Spatially Deconflicted Path Generation for Multiple UAVs in a Bounded Airspace

M.I. Lizarraga and G. H. Elkaim

Abstract— **This paper presents a preliminary framework for generating spatially deconflicted paths for multiple UAVs using Bézier curves. The critical issue addressed is that of guaranteeing that all the paths lie inside a pre-defined airspace volume. Its is shown that Bézier curves reperesent a natural tool for meeting this requirement. The paper reviews the essential properties of the Bézier curves that are used to guarantee spatial deconfliction between the UAV paths as well as airspace volume contsraints. The generated curves are not only non-overlapping but separated by a minimum distance chosen prior to flight. It is then shown that the path generation problem can be formulated as a constrained optimization problem over a finite optimization set and solved using standard MATLAB optimization tools. Simulation results are presented along with its discussion. The paper includes an analysis of numerical solutions obtained as well as discussion of future work.**

I. Introduction

Current civilian applications for UAV's include border surveillance, whale and other marine mammal tracking, forest fire detection and monitoring, power-line verification, and search and rescue missions in disaster areas [8].

As these flying vehicles start to be more widely adopted, and missions become more demanding, the use of multiple vehicles is a natural evolution from single-vehicle systems.

When flying multiple UAVs in a given airspace, one has two options regarding their arrangement: either fly them in formation, or have them fly independently within a constrained section of the available airspace not to interfere with each other. Deciding which of these two options works best is solely dependent on the actual state of the mission. In any given task, one might switch from independent to formation flying and vice versa. For instance, multiple UAVs monitoring sectors of a forest fire independently might be required to converge to an area to search in formation for a fallen firefighter.

Making such a transition presents several challenges. First there is no *a-priori* information on the state of each vehicle when the transition is required. Second, the vehicles must move into (or out of) formation in such a way that their paths are spatially deconflicted. And third, if moving into formation, they must arrive at a given location simultaneously.

The work presented in this paper addresses precisely those transitions. Bézier curves are used to generate a set of feasible paths and selecting an optimal one based on airspace constraints. By using Bézier curves, the paths are guaranteed to stay in a bounded airspace as long as the curve's control points remain inside of that airspace.

Mariano Lizarraga is a Ph.D. student in the Computer Engineering Department at the the University of California, Santa Cruz, 1156 High Street SOE3, Santa Cruz, CA 95064 malife@soe.ucsc.edu

Gabriel Hugh Elkaim is an Assistant Professor in the Computer Engineering Department at the University of California, Santa Cruz, 1156 High Street SOE3, Santa Cruz, CA 95064 elkaim@soe.ucsc.edu

This paper builds on the results obtained earlier. In particular, in [5] Yakimenko, et.al. extended the work presented in [6] for autonomous shipboard landing of UAVs and applied it to schedule sequential recovery of UAVs using direct method[12] to select a suboptimal solution. This idea was further extended in [4] using fifth order polynomials for near real-time generation of sub-optimal solutions. Using Bézier curves to represent flight paths has several advantages:

- The airspace boundaries constraints are satisfied by construction.
- The implementation is recursive but numerically efficient.
- The minimum number of control points is determined by the number of boundary conditions of the problem.

Bézier curves have previously been used for path planning but in different context. For example in [9] the authors used 3rd order Bézier curves to generate reference path points (for automated guided vehicles), and then for creating trajectories along these path points using splines. In [7] Bézier curves are used for path planning of planetary exploration rovers. Although this is inherently a 3D problem (due to planet's topography), planar Bézier curves (2D) are projected onto the digital elevation map to simplify the optimization problem and guarantee path-wheel contact. This treatment [7] addresses obstacle avoidance, but for obvious reasons these obstacles are assumed to be static.

The novelty of the work presented in this paper is in utilizing Bézier curves to formulate a constrained optimization problem that results in spatially deconflicted flight paths for multiple UAVs that lie inside a pre-specified airspace. We obtain this by using a straightforward algebraic framework that takes advantage of some of the properties of Bézier curves discussed.

