Coordinated Control of Multiple UAVs for Time-Critical Applications
Isaac I. Kaminer, Oleg A. Yakimenko
Naval Postgraduate School, Dept. of Mechanical and Astronautical Engineering
700 Dyer Road
Monterey, CA 93943
831-656-2826
{kaminer,oayakime}@nps.edu

Antonio M. Pascoal
Instituto Superior Técnico, Dept. of Electrical Engineering and Computers
1 Avenue Rovisco Pais
Lisbon 1049-001, Portugal
(351)-21-8418051
antonio@isr.ist.utl.pt

Abstract—The paper proposes a solution to the problem of coordinated control of multiple unmanned air vehicles (UAVs) to ensure collision-free maneuvers under strict spatial and temporal constraints. The solution proposed relies on the decoupling of space and time in the problem formulation. First, a set of feasible trajectories are generated for all UAVs using a new direct method of optimal control that takes into account rules for collision avoidance. A by-product of this step yields for each vehicle a spatial path to be followed, together with a desired nominal speed profile along that path. Each vehicle is then asked to execute a pure path following maneuver in three-dimensional space by resorting to a novel 3-D algorithm that enforces temporal constraints aimed at coordinating the fleet of vehicles. Simulations illustrate the potential of the methodology developed.

TABLE OF CONTENTS
1. Introduction ................................................................. 1
2. Near-Optimal Real-Time Trajectories Generation........... 2
3. Path Following of Polynomial Trajectories ..................... 4
4. Simulations ................................................................. 7
5. Summary.......................................................................... 8
References ........................................................................... 8
Biographies.......................................................................... 9

1. Introduction

This paper addresses the problem of coordinated control of multiple unmanned air vehicles (UAVs) under tight spatial and temporal constraints. This topic of research is motivated by the need to develop strategies for coordinated ground target suppression and sequential autolanding for multiple UAVs. Both mission scenarios require a group of UAVs to execute time-critical maneuvers in close proximity of each other. For the case of ground target suppression, a formation of UAVs must break up and execute a coordinated maneuver to arrive at a predefined position over the target at a given time. Similarly, for the case of sequential autolanding, a formation must also break up and arrive at the assigned glideslope point separated by pre-specified safe-guarding time intervals. A key requirement underlying these missions is that all maneuvers be collision-free.

In recent years, there has been widespread interest in the problem of coordinated motion control of fleets of autonomous vehicles. Applications include aircraft and spacecraft formation flying [1]-[4], coordinated control of land robots [5]-[6], and control of multiple surface and underwater vehicles [7]-[10]. The work reported in the literature addresses a large class of topics that include, among others, leader/follower formation flying, control of the center of mass and radius of dispersion of swarms of vehicles, and uniform coverage of an area by a group of surveying robots. There are, however, applications with UAVs that do not fit the scenarios commonly described in the literature. Namely, the missions described in the present work that include spatial as well as temporal requirements.

To deal with the new scenarios, a methodology for coordinated control of UAVs is proposed that unfolds in two basic steps. First, a set of feasible trajectories are generated for all UAVs using a direct method of optimal control that takes explicitly into account the boundary initial and final conditions, the simplified UAV dynamics, and safety rules for collision avoidance. This is done by resorting to an extension of the work reported in [11] to multiple UAVs. A by-product of this step yields - for each vehicle - a spatial path to be followed, together with a desired nominal speed profile along that path. The second step consists of making each vehicle execute a path following maneuver along its assigned path, while enforcing temporal constraints aimed at coordinating the fleet of vehicles. This is achieved by using a new nonlinear path
following algorithm in three-dimensional space that
generalizes the one introduced in [5] for wheeled robots and
by manipulating the speed of progression along the path
about the nominal assigned speed. Clearly, the methodology
proposed relies on the decoupling of space and time in the
problem formulation. The rationale for this procedure stems
from the fact that path following controllers are easier to
design than trajectory tracking controllers and, when
properly designed, yield smooth approaching maneuvers to
the spatial curves that must be tracked. At the same time,
this strategy will naturally generate the control activity that
is required to capture the nominal paths generated during the
path planning phase, even if due to unforeseen disturbances
the vehicle deviates too much from it.

The paper is organized as follows. Section 2 describes the
methodology adopted for near-optimal real-time UAV
trajectory generation. Section 3 offers a solution to the
problem of path following in 3D. Finally, Section 4 includes
the results of simulations with the nonlinear dynamic
models of a small fleet of UAVs.

