g_T$, where c is a constant weight and v_i and Ω_i are measurable and bounded functions. A curve that satisfies the constraint Equation (9) and minimizes the cost function subject to the boundary conditions defines a sub-Riemannian curve on $SE(3)$ [19].

Sub-Riemannian curves are useful for trajectory planning, because they are smooth and globally defined on $SE(3)$, unlike Euler angles or quaternions that use local co-ordinates. They also naturally satisfy the constraint Equation (6). In this particular case, they can also be analytically defined, which reduces the motion planning problem to one of parameter optimisation. The full derivation for the curves is presented in Appendix 6. The curves are defined using standard functions; knowledge and understanding of their derivation are not required for their application. They are obtained by a projection of a particular sub-Riemannian curve $g_p \in SE(3)$ onto $x_p \in R^3$, where $x_p = [x_{p1}, x_{p2}, x_{p3}]^T$ and:

$$\begin{aligned} x_{p1} &= \tfrac{c_2\nu}{\gamma}((1-\cos\gamma t)\sin\beta + c_1\cos\beta\,(\sin\gamma t - \gamma t)) \\ x_{p2} &= \tfrac{c_2\nu}{\gamma}((1-\cos\gamma t)\cos\beta - c_1\sin\beta\,(\sin\gamma t - \gamma t)) \\ x_{p3} &= \tfrac{c_2^2\nu}{\gamma}\sin\gamma t + c_1^2\nu t \end{aligned} \tag{11}$$

where:

$$\begin{aligned} s &= -\sqrt{\lambda_1^2 + \lambda_2^2} \\ \lambda_3 &= \tfrac{\lambda_4\lambda_1}{\lambda_2} \\ r &= -\sqrt{\lambda_3^2 + \lambda_4^2} \\ \beta &= atan2(-\lambda_2, \lambda_1) \\ c_1 &= \tfrac{\nu}{\sqrt{r^2+\nu^2}} \\ c_2 &= \tfrac{r}{\sqrt{r^2+\nu^2}} \\ \gamma &= \tfrac{s\sqrt{r^2+\nu^2}}{rc} \end{aligned} \tag{12}$$

The quadrotor speed ν and weighting c are chosen by the user before the optimisation. A desired final position $x_d = [x_{d1}, x_{d2}, x_{d3}]^T$ is also specified by the user. A numerical optimisation using MATLAB's $fmincon$ that implements a sequential quadratic programming method with an active-set region was used to find the optimisation vector $\Xi = [\lambda_1, \lambda_2, \lambda_4, T]$ that minimizes the error in the final position:

$$C_{x_f} = \sqrt{(x_d(T) - x_p(T))^2} \tag{13}$$

where T is the final time and constrained such that $T > 0$. The translational velocity $\dot{x}_p$ can be defined analytically by taking the first derivative of Equation (11):

$$\begin{aligned} \dot{x}_{p1} &= c_2\nu(c_1 \cos\beta\,(\cos\gamma t - 1) + \sin\beta \sin\gamma t) \\ \dot{x}_{p2} &= c_2\nu(c_1 \sin\beta\,(1 - \cos\gamma t) + \cos\beta \sin\gamma t) \\ \dot{x}_{p3} &= \nu(c_1^2 + c_2^2 \cos\gamma t) \end{aligned} \tag{14}$$

Likewise, the second derivative of Equation (11) gives the translational acceleration:

$$\begin{aligned} \ddot{x}_{p1} &= c_2\nu(\gamma\ \cos(t\gamma)\sin(\beta) - c_1\gamma\cos(\beta)\sin(t\gamma)) \\ \ddot{x}_{p2} &= c_2\nu(\gamma\ \cos(t\gamma)\cos(\beta) + c_1\gamma\sin(\beta)\sin(t\gamma)) \\ \ddot{x}_{p3} &= -{c_2}^2\nu\gamma\ \sin(t\gamma) \end{aligned} \tag{15}$$

The magnitude of the acceleration is:

$$\|\mathbf{a}\| = \sqrt{\ddot{x}_{p1}^2 + \ddot{x}_{p2}^2 + \ddot{x}_{p3}^2} \tag{16}$$

It can be shown that $\|\mathbf{a}\|$ is constant by substituting Equation (15) into Equation (16) and simplifying:

$$\|\mathbf{a}\| = abs\,(c_2\nu\gamma) \tag{17}$$

The magnitude of the velocity is constant and simply ν, as determined by the user before calculating the optimisation vector.