The rest of the paper is organized as follows: Section II presents a brief review of Bézier curves and de Casteljau's algorithm. Section III formally sets up the problem and presents the framework for the independent-to-formation flight transition. Section IV describes the simulation setup and finally results of extensive MATLAB simulations are presented in Section V.

II. Bézier Curves

Bézier curves are named after the French engineer Pierre Bézier, who developed them in the late 50's to facilitate the design of automobile bodies. Today Bézier curves are widely used in computer graphics and computer aided design [10] .

Bézier curves (figure 1) are completely described by a set of coordinates called the *control points*. The polygon that is formed by joining sequential control points is known as the *control polygon*. The *n-th* degree Bézier curve is described by:

$$P(\tau) = \sum_{i=0}^{n} B_i^n(\tau) P_i, \tag{1}$$

where τ is a dimensionless parameter, $B_i^n(\tau)$ are the blending functions and are generally described by the Bernstein polynomials,

$$B_i^n(\tau) = \binom{n}{i} \left(\frac{\tau_f - \tau}{\tau_f - \tau_0}\right)^{n-i} \left(\frac{\tau - \tau_0}{\tau_f - \tau_0}\right)^{i}, i = 0, 1 \ldots n, \tag{2}$$

and P_i are the control points.

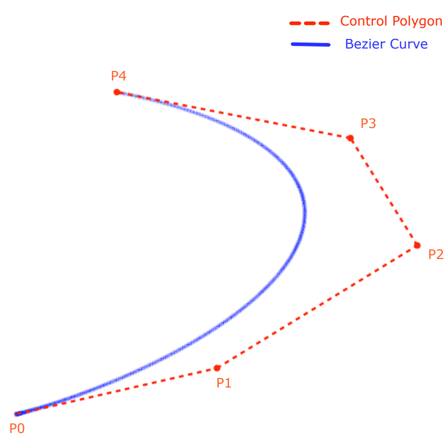

Fig. 1. Fourth Order Bézier Curve

These curves have the following characteristics:

- A curve of degree n has $n + 1$ control points.
- It starts exactly in the first control point P_0 and finishes exactly in the last control point P_n.
- It is tangent to the control polygon at the first and last control points.
- The shape is solely determined by the location of the control points.
- The curve lies within a convex hull determined by the control points.

A. Bézier Curve Derivative: The Hodograph

The derivative of a Bézier curve is generally known as the *hodograph* of the curve and it is solely determined by its control points [10]. The hodograph of an *n-th* degree Bézier curve is just another Bézier curve of degree $n - 1$ whose control points are given by:

$$D_i = \frac{n}{\tau_f - \tau_0}(P_{i+1} - P_i), \tag{3}$$

for $i = 0, 1, \ldots, n-1$. Higher order derivatives can be simply obtained by getting the hodograph of the hodograph.

B. The de Casteljau Algorithm

Named after Paul de Casteljau, a French mathematician who devised this numerical construct algorithm in 1959, the de Casteljau algorithm formally describes a recursive relation to divide a Bézier curve $P_{\tau_0 \to \tau_f}(\tau)$ in two segments $P_{\tau_0 \to \tau_1}(\tau)$ and $P_{\tau_1 \to \tau_f}(\tau)$ [1]. Apart from dividing the curve in two segments, De Casteljau's algorithm has two more practical applications. First, it can be used to sequentially divide a curve until the control polygon converges with it, which is used when computing separation between curves. Second, De Casteljau's algorithm is used to compute the coordinates of any point along the curve, given any parameter value τ_1, since $P(\tau_1) = P_{0,n}$.