2. Near-Optimal Real-Time Trajectories
Generation

This section discusses the algorithm used for real-time
trajectory generation for multiple UAVs. First, the case of a
single UAV is addressed as in [11]. These results are then
generalized to the case of multiple UAVs.

UAV Model

Let \{I\} denote a local level coordinate system with x-axis
pointing East, y – North, and z – Up. Then the set of point-
mass equations for the UAV’s coordinates (x, y, z), speed v,
flight path angle \(\gamma\), heading \(\psi\), and mass \(m\), assuming flat
Earth, and small side-slip angle has the following well
known form [11]-[12]:

\[
\begin{align*}
\dot{x} &= v \cos \gamma \cos \psi, \\
\dot{y} &= v \cos \gamma \sin \psi, \\
\dot{z} &= v \sin \gamma, \\
\dot{\psi} &= \frac{g}{v} (n_x \cos \phi - \cos \gamma), \\
\dot{\gamma} &= \frac{g}{v} (n_y \sin \phi), \\
\dot{\phi} &= -C_s, \\
\dot{n}_x &= \frac{k_s \delta_t - \pi}{t_s}, \\
\dot{n}_y &= \frac{T(\delta_t, \pi) - D}{mg}, \\
\dot{n}_z &= \frac{L}{mg}.
\end{align*}
\]

In (1) \(n_x\) and \(n_y\) denote longitudinal and normal components of
the load factor, that depend on the current thrust \(T\), drag
\(D\), and lift \(L\). \(g\) is the acceleration due to gravity. In turn,
thrust \(T\) depends on relative thrust (throttle setting) \(\delta_t\) and
engine’s revolutions per second \(\pi\). The dynamics for \(\pi\) are
modeled by a first order differential equation. The
remaining two include the bank angle \(\phi\) and the fuel
consumption \(C_s\). Thus, \(\xi = \{x, y, z, v, \gamma, \psi, m\}^T\) is the state
vector for a UAV and \(u = \{\delta_t, n_x, \phi\}^T\) is the vector of
control inputs. The restrictions on control inputs are of the
form \(\delta_t \in [\delta_{t_{\min}}; \delta_{t_{\max}}], n_x \in [n_{x_{\min}}; n_{x_{\max}}], \phi \leq \phi_{\max}\) . The
restrictions on their derivatives take into account engine
build-up and thrust-decay times as well as the characteristics
of UAV’s control system.

Reference Functions for Local Level Coordinates

We assume that the position states \(x, y\) and \(z\) can be
represented by algebraic polynomials of degree \(n\) with the
independent parameter \(s \in [0; s_f]\), where \(s_f\) is the virtual
path length considered to be the first optimization
parameter. This makes it possible to define UAV’s position
states as follows:

\[
x_i(s) = \sum_{k=0}^n a_d (\text{max}(1, k-2))!s^k, \quad i = 1, 2, 3
\]

(where for notational convenience we set \(x_1 = x\), \(x_2 = y\) and
\(x_3 = z\)).

The degree \(n\) of the polynomial \(x_i(s)\), is determined by
whether boundary conditions to be satisfied include
constraints on position; position and velocity; position,
velocity and acceleration. The coefficients of the
polynomials in (2) are determined by satisfying the
boundary conditions. We denote the number of initial
condition constraints by \(d_i\) and final condition constraints
by \(d_f\). Then the minimal degree of each polynomial in (2)
is \(n_i = d_i + d_f + 1\). It is important to point out that the
parameterization (2) completely determines the spatial
profile of the UAV. The trajectory that can be followed with
a variety of speed profiles, since we have separated
trajectory and speed (and haven’t touched the latter yet).

Now, varying the length of the virtual path \(s_f\) changes the
look of the trajectory, therefore providing the flexibility for
deconfliction in case of multiple UAVs. Furthermore, using
\(n > n_i\) allows for more variable parameters and therefore
for more flexible trajectories (increasing however the
required CPU time for optimization). A complete discussion of
this subject can be found in [11] and [13].