3.2. Trajectory Repositioning and Reorientation

For the particular curves described in Section 3.1, the initial position and rotation are, respectively:

$$\begin{aligned} x_p(0) &= [0, 0, 0]^T \\ R(0) &= I \end{aligned} \tag{18}$$

where I is the the identity matrix. The curves can be placed anywhere in the inertial frame $x_i(t)$ using:

$$[1 \quad x_i(t)]^T = g_i(0).[1 \quad x_p(t)]^T \tag{19}$$

where $g_i(0)$ is the initial state of the path. It should be noted that $g_i(t)$ includes a description of the rotation (R) of the curve over time. However, this is not used as part of the trajectory tracking, because it is purely kinematic and does not account for the dynamics (such as gravitational acceleration). Similarly, for an initial and desired state in the inertial frame, the final state $g_p(T)$ for the particular curves is given by:

$$g_p(T) = g_i(0)^{-1}.g_i(T) \tag{20}$$

Since the final rotation is not used when generating the curves using the numerical optimiser, the translation position $x_p(T)$ can be retrieved from:

$$[1 \quad x_p(T)]^T = g_p(T).[1\ 0\ 0\ 0]^T \tag{21}$$

The paths generated using this procedure of joining trajectories are smooth to the second derivative; there are no discontinuities in position or velocity. However, they are not smooth in the third derivative, because the trajectories have different curvatures, so it is not possible to match acceleration.

4. Obstacle Detection

A simple method for checking the validity of a path in the obstacle space is to discretize it into nodes separated by a small time step. Each node lies on the path and can be checked for collisions. The choice of time step is a trade-off between computational time and the completeness of the check. This method can be extended to a hybrid method using a large time step and then performs a numerical minimization to find the smallest collision distance d along the path. The collision distance for an obstacle is defined as the Euclidean distance between the edge of the object and the quadrotor:

$$d(t) = \|x_{obs} - x_i(t)\| - r_{ob} \tag{22}$$

where $x_{obs} \in R^3$ is the centre of the sphere, x_i is the planned translation position of the quadrotor Equation (19) using the curves described in Section 3.1 and r_{ob} is the radius of the sphere. For practical purposes, some additional margin should be given, because the actual size of the quadrotor is not accounted for by Equation (22).

The paths are composed of trajectories patched together using the method explained in Section 3.2. To find the location on a path where an object is closest, the optimisation vector for the relevant segment must be loaded. The concatenated nature of the path is not an issue, because there are no discontinuities in position when transitioning between trajectories. Time was constrained, such that:

$$0 \leq t \leq T \tag{23}$$

and the function to minimize is:

$$d_{min} = min\,(d(t)) \tag{24}$$

where d_{min} is the minimum collision distance along a path for a specified object. If $d_{min} \geq 0$, then the quadrotor will not collide with the obstacle, providing that the tracking controller guides the vehicle along the planned trajectory.

The simple local optimiser ($fmincon$) used in this paper has the potential to get caught in local minima. A global optimiser may be more suitable (such as a genetic algorithm [20]), but the trade-off between computational expense and accuracy needs to be investigated in more detail. However, in this paper, we demonstrate the implementation of the general analytical method by using the local optimiser and by providing a suitable initial guess. The hybrid method used in this paper combines discretization with a large time step and then uses a numerical minimizer to determine the precise location and magnitude of the collision distance. If any of the points are $d(t) < 0$, then the algorithm terminates

and the path is reported as invalid. The process of the hybrid algorithm is represented pictorially in Figure 1. An evaluation of the collision detection algorithm is performed in Section 5.3 and compared to discretizing the trajectory with a small time step.