Let $P_{\tau_0 \to \tau_f}(\tau)$ be defined by the control points $\{P_{0,0}, P_{1,0}, P_{2,0} \ldots, P_{n,0}\}$. The resulting segments obtained after applying de Casteljau's algorithm will be defined by the following sets of control points (figure 2):

$$\begin{aligned} P_{\tau_0 \to \tau_1}(\tau) &= \{P_{0,0}, P_{0,1}, P_{0,2} \ldots, P_{0,n}\} \\ P_{\tau_1 \to \tau_f}(\tau) &= \{P_{0,n}, P_{1,n-1}, P_{2,n-2} \ldots, P_{n,0}\} \end{aligned} \tag{4}$$

where the $P_{i,j}s$ are given by:

$$P_{i,j} = (1 - \lambda) P_{i,j-1} + \lambda P_{i+1,j-1}, \tag{5}$$

for $j = 1, 2, \ldots, n$, $i = 0, 1, \ldots, n - j$ and

$$\lambda = \frac{\tau_1 - \tau_0}{\tau_f - \tau_0}. \tag{6}$$

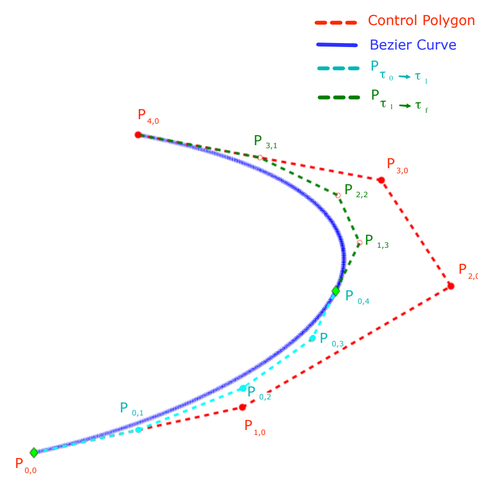

Fig. 2. Bézier Curve Divided Using de Casteljau's Algorithm

III. The Spatially Deconflicted Paths Problem

This section describes the general framework for generating spatially deconflicted paths for multiple UAVs. First it introduces the notation and setup for one vehicle (subsection III-A), then extends it to multiple vehicles in subsection III-B and finally presents the complete problem formulation and objective function to use in path deconfliction algorithm. It is important to note that in the following subsections, the problem is described in terms of 4$^{\text{th}}$ order Bézier curves and the drawings are presented in 2D only for the sake of clarity, However, nothing presented herein limits either the order of the Bézier curve, or its applicability to non-planar trajectories, as it will be demonstrated in section IV.

A. One-Vehicle Bézier Path

Let P_0, P_4, v_0, v_f be the initial and final position and velocities of the vehicle. Let $P_c(\tau)$ be the Bézier path that takes the vehicle from P_0 to P_4.

Furthermore, let the control point P_1 lie along a line in the same direction than the initial velocity vector $\mathbf{v_0}$ and the control point P_3 lies along a line with the same direction as the finial velocity vector $\mathbf{v_f}$ (figure 3) and τ be a dimensionless parameter of the curve.

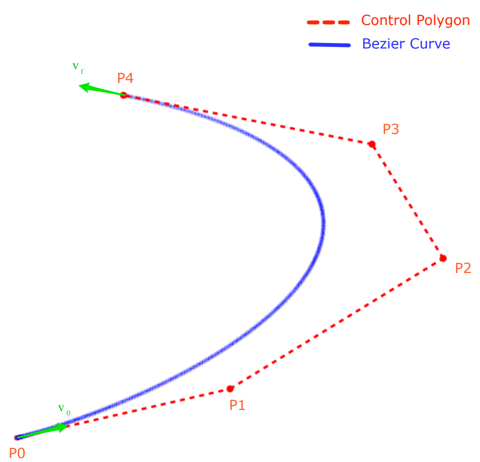

Fig. 3. One vehicle path description based on initial and final position and velocities using Bézier curves