Determination of the Velocity Profile

The UAV velocity along the path defined in (2) \(v(s)\) may
be determined in two ways. The first approach is to integrate
the corresponding equations of motion using given thrust vs.
time profile to yield
\[
v'(s) = g(n_x - \sin \gamma) \frac{dt}{ds} = \frac{g(n_x - \sin \gamma)}{\lambda(s)}, \quad (3)
\]

where
\[
\lambda(s) = \frac{ds}{dt} \tag{4}
\]

represents velocity along the virtual path \( s \). For instance, consider the on/off throttle control for landing approach (nominal power at the beginning and idle at the end). If we set a single throttle switching point from \( \delta_{T \text{max}} \) to \( \delta_{T \text{min}} \) to occur at the moment \( t^*_f \left( s^*_f \right) \), then the search for the near-optimal control will be made over admissible path length \( 0 \leq s^*_f \leq s_f \) (relative thrust \( \bar{T} \) then can be calculated with regard to the thrust build-up time and thrust-Decay time). The second approach is to predefine a separate reference polynomial function for \( v(s) \).

**Solution of the Inverse Dynamics Problem**

The trajectory parameters are determined numerically at \( N \) points equally spaced over the virtual path with increments
\[
\Delta s = s_f (N-1)^{-1}.
\]

This corresponds to the time intervals
\[
\Delta t_j = v^{-1} \sqrt{\sum_{i=1}^{N} (x_{i,j+1} - x_{i,j})^2}, \quad (j = 1, N-1).
\]

Using \( \Delta s \) and \( \Delta t_j \), the parameter \( \lambda \) (4) is calculated at each step. Explicit expressions for UAV position (2), with the velocity (3) calculated at the corresponding time instants, uniquely determine all the motion parameters: \( \gamma(t), \psi(t), \phi(t) \) and \( n_x(t) \) [11], [13].

It should be emphasized here that for cooperative control applications, optimization of the trajectories itself, say from the standpoint of minimum time or minimum fuel, is not the major goal. Deconfliction and feasibility of the trajectories for the flock of UAVs - that’s where the main accent shifts and that’s where the power of the suggested direct method is.

**Trajectory Optimization Algorithm**

Summarizing aforementioned steps, the trajectory generation algorithm includes the following steps.

1. Using an arbitrary value of the virtual path length \( s_f \) and a set of free polynomial coefficients \( \mathbf{R} \) (when \( n > n^* \)), compute the reference polynomial (2).

2. Using arbitrary initial guess of the throttle switching parameter \( s^*_f \), integrate equation (3) over the interval \( s \in \left[ 0; s_f \right] \) with the integration step \( \Delta s \).

3. Using inverse dynamics and expressions (2) and (3) obtain the values for UAV states and controls [11], [13]-[14]. At the end of the trajectory, compute the functional \( J \) and the penalty function \( G \) expressed as a weighted difference of the final velocity errors and constraint violations.

4. Iterate over steps 1–3 to solve a minimization problem of the following form
\[
\Xi_{opt} = \arg \min_{G(\Xi) \leq c} J(\Xi),
\]

where the parameter space \( \Xi = \left\{ s_f, \mathbf{R}, s^*_f \right\} \in \mathbb{R}^3 \), and \( c \) is the predefined tolerance.

This problem can be effectively solved with the help of any zero-order method, e.g. the Nelder-Mead downhill simplex algorithm or the Hooke-Jeeves pattern direct search algorithm. Of course, both algorithms should be modified in order to search the functional extremum only when penalty function is less than specified value (correspondent scripts were written in MATLAB). The weights of the penalty function \( G \) are chosen heuristically to ensure the specified accuracy when matching the terminal value of the UAV’s velocity and all the restrictions.

Setting the cost functional \( J = s_f \) corresponds to the time-optimal control problem. However, any other parameter, different from the time \( t_f \), or even a combination of several parameters can be used to define \( J \). This is one of the main advantages of the direct method. Two other advantages include: i) intuitively understandable analytic presentation of near-optimal solution, and ii) convergence robustness.

**Extension to Multiple UAVs**

For the case of multiple UAVs, say \( M \) UAVs, the dimension of the problem increases to \( 3M \). To guarantee collision avoidance two sets of constraints are added (corresponding errors are added to the penalty function \( G \)). The first one introduces a scheduled (with interval \( \Delta \)) arrival at the top of the glide slope:
\[
t'_f - t''_f = t'_f - t''_f = \ldots = t'_{M-1} - t''_{M} = \Delta.
\]

The second one takes care of the collision avoidance (assuming minimal separation of \( E \)):
\[
\min_{j,k=1\ldots M} \sum_{i=1}^{3} (x'_i - x''_i)^2 \geq E^2.
\]
Numerical Examples

As an example, Fig.1 illustrates flexibility of the reference polynomials to compute a coordinated sequential (time-optimum for each UAV and collision free) target suppression mission by three UAVs. Fig.2 shows that this approach can be easily extended to compute an entire landing approach trajectory (this trajectory does not require glideslope tracking). Finally, Fig.3 shows an example of trajectory generation for a scheduled arrival at the top of the glideslope for a cluster of three UAVs initially flying in compact formation.