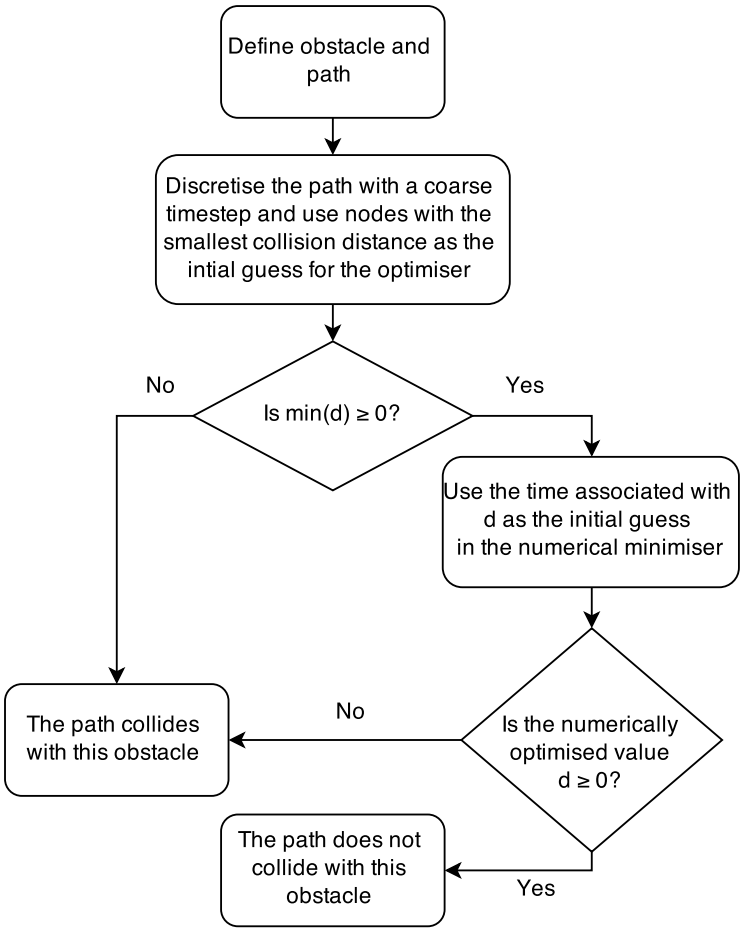


Figure 1. Obstacle detection algorithm.

5. Simulations

In the first of the following subsections, the tracking controller is described that is used to simulate the path generated in Section 5.2. Finally, Section 5.3 tests the hybrid obstacle detection algorithm and compares it to discretizing with a coarse time step.

5.1. Tracking Controller

The tracking controller was used in this paper to demonstrate the feasibility of the trajectories and was first presented in [15]. In addition to the favourable tracking performance, this controller was chosen because the trajectories are generated on the $SE(3)$ group. The tracking errors for position, velocity and angular velocity are defined as:

$$e_x = x - x_p \tag{25}$$

$$e_v = \dot{x} - \dot{x}_p \tag{26}$$

$$e_\Omega = \Omega - R^T R_d \Omega_d \tag{27}$$

where $\Omega_d = R_d^T \dot{R}_d$, and the desired rotation is defined as:

$$R_d = [\vec{b}_{1_d}; \vec{b}_{3_d} \times \vec{b}_{1_d}; \vec{b}_{3_d}] \tag{28}$$

and:

$$\vec{b}_{3_d} = R_d e_3 = -\frac{-k_x e_x - k_v e_v - m g_a e_3 + m\ddot{x}_p}{\|-k_x e_x - k_v e_v - m g_a e_3 + m\ddot{x}_p\|} \tag{29}$$

where $\vec{b}_{3_d}$ is chosen to minimize the attitude tracking error in the term FRe_3 from Equation (3) and is the tracking controller in the translational direction. The attitude error is chosen as:

$$e_R = \frac{1}{2}(R_d^T R - R^T R_d)^\vee \tag{30}$$

where the $veemap^\vee : \mathfrak{so}(3) \rightarrow R^3$. From a given desired trajectory $x_i(t)$ and heading vector $\vec{b}_{1_d}$, the control inputs can be determined:

$$\begin{aligned} F &= -(-k_x e_x - k_v e_v - m g_a e_3 + m\ddot{x}_d) \cdot Re_3 \\ M &= -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega \end{aligned} \tag{31}$$

where the tracking controller gains k_x, k_v, k_R and k_Ω must be defined and greater than zero.

5.2. Waypoints

This section gives an example of generating a path through a set of predetermined waypoints and following it using the tracking controller. The waypoints in this paper were arbitrarily chosen to demonstrate the trajectory generation algorithm. In practice, waypoints could be used to give the quadrotor-specific locations to take measurements or generated by a sampling-based planning algorithm, such as rapidly-exploring random trees.

The physical parameters used to represent a realistic quadrotor were taken from the UAV developed in [21]: $m = 4.2$ kg, $J = diag[0.0820, 0.0845, 0.1377]$ kg m^2, $d = 0.315$ m and $c_{\tau f} = 8.004 \times 10^{-3}$. For the tracking controller, the gains were those used in [15]: $k_x = 16$ m, $k_v = 5.6$ m, $k_R = 8.81$ and $k_\Omega = 2.54$. The desired heading angle was fixed as $\vec{b}_{1_d}(t) = [1, 0, 0]$. Likewise, the magnitude of the velocity was set as $\nu = 1$ and the weight $c = 200$. The initial conditions were set to:

$$\begin{aligned} x(0) &= [0, 0, 0] \\ \dot{x}(0) &= [0, 0, 0] \\ \Omega(0) &= [0, 0, 0] \\ R(0) &= I \end{aligned} \tag{32}$$

A 30% control margin [22] was used, so for a mass of 4.2 kg, the maximum thrust that can be provided by each motor is 13.4 N. This constraint was applied to the simulation.