From the above description it is clear that if one needs to modify the shape of $P_c(\tau)$ (without changing initial nor final conditions) one has $(n-3) \times 3$ degrees of freedom: $P_2, \ldots, P_{n-2}$ to move around and reshape the path. It is also clear, that even though this paper does not deal with the inverse dynamics of the problem, the curve reshaping can not be made arbitrarily since the vehicle moving along $P_c(\tau)$ has flight dynamics limitations (maximum and minimum speed, acceleration, turn rate and such) that it can achieve. To guarantee that the generated paths are feasible for the UAV dynamics, a restriction on the first and second order derivatives of the path are imposed to limit the rate of change of the curve.

$$\begin{aligned} f_{min} \leq \|P_c'(\tau)\| \leq f_{max} \\ \|P_c''(\tau)\| \leq s_{max}. \end{aligned} \tag{7}$$

Finally, Let R be the radius of a sphere bounding the permissible airspace. Then, assuming that P_0, P_1 and $P_n, P_n - 1$ are located inside the airspace, one must place the rest of the control points inside the convex hull given by the sphere :

$$\|P_i - P_{ct}\| \leq R \qquad \forall i = 2 \ldots n-2, \tag{8}$$

where P_{ct} is the location of the center of such sphere.

B. Multiple-Vehicles Bézier Path Deconfliction

The framework presented for one vehicle in the previous section can be extended to multiple vehicles with two additional constraints: the generated paths must have a minimum separation E_{min} among them; and the vehicles must use the shortest possible path from P_0 to P_4. For the minimal path, one can use a cost function that penalizes excessive lengths, as proposed in [3]:

$$J = \int_{\tau_0}^{\tau_f} kF(\tau) d\tau. \tag{9}$$

For the minimal separation, since there is no closed form of determining the closest point of approach of a Bézier curve to a given point, then a numerical solution can be used. For this, the Bézier curve is iteratively subdivided using de Casteljau's algorithm until the distance $d = \|P_0 - P_4\|$ for each segment, is smaller than the minimum separation E_{min}. This has two effects, as shown in figure 4: First, it makes the control polygon converge to the actual curve after a small number of iterations (typically less than 50); and second, it provides a set of sample points (not equally spaced) that can be used as key points to perform exhaustive search against those of the other paths to find the closest distance $D_{i,j}$, between paths i and j correspondingly.

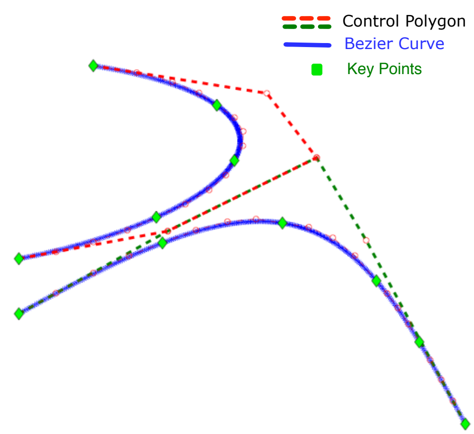

Fig. 4. Curve Poligonization using de Casteljau's Algorithm

C. Complete Problem Formulation

Let Ξ_i be the vector of optimization parameters given by: $P_2, \ldots, P_{n-2}$ for $i = 1, \ldots, m$ where m represents the number of UAVs and n the Bézier curve's order. Then the spatially deconflicted paths can be obtained by solving the following non-linear, constrained minimization problem:

$$F = \begin{cases} \displaystyle\min_{\Xi_{i=1,\ldots,m}} J = \sum_{i=1}^{m} J_i & \text{subject to (7) and (8)} \\ D_{i,j} \geq E_{min} & \forall i \neq j, i,j = 1, \ldots, m. \end{cases} \tag{10}$$

IV. Simulation Setup

Several Matlab scripts were written to set up a simulation scenario that takes three UAVs (labeled UAV_1, UAV_2 and UAV_3) from arbitrary starting locations and solves (10). This task brings UAVs into a randomly selected final position and heading 150 m apart in formation using 5^{th} order Bézier paths; the order of the curves was incremented to 5 in order to allow the additional degree of freedom to solve spatial separation (minimal distance) constraint. Table I shows details of the initialization that were passed into Matlab's Optimization Toolbox for non-linear constrained minimization by using `fmincon` [11] function.