3. Path Following of Polynomial Trajectories

The algorithm for trajectory generation introduced in Section 2 yields - for each vehicle - a spatial path to be followed, together with the corresponding nominal speed profile. To follow the paths computed, this section describes a path following algorithm that extends that in [5] to a 3-D setting and introduces a further modification aimed at meeting time-critical inter-vehicle constraints. At this level, only the simplified kinematic equations of the vehicle will be addressed by taking pitch rate and yaw rate as virtual outer-loop control inputs. The dynamics will be dealt with at a later stage by introducing an inner-loop control law.

The notation required is introduces next (Fig.4). The frame \{I\} introduced in the previous section denotes a local level coordinate system with x-axis pointing East, y – North, and z – Up. We let \{F\} be a Serret-Frenet frame attached to the path, and \{W\} the wind frame attached to the UAV. We denote by \( \omega_{r_f} \) the angular velocity of \{F\} with respect to \{I\} resolved in \{F\}.

Further let \( P \) be an arbitrary point on the path that plays the role of a “virtual” aircraft to be followed. This is in contrast with the set-up for path following originally proposed in [16] where \( P \) was simply defined as the point on the path that is closest to the vehicle. Since this point may not be uniquely defined, the strategy in [16] led to very conservative estimates for the region of attraction about the path to be followed. Endowing \( P \) with an extra degree of freedom (that will be exploited later) is the key to the algorithm presented in [5] that is extended in this paper to the 3-D case. Let \( Q \) denote the center of mass of the aircraft.

Further let \( P \) be an arbitrary point on the path that plays the role of a “virtual” aircraft to be followed. This is in contrast with the set-up for path following originally proposed in [16] where \( P \) was simply defined as the point on the path that is closest to the vehicle. Since this point may not be uniquely defined, the strategy in [16] led to very conservative estimates for the region of attraction about the path to be followed. Endowing \( P \) with an extra degree of freedom (that will be exploited later) is the key to the algorithm presented in [5] that is extended in this paper to the 3-D case. Point \( Q \) can be resolved in \{I\} as \( \mathbf{q}_I = [x, y, z] \) or in \{F\} as \( \mathbf{q}_F = [s, y, z] \), so that \( \mathbf{q}_P = \mathbf{p}_v + \mathbf{q}_F \). From the previous section, the simplified UAV kinematic equations can be written as

\[
\begin{align*}
\dot{s} &= v \cos \gamma \cos \psi \\
\dot{y} &= -v \cos \gamma \sin \psi \\
\dot{z} &= v \sin \gamma \\
\dot{\gamma} &= u_v \\
\dot{\psi} &= u_z,
\end{align*}
\]

where \( u_v \) and \( u_z \) denote the virtual control inputs pitch rate and yaw rate, respectively. Following standard nomenclature [17]-[18]

\[
\mathbf{T}(s) := \frac{d\mathbf{p}(s)}{ds}, \quad \mathbf{N}(s) := \frac{d\mathbf{T}(s)}{ds}, \quad \mathbf{B}(s) := \mathbf{T}(s) \times \mathbf{N}(s)
\]

and

\[
\frac{d\mathbf{B}(s)}{ds}
\]

denote the tangent, normal, and binormal, respectively to the path. The vectors \( \mathbf{T}, \mathbf{N}, \mathbf{B} \) are orthonormal and define the basis vectors of \{F\} as well as the rotation matrix \( \mathbf{R} = [\mathbf{T}, \mathbf{N}, \mathbf{B}] \) from \{F\} to \{I\}. It is well known that

\[
\frac{d\mathbf{R}}{ds} = [\mathbf{R}, \mathbf{k}(s)],
\]

where \( \mathbf{k}(s) := \frac{d\mathbf{T}(s)}{ds} \) is the curvature of \( P_v(s) \) and \( \varsigma(s) := \pm \frac{d\mathbf{B}(s)}{ds} \) is its torsion.
Now, from the one hand, using the notation above (see Fig.4) we may state that

\[ \mathbf{q}_\ell = \mathbf{p}_\ell(s) + \frac{d}{dt} \mathbf{R}_F, \]