The path consists of five waypoints specified in Table 1. The 3D path generated by the trajectory planner is shown in Figure 2 with the start of a segment and endpoint denoted by an asterisk. Figure 3 plots the inertial frame velocity components and the magnitude of the velocity, which is constant and equal to one. The total path length is 42.8 m and a total flight time of 42.8 s. The optimisation vector parameters and solution time for each segment are given in Table 2.

Table 1. Waypoints.

Waypoint	x_1	x_2	x_3
1	3	4	5
2	−2	7	3
3	−2	0	6
4	3	−4	6
5	2	0	0

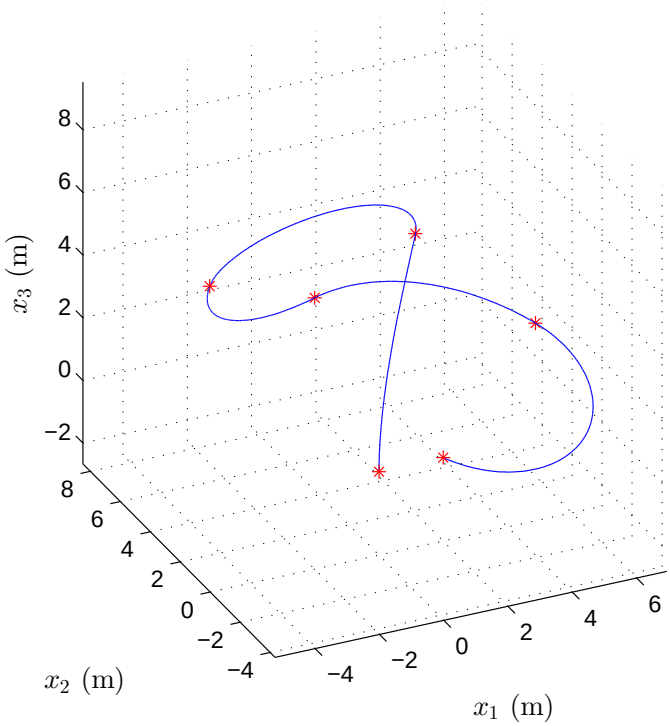


Figure 2. Concatenated trajectory passing the prescribed waypoints.

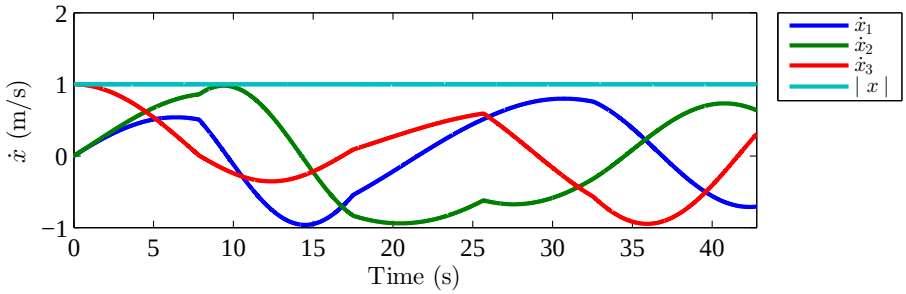
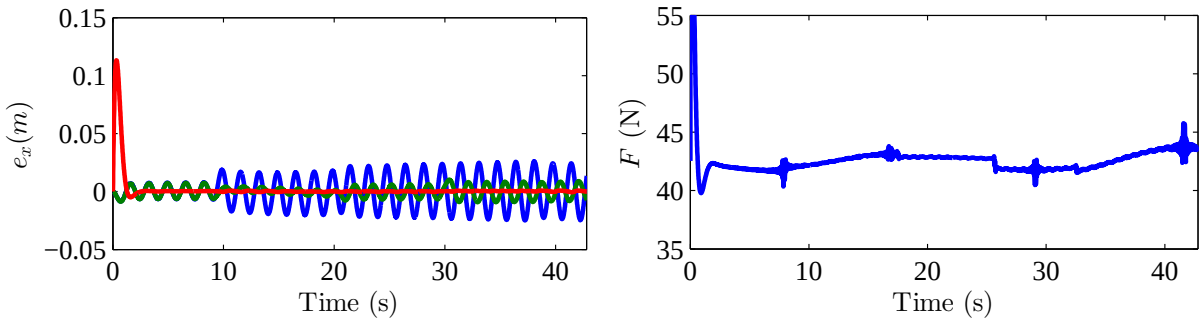


Figure 3. Quadrotor velocity during the concantenated trajectory.