TABLE I

Simulation Initialization and Constraints Values

	Initial Conditions
Position	Random within a 5 Km radius.
Orientation	Random
	Final Conditions
Position	Random for UAV_1; 150 m spaced in x direction for UAV_2 and UAV_3.
Orientation	Random for UAV_1; UAV_2 and UAV_3 identical to UAV_1.
	Constraints
f_{min}	10
f_{max}	40
s_{max}	0.8
E_{min}	60 m
	Optimization Vector Ξ Initialization
τ	400 s
P_2, P_3	As described in section IV-A

A. Control Points Initialization

Four of the $n+1$ control points are explicitly determined by the initial and final conditions of the problem (see Section III-A). But care must be taken in choosing the initial location of the rest of the control points, for this case P_2 and P_3, before passing Ξ to the optimization function. Since it is known that the path will travel in some way from P_1 to P_4 then it results as an obvious choice to try to place them in the vicinity of that "must-follow" path. For the initial location of P_2 and $P-3$, two approaches were used. The first consisted in placing P_2 and P_3 equally spaced along the line P_1P_4. This approach worked well but sacrificed some of the flexibility of using a higher order Bézier path, by having effectively four of the six control points in a line at the start of the problem. The second approach, which was used to produce all the results presented in the following section, was to place the control points 45° off the line P_1P_4.

V. Results

The results presented in this section correspond to 252 independent simulation runs with different initial and final conditions as described in section IV. Of these runs, 12.4% failed to converge and are therfore discarded from the data presented in figure 5.

As expected, the number of iterations and cost function evaluations performed by the constrained minimization function is heavily dependent on the initial conditions. But for those that converged, it did so by using between 150 and 300 iterations and four to ten function evaluations (figure 5).

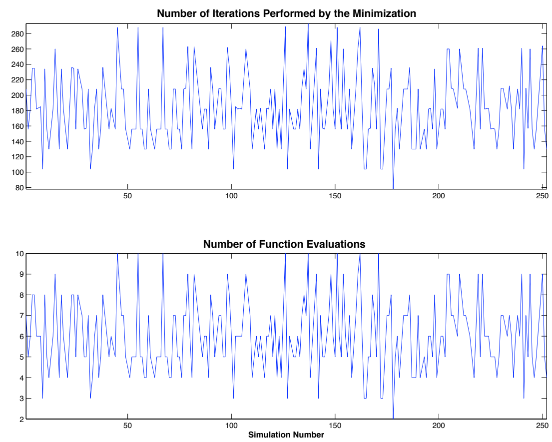

Fig. 5. Number of Iterations and Function Evaluations for Multiple Simulations

When comparing the results from [4] with those obtained here, although the same problem is addressed in both and final results are very similar (figure 6), the Bézier method has the advantage of guaranteeing the boundedness of the created flight paths by simply limiting the location of the control points.

For each run the proposed flight paths where plotted. Figure 7 presents the generated spatially deconflicted Bézier paths of a typical simulation run in Matlab. Another question that was raised during the initial testing was how much is the algorithm really modifying the flight path form the starting conditions. Figure 8 shows a set of spatially deconflicted Bézier flight path and shows for each flight path the initial and final location of the control points and consequently the change in the shape of the path.

VI. Conclusions and Future Work

This paper presented a preliminary framework for generation of spatially deconflicted paths for multiple UAVs using Bézier curves. It has demonstrated that a key advantage in using Bézier curves lies in guaranteeing that the resulting flight paths lie inside a bounded airspace by construction. Results also show good convergence properties and an acceptable computational load. This leads us to believe that performance will improve when the algorithm is translated form Matlab's interpreted language to C. However, further study is required to determine for which initial conditions the algorithm fails to converge.