so that the full derivative of this expression becomes

\[ \dot{\mathbf{q}}_\ell = \frac{d}{dt} \mathbf{R} \begin{bmatrix} \dot{\mathbf{t}} \\ \dot{\mathbf{y}} \\ \dot{\mathbf{z}} \end{bmatrix} + \frac{d}{dt} \mathbf{R} \begin{bmatrix} \mathbf{R}_F \dot{\mathbf{x}} \\ \mathbf{R}_F \dot{\mathbf{y}} \\ \mathbf{R}_F \dot{\mathbf{z}} \end{bmatrix} = \frac{d}{dt} \mathbf{R} \begin{bmatrix} \mathbf{R}_F \dot{\mathbf{x}} \\ \mathbf{R}_F \dot{\mathbf{y}} \\ \mathbf{R}_F \dot{\mathbf{z}} \end{bmatrix}. \]

From the other hand the derivative \( \dot{\mathbf{q}}_\ell \) can be expressed directly in \( \{F\} \) as \( \dot{\mathbf{q}}_\ell = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \). Equating two expressions for \( \dot{\mathbf{q}}_\ell \) we get

\[ \frac{d}{dt} \mathbf{R} \begin{bmatrix} \dot{\mathbf{t}} \\ \dot{\mathbf{y}} \\ \dot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}. \]

Additionally noting that

\[ \frac{d}{dt} \mathbf{R} = \mathbf{R} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \\ \dot{\mathbf{Z}} \end{bmatrix} = \frac{d}{dt} \mathbf{R} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \]

we finally obtain

\[ \begin{bmatrix} \dot{\mathbf{t}} \\ \dot{\mathbf{y}} \\ \dot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{t}}(1 - \kappa \gamma) + \dot{\mathbf{t}}_1 \\ \dot{\mathbf{y}}_1 + \dot{\mathbf{y}}(\kappa \mathbf{s} - \mathbf{z}) \\ \dot{\mathbf{z}}_1 + \mathbf{s} \dot{\mathbf{y}}_1 \end{bmatrix}. \]

where

\[ Q(\lambda_v) = \begin{bmatrix} 1 & \sin \phi_v \tan \theta_v & \cos \phi_v \tan \theta_v \\ 0 & \cos \phi_v & -\sin \phi_v \\ 0 & \sin \phi_v \cos \theta_v & \cos \phi_v \cos \theta_v \end{bmatrix} \]

is nonsingular for \( \theta_v \neq \pm \frac{\pi}{2} \) and \( \omega_{W} \) denotes the angular velocity of \( \{W\} \) with respect to \( \{F\} \) resolved in \( \{W\} \).

We may further express \( \omega_W \) as

\[ \omega_W = \omega_{W_{BF}} - \omega_{W_{F}}. \]

Next, since

\[ \omega_{W_{F}} = \frac{d}{dt} \mathbf{R} \omega_{W} = \frac{d}{dt} \mathbf{R} \begin{bmatrix} \nu \\ \gamma \\ \kappa \end{bmatrix}, \]

we obtain that

\[ \dot{\lambda}_v = Q(\lambda_v) \left( \omega_{W_{BF}} - \omega_{W_{F}} \right) \]

\[ = \begin{bmatrix} \dot{\phi}_v \\ \dot{\theta}_v \\ \dot{\psi}_v \end{bmatrix} = \begin{bmatrix} -\nu \sin \gamma + \gamma \cos \phi_v \tan \theta_v + \psi \cos \gamma \cos \phi_v \tan \theta_v \\ -\gamma \sin \phi_v \tan \theta_v + \psi \cos \gamma \cos \phi_v \tan \theta_v + \psi \nu \sin \gamma \\ \gamma \cos \phi_v \tan \theta_v + \psi \cos \gamma \cos \phi_v \tan \theta_v + \psi \nu \sin \gamma \end{bmatrix}. \]

Note that

\[ \frac{\dot{\theta}_v}{\dot{\psi}_v} = \begin{bmatrix} \sin \psi_v \frac{\nu}{\kappa} \\ -\sin \psi_v \frac{\nu}{\kappa} \\ \cos \phi_v \sin \psi_v \frac{\nu}{\kappa} \cos \gamma + \sin \psi_v \frac{\nu}{\kappa} \end{bmatrix} + \begin{bmatrix} \cos \phi_v \sin \psi_v \frac{\nu}{\kappa} \\ -\sin \psi_v \frac{\nu}{\kappa} \\ \cos \phi_v \sin \psi_v \frac{\nu}{\kappa} \cos \gamma + \sin \psi_v \frac{\nu}{\kappa} \end{bmatrix} \cdot \frac{\dot{\gamma}}{\dot{\psi}}, \]

where \( G \) is nonsingular for all \( \gamma \neq \pm \frac{\pi}{2} \).
Let
\[
\begin{bmatrix}
    u_i \\
    u_s
\end{bmatrix} = \begin{bmatrix}
    \dot{\gamma} \\
    \dot{\psi}
\end{bmatrix} = G^{-1} \begin{bmatrix}
    u_\theta \\
    u_\varphi
\end{bmatrix} - \mathbf{D},
\] (8)
where \( \begin{bmatrix}
    u_\theta \\
    u_\varphi
\end{bmatrix} \) is an auxiliary vector to be determined later.