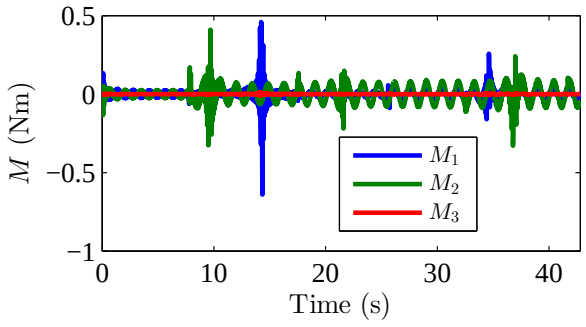
Table 2. Solution details.

Segment	λ_1	λ_2	λ_4	T	Solution Time (ms)
1	−14.136	14.281	2.773	7.858	526
2	−28.675	−15.967	−9.2926	9.656	183
3	−11.621	−10.550	3.025	8.161	108
4	−15.955	10.789	2.482	6.898	194
5	−23.425	13.467	4.527	10.224	130

The tracking controller's ability to follow the curve and demonstrate the dynamic feasibility of the path is shown in Figure **??**. The required thrust, F, is plotted against time in Figure 4b. The initial spike at 0 s is the quadrotor accelerating from rest to 1 m/s, and the additional fluctuations are caused by changing from one segment to another. This is because the paths are only smooth to the first derivative (that is, acceleration is not continuous between them). However, Figure 4a shows that the position errors oscillate around 0 m, but always remain less than 0.04 m, so the acceleration discontinuity between the segments is handled well by the tracking controller. Similarly, Figure 4c shows that the moments over time fluctuate, but none of the components have a magnitude larger than 0.5 Nm.



(a) **(b)**



(c)

5.3. Obstacle Collision Detection

The path from Section 5.2 was used to test the obstacle detection algorithm on a standard 3.4-GHz desktop computer. Five spheres were placed in the vicinity of the curve, and the points on the path closest to each obstacle were calculated using two methods.

The first method discretized the trajectory into 1,000 nodes and took 463 ms to find the closest position on the trajectory for each of the five obstacles. The second hybrid method uses a coarse discretization and then a numerical minimizer (as described in Section 4) to find the collision distance. The solution time for the hybrid method was 309 ms and is shown in Figure 5. The circles on the path mark the points where an obstacle was closest. The percentage difference between the two methods is 40%, so reduced computing time is possible, especially in cases were there are many obstacles. Additionally, since MATLAB is an interpreted language, it is slower than compiled code. If the algorithms were rewritten in a different language, such as C, and compiled, then the computation time would be reduced. However, since we are interested in comparing the methods, MATLAB was chosen for its ease of use.

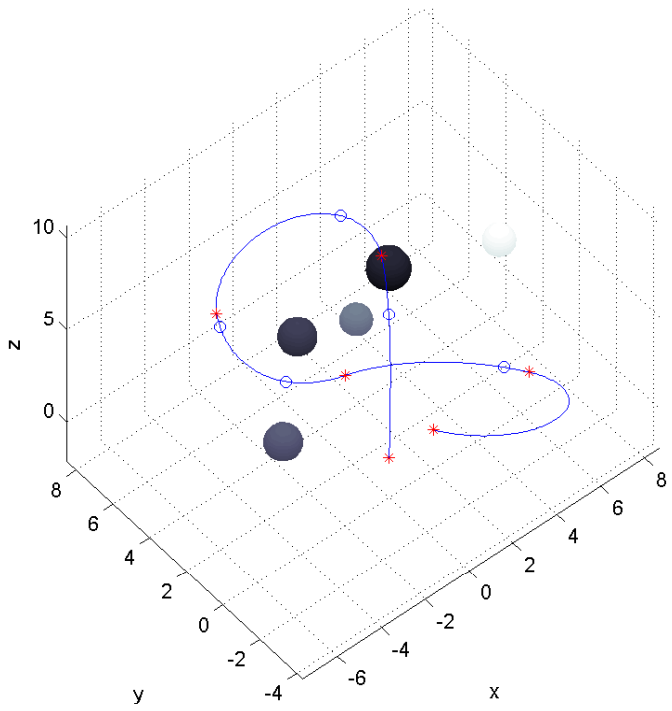


Figure 5. Obstacle detection.

6. Conclusions

This paper presents a method for concatenating separate trajectories to form smooth, analytically-defined paths for a quadrotor along a set of waypoints. Since the functions are defined using analytical expressions, only a numerical minimizer is needed to calculate the optimisation vector that controls the shape of the curve. A controller was used to ensure that the generated trajectory is trackable and dynamically feasible. Additionally, it provided information about the thrust and moments required to fly a typical quadrotor along the trajectory. An obstacle detection algorithm is presented that is a hybrid method of discretizing the path into nodes and using a numerical minimizer to find the

minimum collision distance to an obstacle. Compared to just discretizing the path, this method is faster and able to, more precisely, locate the correct collision distance.

