First Results in Quaternion-Based Parameterized Continuous Motion Primitive Trajectory Generation for Unmanned Aerial Vehicles

Thomas M. Howard
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
thoward@ri.cmu.edu

ABSTRACT – This paper details extending ground-based trajectory generation approaches for mobile robots to unmanned aerial vehicles. This change in vehicle type requires a shift in the representation of the vehicle orientation from Euler Angles to Quaternions. An optimization algorithm based on Newton’s method is presented to minimize the boundary state error for a forward simulation of the parameterized controls to find the correct solution. First results in fully three-dimensional trajectory generation for unmanned aerial vehicles using continuous parameterized controls are then presented and discussed.

I. INTRODUCTION

Trajectory generation is the capacity to generate controls which meet a set of boundary states (position, orientation, etc.) given some description of its dynamics and mobility constraints. Solutions to this problem generally fall into one of three categories: simple geometric planners, discrete search via a sequence of low-order primitives, or optimization via a high-order primitive to minimize boundary state errors. This is an important problem in manipulation which applies to such fields as robotic manufacturing (planning motions to pick and place a part) and mobile robot navigation (instrument placement).

This paper deals with extending existing parameterized continuous control trajectory generation algorithms for terrestrial vehicle models to unmanned aerial vehicles. The continuous control terrestrial algorithms rely on numerical optimization of a set of controls in order to minimize the difference between the terminal boundary state goal and the result of a forward simulation of the controls.

Extending these algorithms for unmanned aerial vehicles is a difficult problem because it requires a change of the vehicle mobility model, controls, and coordinate system. Section II will review the methods [1] that will be used for the base trajectory generation algorithm. Section III will detail the changes to the algorithm required by unmanned aerial vehicles. Section IV will demonstrate an example of solving trajectory generation problems using the UAV-based trajectory generation algorithm and Section V will detail conclusions of this work.

II. REVIEW OF PARAMETERIZED CONTROL TRAJECTORY GENERATION ALGORITHMS FOR TERRESTRIAL VEHICLE MODELS

This section will review some of the recent work in continuous control terrestrial trajectory generation methods that will be used as the foundation of the UAV-extended algorithms. Generally, trajectory generation is a form of the two-point boundary value problem. A vehicle state (q) is defined to include all degrees of freedom necessary to initialize an integral of the differential equation that describes the dynamics of the system. Typically a state vector would include information describing global position (x,y,z), orientation represented in Euler angles (φ,θ,ψ), and the body-frame linear (vx,vy,vz) and angular velocities (ωx,ωy,ωz):

\[ q = [x, y, z, \phi, \theta, \psi, v_x, v_y, v_z, \omega_x, \omega_y, \omega_z, \ldots]^T \] (1)

Note that the vehicle state for terrestrial vehicles includes information which is not directly controllable. The vehicle is assumed to move in the local tangent plane of the terrain and has its roll (φ), pitch (θ), elevation (z), linear velocity about the z axis (vz), and angular velocities about the x and y axes (ωx,ωy) determined as a function of its position and motion over the terrain.

Parameterized controls are formed using only the controllable velocities (vx,vy,vz) and typically take the form of polynomials or splines where coefficients determine the shape of the control. For example, an Ackermann steered vehicle could have a set of controls composed of a linear polynomial vx and a cubic polynomial ωz (Ackermann steered vehicles cannot directly control velocity in the y-direction), which results in a control vector (u) of length 5 (including the time required to reach the terminal state):

\begin{align*}
    v_x(u,t) &= v_{x,0} + a_0 t \\
    \omega_z(u,t) &= \omega_{z,0} + a_1 t + b_1 t^2 + c_1 t^3 \\
    u &= [a_0, a_1, b_1, c_1, t]^T \end{align*} (2)

The set of differential equations that represents the mobility model of a terrestrial vehicle form by projecting the body-
frame linear and angular velocity vectors into the world frame through a rotation matrix defined using Euler angles:

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z}
\end{bmatrix} = \begin{bmatrix}
(cy \phi \theta)(v_x + (cy \phi \theta + cy \psi \phi + cy \psi \theta + cy \phi \psi) v_y) \\
(sy \phi \theta)(v_x + (sy \phi \theta + sy \phi \psi + cy \phi \psi + cy \phi \phi) v_y + (cy \phi \theta + sy \phi \phi) v_z) \\
\frac{1}{2} (s \phi \theta)\frac{dv_z}{dt} + \frac{1}{2} (s \phi \phi)\frac{dv_y}{dt} + \frac{1}{2} (c \phi \theta)\frac{dv_y}{dt} + \frac{1}{2} (c \phi \phi)\frac{dv_z}{dt} + \frac{1}{2} (v_x + (v_x + v_y + v_z) + v_x)
\end{bmatrix}
\]

Notice that the rates of change for the velocities are kept general to allow the inclusion of actuator and wheel slip models as in [1]. The numerical optimization algorithm relies on Newton’s method to minimize the boundary state error [2]:

\[
\dot{q}(u) = \frac{\partial q(u)}{\partial u} \dot{u}
\]

This method is used because the space of the solution has shown to be generally free of local minima, quite robust in practice, and only requires that the Jacobian of the system be determined (the Hessian is much more costly to compute numerically). Since the differential equation which governs the motion of the system is non-integrable (the position and orientation dynamics take the form of the Fresnel integrals), the Jacobian must be formed using local linearizations of forward solutions of the system. For systems where the Jacobian is non-square, a least-squares or least-norm approximation of the inverse could be taken or the constrained optimization approach detailed in [1].