Then, by combining equations (5) and (4) we obtain the equations for the (path following) error dynamics:
\[
\begin{bmatrix}
    \dot{s} = -\xi \hat{y} + v \cos \theta \cos \varphi \\
    \dot{y} = -\xi \hat{y} - v \cos \theta \sin \varphi
\end{bmatrix}
G_e = \begin{bmatrix}
    \dot{s} \\
    \dot{y}
\end{bmatrix} = \begin{bmatrix}
    \xi \hat{y} - v \sin \theta \\
    \frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi)
\end{bmatrix}
\] (9)

Notice how the rate of progression \( ds/dt \) of point \( P \) along the path becomes an extra variable that can be manipulated at will. A globally asymptotically stable (GAS) control law is now derived for \( G_e \) to drive all error variables to 0 using \( u_\theta \) and \( u_\varphi \) as control inputs. To this effect, consider the candidate Lyapunov function
\[
V = \frac{1}{2} (s_i^2 + y_i^2 + z_i^2) + \frac{1}{2c_1} (\theta - \delta_\theta)^2 + \frac{1}{2c_2} (\varphi - \delta_\varphi)^2,
\] (10)

where
\[
\delta_\theta = \sin^{-1} \left( \begin{split}
\frac{1}{\epsilon} \\
\frac{\sin \theta}{z_i} + \epsilon
\end{split} \right)
\] and
\[
\delta_\varphi = \sin^{-1} \left( \begin{split}
\frac{1}{\epsilon} \\
\frac{\sin \varphi}{y_i} + \epsilon
\end{split} \right)
\] (11)
for some \( \theta > 0, \varphi > 0 \) and \( \epsilon > 0 \). Inspired by the work of Samson, the above equations capture “desired” approach angles to the path. Computing the time-derivative of \( V \) yields

\[
\dot{V} = s_i \dot{s}_i + y_i \dot{y}_i + z_i \dot{z}_i + \frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi)
\]

\[
\dot{s}_i = (s_i - \xi \hat{y} + v \cos \theta \cos \varphi) + y_i (-\xi \hat{y} - v \cos \theta \sin \varphi) + z_i (-\xi \hat{y} - v \sin \theta) +
\]
\[
\frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi)
\]
\[
\dot{y}_i = (-\xi \hat{y} + v \cos \theta \cos \varphi) + y_i v \cos \theta \sin \varphi - z_i v \sin \theta +
\]
\[
\frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi) + y_i v \cos \theta \sin \varphi - y_i v \cos \theta \sin \varphi - z_i v \sin \theta +
\]
\[
\frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi)
\]
\[
\dot{z}_i = -\xi \hat{y} - v \sin \theta + y_i v \cos \theta \sin \varphi - y_i v \cos \theta \sin \varphi - z_i v \sin \theta +
\]
\[
\frac{\theta - \delta_\theta}{c_1} (u_\theta - \delta_\theta) + \frac{\varphi - \delta_\varphi}{c_2} (u_\varphi - \delta_\varphi)
\]

Let
\[
\dot{s} = K_1 s_i + v \cos \theta \cos \varphi
\]
\[
u_\theta = -K_2 (\theta - \delta_\theta) + c_1 z_i \sin \theta \sin \delta_\theta + c_2 \sin \delta_\theta +
\]
\[
u_\varphi = -K_3 (\varphi - \delta_\varphi) + c_1 y_i \cos \theta \sin \delta_\varphi +
\]