Further work will look at using the analytically-defined curves for sample-based planning methods, allowing for a full path planning method that does not rely on the input of waypoints to navigate an area. This paper focuses on trajectory planning for quadrotor unmanned aerial systems, but the techniques used could also be applied to other types of vehicles.

Author Contributions

The paper was was written by Jonathan Jamieson and James Biggs assisted with the proof for the sub-Riemannian curves given in the appendix.

Appendix: Analytical Curves Proof

An application of the coordinate-free maximum principle [19,23] that minimizes Equation (10) subject to Equation (9) yields the Hamiltonian:

$$H(p,u,g) = v_3 p_3 + M_1\Omega_1 + M_2\Omega_2 - \frac{\rho_0}{2}(v_3^2 + c(\Omega_1^2 + \Omega_2^2)) \tag{A1}$$

where $\rho_0 = 1$ for regular extremals and $\rho_0 = 0$ for abnormal extremals. In this paper, we only consider the regular extremals; therefore, we set $\rho_0 = 1$. Noting that Equation (A1) is a concave function, then to satisfy the conditions of the maximum principle, we need $\frac{\partial H}{\partial \Omega_i} = 0$ and $\frac{\partial H}{\partial v_i} = 0$. The control functions are thus:

$$v_3^* = p_3, \quad \Omega_1^* = \frac{M_1}{c}, \quad \Omega_2^* = \frac{M_2}{c} \tag{A2}$$

Substituting Equation (A2) back into Equation (A1) gives the optimal Hamiltonian:

$$H^* = \frac{1}{2}(p_3^2 + \frac{M_1^2}{c} + \frac{M_2^2}{c}) \tag{A3}$$

The corresponding Hamiltonian vector fields describing the necessary conditions for optimality are calculated using the Poisson bracket $\{\hat{p}(\cdot), \hat{p}(\cdot)\} = -\hat{p}([\cdot,\cdot])$ where $(\cdot) \in \mathfrak{se}(3)$ [19]. Then, the Hamiltonian vector fields are given by:

$$\frac{d(\cdot)}{dt} = \{\cdot, H^*\} \tag{A4}$$

where $(\cdot) \in \mathfrak{se}(3)^*$. Explicitly:

$$\begin{array}{l} \dot{p}_1 = -\frac{M_2 p_3}{c}, \quad \dot{p}_2 = \frac{M_1 p_3}{c}, \quad \dot{p}_3 = \frac{p_1 M_2 - M_1 p_2}{c}, \\ \dot{M}_1 = p_2 p_3 - \frac{M_2 M_3}{c}, \quad \dot{M}_2 = -p_1 p_3 + \frac{M_1 M_3}{c}, \quad \dot{M}_3 = 0 \end{array} \tag{A5}$$

using the conserved quantity:

$$I_2 = p_1^2 + p_2^2 + p_3^2 \tag{A6}$$

and identifying a particular solution by assuming $\dot{p}_3 = 0$, we can reduce the conserved quantities to $s^2 = M_1^2 + M_2^2 \; \forall \; t$, where $s^2 = c(2H^* - p_3^2)$; so we can write:

$$s^2 = M_1(0)^2 + M_2(0)^2 \tag{A7}$$

and defining $r^2 = I_2 - p_3^2$ from Equation (A6), we can write $r^2 = p_1^2 + p_2^2 \; \forall \; t$; thus:

$$r^2 = p_1(0)^2 + p_2(0)^2 \tag{A8}$$

These reduced conserved quantities suggest using polar coordinates to solve the differential Equation (A5), so we try the ansatz solution:

$$\begin{aligned} M_1 &= -s\cos(\alpha t + \beta), \quad M_2 = s\sin(\alpha t + \beta) \\ p_1 &= -r\cos(\alpha t + \beta), \quad p_2 = r\sin(\alpha t + \beta) \end{aligned} \tag{A9}$$

where r and s are defined in Equations (A7) and (A8). Substituting into Equation (A5), we obtain the following two solutions for α:

$$\alpha = -\frac{sp_3(0)}{cr}, \quad \alpha = \frac{p_3(0)r}{s} - \frac{M_3(0)}{c} \tag{A10}$$