**III. EXTENSION TO UNMANNED AERIAL VEHICLES AND QUATERNION-BASED ORIENTATION**

This section will detail the change in the vehicle model and coordinate system required to adapt terrestrial trajectory generation methods to unmanned aerial vehicles. Generally, air-based vehicles can freedom to move and rotate about any direction in Cartesian three-space. This dramatically increases the number of freedoms in the system and reduces the order of each control (compared to the terrestrial trajectory generation problem) if the Jacobian of the system is to remain square.

For this paper, we will assume a typical air-based mobile robot with some constraints on its mobility. The vehicle model which to be used is free to rotate about any axis but can only translate about the forward x-axis of the vehicle (Figure 1).

The second major change to the system is the change of the coordinate system from using Euler angles to unit quaternions. Unit quaternions are required because complicated aerial maneuvers can easily travel into the undefined regions of the Euler angle orientation definition \((\theta \pm 90^\circ)\).

\[
\begin{bmatrix}
\dot{q}_x \\
\dot{q}_y \\
\dot{q}_z \\
\dot{q}_w
\end{bmatrix} = \begin{bmatrix}
\frac{1}{2} (q_y q_w - q_z q_x) \\
\frac{1}{2} (q_z q_x - q_y q_w) \\
\frac{1}{2} (q_y q_x + q_z q_w) \\
\frac{1}{2} (q_z q_y - q_x q_w)
\end{bmatrix}
\]

where \(\varepsilon\) is a correction factor that requires that the quaternion magnitude remains unity.
IV. UNMANNED AERIAL VEHICLES TRAJECTORY GENERATION EXAMPLES

This section will demonstrate how this algorithm is applied to a specific case of trajectory generation for unmanned aerial vehicles. To test the algorithm, I looked to solve for position and orientation state constraints for unmanned aerial vehicle traveling at a constant unity velocity. Since all that we are about is the terminal position and heading, the terminal boundary state vector consists of only those elements:

\[ q_f = [x, y, z, q_0, q_1, q_2, q_3]^T \]  

For the initial state, I set the vehicle start at the origin with a unit quaternion that represents an orientation directed along the x-axis:

\[ q_i = [x = 0, y = 0, z = 0, q_0 = 1, q_1 = 0, q_2 = 0, q_3 = 0]^T \]  

Since the UAV is assumed to travel at a unity velocity, there are no free parameters from the linear velocity control. It is desirable that the length of the control vector be equal to that of the constraint vector (length = 7), we assume quadratic polynomial controls for the three angular velocity controls (+3*2) plus the time to reach the terminal state (+1):

\[ \omega_1(t) = a_0 + a_1t + a_2t^2 \]
\[ \omega_2(t) = b_0 + b_1t + b_2t^2 \]
\[ u = [a_0, b_0, a_1, b_1, a_2, b_2, t_f]^T \]  

Figure 2 shows the results for a variety of terminal position constraints (y and z vary around a circle) with constant terminal state orientation (same as the original orientation). The yellow spheres represent the goal states and the connected paths represent solved trajectories.

Figure 3 shows a close-up of one of the solved trajectory generation problems from multiple views, which show the position and orientation of the unmanned aerial vehicle as it moves through space.

V. CONCLUSIONS AND FUTURE WORK

While this method has successfully proved that it is capable to generate trajectories for unmanned aerial vehicles, the first tests also show that convergence issue exist. Several hypotheses could be made for the reason why, with the first being that the shape of the solutions is not well approximated by a linear function which results in poor convergence of the Secant method (as opposed to the terrestrial trajectory generation problem, where it proves to be very efficient). Another possible reason for failure would be poor approximation of the function due to the simplistic Euler forward method used in the numerical integration. Further study into the convergence behaviour of this algorithm and experimentation into different optimization methods may provide insight some of the drawbacks of this implementation.
Despite some of the tests failure to reach convergence, it was demonstrated that this approach can be used to determine parameterized controls which reach terminal state constraints for unmanned aerial vehicles. Further work (including a study of the convergence behaviour) can include modelling more complex vehicle dynamics and the influence of environmental forces such as wind and drag.

REFERENCES