where \( K_1 > 0, K_2 > 0, K_3 > 0 \) (the latter two expressions can be used in (8)). Then
\[
\dot{V} = -K_1 s_i^2 - K_2 \frac{(\theta - \delta_\theta)^2}{c_1} - K_3 \frac{(\varphi - \delta_\varphi)^2}{c_2} -
\]
\[
k_2 v_i^2 + y_i v \cos \theta \sin \delta_\varphi - z_i v \sin \delta_\theta =
\]
\[
-k_2 s_i^2 - K_2 \frac{(\theta - \delta_\theta)^2}{c_1} - K_3 \frac{(\varphi - \delta_\varphi)^2}{c_2} -
\]
\[
\theta \left| \frac{y_i^2}{|y_i| + \epsilon} \right| v \cos \theta \left| \frac{\theta - \delta_\theta}{|z_i| + \epsilon} \right| \left| v \right| < 0
\] (13)

thus proving global attractivity to the path as long as \( v(t) \) does not tend to 0 as \( t \) grows unbounded. The control law given by (8) and (12) guarantees “time-independent” tracking of a single trajectory because it is assumed that a desired speed profile for \( v(t) \) is set in advance. We now extend this circle of ideas to deal with the case of time critical missions for multiple UAVs. In particular, we are interested in having each UAV reach the end of its trajectory at a pre-specified time of arrival. To account for time-of-arrival error we introduce new variables: \( s_{fi} \) - desired arc length of the trajectory of the \( i^{th} \) UAV, \( t_{oai} \) - desired time of arrival of the \( i^{th} \) UAV, and \( v_{di} \) - desired velocity profile along \( i^{th} \) trajectory obtained using direct methods. Clearly,
\[
s_{fi} = s_{fi}(t_i) + \int_{t_i}^{t_f} v_i \, dt,
\]
where \( t_i \) represents current time. Notice also that the actual path length \( s_{a,i} \) is given by
\[
s_{a,i} = s(t_i) + \int_{t_i}^{t_f} v \, dt.
\]
As a consequence, the total path error becomes 
\[ \Delta s_i = s_{f,i} - s_{a,i} \] and its derivative is \[ \dot{\Delta s}_i = v_{d,i} - v_i \]. After enlarging the error variables to include \( \Delta s_i \), the complete generalized error dynamics for the \( i \)th vehicle can be written as

\[
\begin{align*}
G_{e,i} = & \begin{cases}
\dot{s}_{i,j} = -\dot{s}_i -(1-\kappa_j s_{i,j}) + v_i \cos \theta_{e,j} \cos \psi_{e,j} \\
\dot{y}_{i,j} = -\dot{y}_i -(\kappa_j s_{i,j} - \zeta_j z_{i,j}) + v_i \cos \theta_{e,j} \sin \psi_{e,j} \\
\dot{z}_{i,j} = -\zeta_j v_{i,j} - v_i \sin \theta_{e,j} \\
\dot{\theta}_{e,i} = u_{\theta,i} \\
\dot{\psi}_{e,i} = u_{\psi,i} \\
\dot{\Delta s}_i = v_{d,i} - v_i.
\end{cases}
\end{align*}
\]

Let

\[
V_i = \frac{1}{2} (s_{i,j}^2 + y_{i,j}^2 + z_{i,j}^2) + \frac{1}{2c_1} (\theta_{e,i} - \delta_{\theta,i})^2 + \frac{1}{2c_2} (\psi_{e,i} - \delta_{\psi,i})^2 + \frac{1}{2} \Delta s_i^2
\]

Then the control law

\[
\begin{align*}
\dot{s}_i &= K_{s,s_i} + v_i \cos \theta_{e,i} \cos \psi_{e,i} \\
\dot{y}_i &= -K_{s,s_i} (\theta_{e,i} - \delta_{\theta,i}) + c_1 z_{i,j} v_i \frac{\sin \theta_{e,i} - \sin \delta_{\theta,i}}{\theta_{e,i} - \delta_{\theta,i}} + \delta_{\theta,i} \\
u_{\theta,i} &= -K_{\theta,s_i} (\psi_{e,i} - \delta_{\psi,i}) \\
u_{\psi,i} &= -c_2 z_{i,j} v_i \cos \theta_{e,i} \frac{\sin \psi_{e,i} - \sin \delta_{\psi,i}}{\psi_{e,i} - \delta_{\psi,i}} + \delta_{\psi,i} \\
v_i &= K_{v,s_i} \Delta s_i + v_{d,i}
\end{align*}
\]

stabilizes the origin of system \( G_{e,i} \). This can be easily shown by direct substitution of (15) into the expression for \( V_i \). Expression (8) can be used to compute the corresponding yaw rate and pitch rate commands for each UAV.