The next step will be to incorporate inverse dynamics into the simulation and then to migrate code to C for hardware in the loop simulation. Furthermore, work is already underway at UCSC's Autonomous Systems Lab to incorporate parts

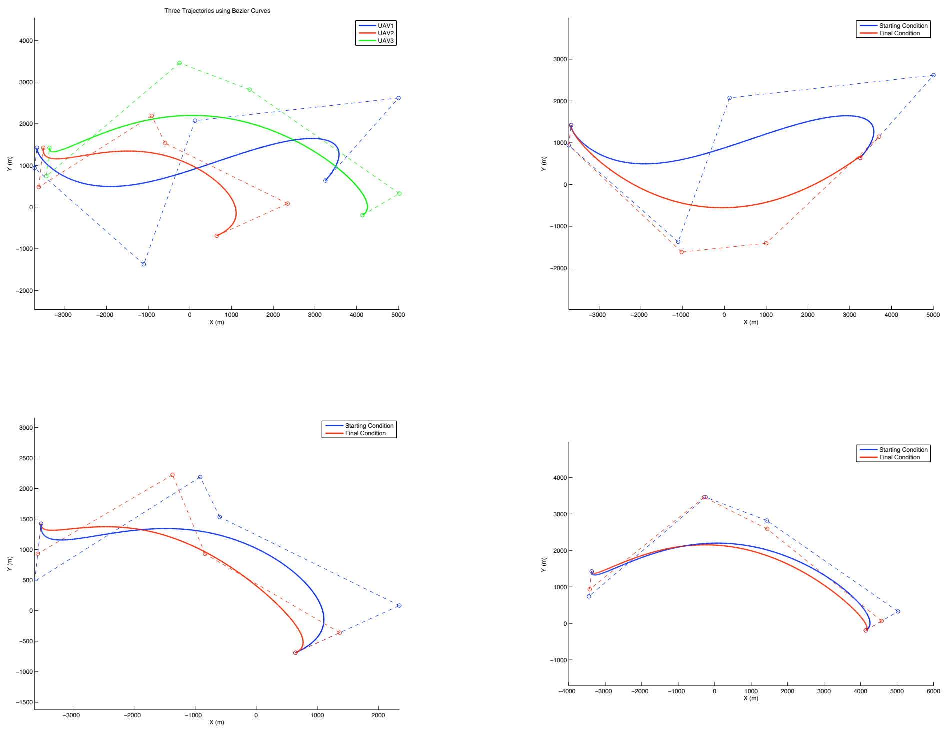

Fig. 8. Spatially Deconflicted Flight Paths and Displacement Performed by the Minimizations

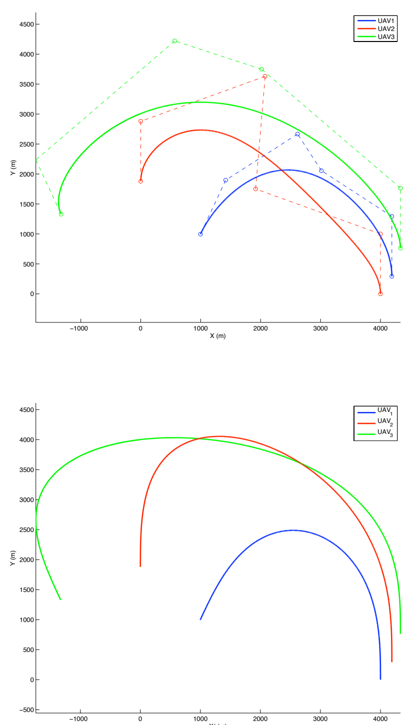

Fig. 6. Comparing Results from the Polynomial Method Presented in [4] and Bézier Paths. Top: Bézier method. Bottom: Polynomial Method (after [4])