Thus, Equation (A9) is a particular solution if and only if:

$$M_3(0) = p_3(0)\left(\frac{s^2 + cr^2}{sr}\right) \tag{A11}$$

Therefore, the particular solution in terms of the initial conditions are:

$$\begin{aligned} &p_1 = -r\cos\theta, \quad p_2 = r\sin\theta, \quad p_3 = p_3(0), \\ &M_1 = -s\cos\theta, \quad M_2 = s\sin\theta, \quad M_3 = p_3(0)\left(\tfrac{s^2+cr^2}{sr}\right) \end{aligned} \tag{A12}$$

where $\theta = -\frac{sp_3(0)}{cr}t + \beta$ and s and r are defined in terms of the initial conditions in Equations (A7) and (A8) and where $\beta = \operatorname{atan2}(-M_2(0), M_1(0))$. For convenience, define a constant $K^2 = I_2$, where $I_2 = r^2 + p_3^2$ is the Casimir function Equation (A6), then as $R \in SO(3)$ is known [19] to satisfy the equation:

$$RPR^{-1} = \rho \tag{A13}$$

where $P = p_1E_1 + p_2E_2 + p_3E_3$ and E_1, E_2 and E_3 is a basis for the Lie algebra of $SO(3)$ and $\rho \in \mathfrak{so}(3)$ is a constant matrix. This fact implies that any orbit $RPR^{-1} = \rho$ is conjugate to $\rho = KE_3$, and therefore, it suffices to integrate the particular orbit:

$$R_pPR_p^{-1} = KE_3 \tag{A14}$$

Then, let ϕ_1, ϕ_2 and ϕ_3 denote the coordinates of a point in $SO(3)$ according to the formula:

$$R_p = \exp(\phi_1E_3)\exp(\phi_2E_2)\exp(\phi_3E_3) \tag{A15}$$

Then, substituting Equation (A15) into Equation (A14) yields:

$$P = K\exp(-\phi_3E_3)\exp(-\phi_2E_2)E_3\exp(\phi_2E_2)\exp(\phi_3E_3) \tag{A16}$$

which gives:

$$P = K\begin{pmatrix} 0 & -\cos\phi_2 & \sin\phi_2\sin\phi_3 \\ \cos\phi_2 & 0 & \sin\phi_2\cos\phi_3 \\ -\sin\phi_2\sin\phi_3 & -\sin\phi_2\cos\phi_3 & 0 \end{pmatrix} \tag{A17}$$

recalling that $p_3 = v_3(0)$, then:

$$\begin{aligned} r\cos\theta &= K\cos\phi_3\sin\phi_2 \\ r\sin\theta &= K\sin\phi_3\sin\phi_2 \\ p_3(0) &= K\cos\phi_2 \end{aligned} \tag{A18}$$

$$\cos\phi_2 = \frac{p_3(0)}{\sqrt{r^2 + p_3(0)^2}}, \sin\phi_2 = \frac{r}{\sqrt{r^2 + p_3(0)^2}} \tag{A19}$$

and as $\tan\theta = \tan\phi_3$, then $\phi_3 = \theta$. To calculate ϕ_1, substitute (A15) into (4), and we can obtain the following relationships:

$$\begin{aligned} \sin\phi_2\sin\phi_3\dot{\phi}_1 + \cos\phi_3\dot{\phi}_2 &= \Omega_2^* \\ -\sin\phi_2\cos\phi_3\dot{\phi}_1 + \sin\phi_3\dot{\phi}_2 &= \Omega_1^* \end{aligned} \tag{A20}$$

which can be manipulated to give:

$$\dot{\phi}_1 = \frac{\Omega_2^*\sin\phi_3 - \Omega_1^*\cos\phi_3}{\sin\phi_2} \tag{A21}$$

which can be simplified and integrated assuming $\phi_1(0) = 0$ to yield:

$$\phi_1 = (\frac{s\sqrt{r^2 + p_3(0)^2}}{rc})t \tag{A22}$$

substituting all of the values into Equation (A15) to obtain a particular optimal solution for R_p, we can then obtain x_p analytically by integration of the equation $\frac{dx_p}{dt} = R_p v$ to give Equation (11). These analytic solutions for x_p and R_p can then be used to form a particular solution $g_p(t)$ of Equation (8). For convenience, we pull this solution back to the identity by computing $g(t) = g_p(0)^{-1}g_p(t)$, so that the quadrotor starts at the origin; then, x_p Equation (11) is obtained via the following projection $[1\ x]^T = g.[1\ 0\ 0\ 0]^T$. The following notation has been used in Equation (11)–(17): $\lambda_1 = M_1(0)$, $\lambda_2 = M_2(0)$, $\lambda_3 = p_1(0)$, $\lambda_4 = p_2(0)$ and $\nu = v_3$.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. Matsumoto, T.; Kita, K.; Suzuki, R.; Oosedo, A.; Go, K.; Hoshino, Y.; Konno, A.; Uchiyama, M. A hovering control strategy for a Tail-Sitter VTOL UAV that increases stability against large Disturbance. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010), Anchorage, AK, USA, 3–8 May 2010; pp. 54–59.
2. Mueller, M.W.; Hehn, M.; D'Andrea, R. A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013), Tokyo, Japan, 3–7 November 2013; pp. 3480–3486.
3. Bouktir, Y.; Haddad, M.; Chettibi, T. Trajectory planning for a quadrotor helicopter. In Proceedings of the 2008 16th Mediterranean Conference on Control and Automation, Ajaccio Corsica, France, 25–27 June 2008; pp. 1258–1263.