4. Simulations

The nonlinear path following algorithm derived in Section 2 relied on a kinematic model of the vehicle under consideration. The final control law manipulates directly pitch and yaw rate, which should be viewed as virtual control inputs in the outer loop of inner-outer control architecture. An inner loop must be designed at a later stage to actually generate the required pitch and yaw rate commands. This problem will not be addressed here. Instead, we assume that an inner loop autopilot is available that yields adequate performance. The control law (16) was implemented on the nonlinear 6DOF model of a Telemaster UAV used at NPS (see Fig.5).

![Figure 5 – Modified Telemaster UAV (wing span 2.5m, take-off weight 8kg)](image)

Control commands generated by (8) and (16) were used to drive the inner loop autopilot currently implemented on the Piccolo autopilot onboard Telemaster as shown in Fig.6.

![Figure 6 - Inner/outer control structure](image)

Figure 7 includes the results of a nonlinear simulation of a sequential autoland by three Telemasters. Each UAV is required to arrive at the top of glideslope 60 sec apart with the leader arriving 280 sec from the autoland initiation. Table 1 contains time of arrival results for each UAV.

<table>
<thead>
<tr>
<th>UAV</th>
<th>Times of arrival at the top of glideslope</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>278.1s</td>
</tr>
<tr>
<td>2</td>
<td>338.0s</td>
</tr>
<tr>
<td>3</td>
<td>400.8s</td>
</tr>
</tbody>
</table>

![Figure 7](image)

Figure 8 shows the path following errors exhibited by each UAV during the maneuver. Clearly, errors in \( y \) and \( z \) channels are very small illustrating good tracking performance. Notice that values for \( s_i \) vary significantly – however - by design this has no impact on performance.

![Figure 8](image)

Figures 9 includes the pitch and roll response by each UAV and shows that the polynomial trajectories being tracked required feasible roll and pitch angles.
5. Summary

The paper presents the theory and practical implementation of on-line generation of near-optimal collision-free trajectories for UAV formation and their following automatic tracking. The results of simulations and preliminary flight tests demonstrate a great potential of the developed algorithms.

References


**Biographies**

**Isaac Kaminer** obtained the M.S.E. degree from the University of Minnesota, Minneapolis, in 1985. He received the Ph.D. degree from the University of Michigan, Ann Arbor, in 1992. He worked for the Boeing Company between his M.S.E. degree and Ph.D. degree, first on the 757/767 program and then in the guidance and control research group. He is currently an Associate Professor at the Department of Aeronautics and Astronautics at the Naval Postgraduate School, Monterey, CA, (NPS) where he has been a faculty member since August of 1992. His research interests include Integrated Plant-Controller Optimization and Integrated Guidance, Navigation and Control with applications to UAVs and Aerodynamic Decelerating Systems.

**Oleg Yakimenko** received his M.S.E. degree from the Moscow Institute of Physics and Technology, Moscow, Russia, in 1986. In 1988 he received another M.S. degree in Aeronautical Engineering and Operations Research from the Air Force Engineering Academy, Moscow, Russia (AFEIA). In the same academy he received the degree of the Candidate of Technical Sciences (Ph.D.) (1991) and Doctor of Technical Sciences (1996). He had progressed though all professorial ranks at the AFEA and since late 1998 he has been a visiting professor at the NPS. His research interests include Atmospheric Flight Mechanics, Optimal Control, Integrated Guidance, Navigation and Control with applications to UAVs and Aerodynamic Decelerating Systems.

**Antonio Pascoal** received his Ph.D. in Control Science from the University of Minnesota, Minneapolis, MN, USA in 1987. In 1987-1988 he was a Research Scientist with Integrated Systems Incorporated, Santa Clara, California, USA. He has held visiting positions with the Dept. Electrical Eng., Univ. Michigan, USA, the Dept. Aeronautics and Astronautics, NPS, and the National Institute of Oceanography, Goa, India. He was the coordinator of two EU funded projects that led to the development of the first civilian autonomous underwater
vehicle (AUV) named MARIUS. He has been active in the
design, development, and operation of autonomous
underwater and surface vehicles for scientific applications,
in the scope of several projects funded by the Portuguese
Government, the NSF/NPS (USA), and the EU. He is an
Associate Professor of the Instituto Superior Técnico (IST),
Lisbon, Portugal and Coordinator of the Dynamical Systems
and Ocean Robotics Lab of Institute for Systems and
Robotics of IST. His areas of expertise include Dynamical
Systems Theory, Navigation, Guidance, and Control of
Autonomous Vehicles, and Coordinated Motion Control of
Marine Robots.