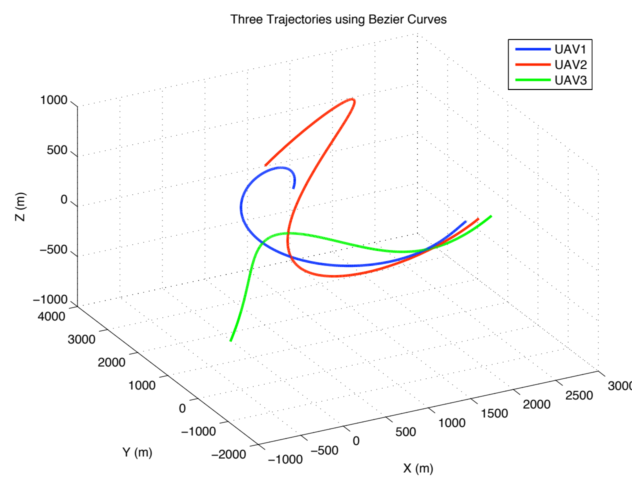

Fig. 7. Simulated Flight Trajectory

of the work presented here into its autonomous ground vehicle: The Overbot [2]. Ground tests with the Overbot are scheduled for the end of the year with following flight tests.

VII. Acknowledgments

The authors wish to thank the Mexican National Science and Technology Council (CONACyT) for partially supporting this research.

REFERENCES

[1] W. Boehm and A. Muller. On de Casteljau's algorithm. *Journal of Computer Aided Geometric Design*, 16(7):587–605, August 1999.

[2] G. Elkaim, J. Connors, and J. Nagel. The Overbot: An off-road autonomous ground vehicle testbed. *ION Global Navigation Satellite Systems Conference (ION-GNSS 2006)*, 1, Sept. 22-24, 2006.

[3] I. Kaminer, O. Yakimenko, V. Dobrokhodov, A. Pascoal, N. Hovakimyan, C. Cao, A. Young, and V. Patel. Coordinated path following for time-critical missions of multiple UAVs via $\mathcal{L}_1$ adaptive output feedback controllers. *AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007.

[4] I. Kaminer, O. Yakimenko, A. Pascoal, and R. Ghabcheloo. Path generation, path following and coordinated control for timecritical missions of multiple UAVs. *American Control Conference, 2006*, pages 4906–4913, 14-16 June 2006.

[5] I.I. Kaminer, O.A. Yakimenko, V.N. Dobrokhodov, M.I. Lizarraga, and A.M. Pascoal. Cooperative control of small UAVs for naval applications. *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, 1:626–631 Vol.1, 14-17 Dec. 2004.

[6] M. I. Lizarraga. Autonomous landing system for a UAV. Master's thesis, Naval Postgraduate School, Monterey, CA, USA., 2004.

[7] M. Massari, E. Ceriani, L. Rigolin, and F. Bernelli-Zazzera. Optimal path planning for planetary exploration rovers based on artificial vision system for environment reconstruction. *Advanced Intelligent Mechatronics. Proceedings, 2005 IEEE/ASME International Conference on*, pages 987–992, 24-28 July 2005.

[8] K. Nonami. Prospect and recent research & development for civil use of autonomous unmanned aircraft as UAV and MAV. volume 1 of *Journal of System Design and Dynamics*, 2007.

[9] K. Petrinec and Z. Kovacic. The application of spline functions and bezier curves to agv path planning. *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium on*, 4:1453–1458, June 20-23, 2005.

[10] T. W. Sederberg. Computer aided geometric design. CAGD Course Notes, Brigham Young University, Provo, UT, 84602, April 2007.

[11] The Mathworks, Inc. *Optimization Toolbox User's Guide*. Natick, MA 01760-2098, 2008.

[12] O.A. Yakimenko. Direct method for rapid prototyping of near-optimal aircraft trajectories. *Journal of Guidance, Control and Dynamics*, 23(5):865–875, September - October 2000.