4. Babel, L. Three-dimensional route planning for unmanned aerial vehicles in a risk environment. *J. Intell. Robot. Syst.* **2013**, *71*, 255–269.
5. Lizarraga, M.; Elkaim, G. Spatially deconflicted path generation for multiple UAVs in a bounded airspace. In Proceedings of the 2008 IEEE/ION Position, Location and Navigation Symposium, Monterey, CA, USA, 5–8 May 2008; pp. 1213–1218.
6. Lewis, L.R.; Ross, I.M.; Gong, Q. Pseudospectral motion planning techniques for autonomous obstacle avoidance. In Proceedings of the 2007 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007 ; pp. 5997–6002.
7. Lewis, L.P.R. Rapid Motion Planning and Autonomous Obstacle Avoidance for Unmanned Vehicles. Ph.D. Thesis, Naval Postgraduate School, Monterey, CA, USA, 2006.
8. Lin, Y.; Saripalli, S. Path planning using 3D dubins curve for unmanned aerial vehicles. In Proceedings of the 2014 IEEE International Conference on Unmanned Aircraft Systems (ICUAS 2014), Orlando, FL, USA, 27–30 May 2014; pp. 296–304.
9. Van den Berg, J.; Lin, M.; Manocha, D. Reciprocal velocity obstacles for real-time multi-agent navigation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008), Pasadena, CA, USA, 19–23 May 2008; pp. 1928–1935.
10. Shim, D.H.; Sastry, S. An evasive maneuvering algorithm for UAVs in see-and-avoid situations. In Proceedings of the 2007 IEEE American Control Conference (ACC 2007), New York, NY, USA, 9–13 July 2007; pp. 3886–3891.
11. Jamieson, J.; Biggs, J. Trajectory generation using sub-riemannian curves for quadrotor UAVs. In Proceedings of the 2015 European Control Conference (ECC 2015), Linz, Austria, 15-17 July 2015.
12. Bouabdallah, S. Design and Control of Quadrotors with Application to Autonomous Flying. Ph.D. Thesis, Swiss Federal Institute of Technology in Lausanne, Lausanne, Switzerland, 2007.
13. Martinez, V. Modelling of the Flight Dynamics of a Quadrotor Helicopter. M.Sc. Thesis, Cranfield University, Bedford, UK, 2007.
14. Amir, M.; Abbass, V. Modeling of quadrotor helicopter dynamics. In Proceedings of the 2008 International Conference on Smart Manufacturing Application (ICSMA 2008), Goyang-si, Korea, 9–11 April 2008; pp. 100–105.
15. Lee, T.; Leoky, M.; McClamroch, N. Geometric tRacking control of a quadrotor UAV on SE(3). In Proceedings of the 2010 49th IEEE Conference on Decision and Control (CDC 2010), Atlanta, GA, USA, 15–17 December 2010; pp. 5420–5425.
16. Biggs, J.; Holderbaum, W.; Jurdjevic, V. Singularities of Optimal Control Problems on Some 6-D Lie Groups. *IEEE Trans. Autom. Control* **2007**, *52*, 1027–1038.
17. Biggs, J.; Holderbaum, W. Optimal Kinematic Control of an Autonomous Underwater Vehicle. *IEEE Trans. Autom. Control* **2009**, *54*, 1623–1626.
18. Walsh, G.; Montgomery, R.; Sastry, S. Optimal path planning on matrix lie groups. In Proceedings of the 33rd IEEE Conference Decision and Control, Lake Buena Vista, FL, USA, 14–16 December 1994; Volume 2, pp. 1258–1263.
19. Jurdjevic, V. *Geometric Control Theory*; Cambridge University Press: Cambridge, UK, 1996.

20. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197.
21. Pounds, P.; Mahony, R.; Corke, P. Modelling and control of a large quadrotor robot. *Control Eng. Pract.* **2010**, *18*, 691–699.
22. Pounds, P.; Mahony, R.; Gresham, J.; Corke, P.; Roberts, J.M. Towards dynamically-favourable quad-rotor aerial robots. In Proceedings of the 2004 Australasian Conference on Robotics & Automation, Canberra, Australia, 26 April–1 May 2004.
23. Sussmann, H.J. An introduction to the coordinate-free maximum principle. In *Geometry of Feedback and Optimal Control*; Jakubczyk, B., Respondek, W., Eds.; Marcel Dekker, Inc.: New York, NY, USA, 1997; pp. 463–557.