ISSN 2176-5480

22nd International Congress of Mechanical Engineering (COBEM 2013)
November 3-7, 2013, Ribeirão Preto, SP, Brazil
Copyright © 2013 by ABCM

PARALLEL SIMULATION FOR AUTONOMOUS AIRCRAFT SQUADRONS USING VIRTUAL STRUCTURE AND A 3D MANEUVERS SCHEME

Paulo André Sperandio Giacomin
State University of Santa Cruz,Technological and Exact Sciences Department, Informatics area, Ilhéus-BA, Brazil
pasgiacomin@uesc.br

Elder Moreira Hemerly
Technological Institute of Aeronautics, Systems and Control Depart. - Electronics Eng. Division, São José dos Campos-SP, Brazil
hemerly@ita.br

Abstract. *A parallel algorithm for simulation of autonomous aircraft squadrons is proposed in this paper. A 3D maneuvers control scheme is presented for the aircraft to follow the reference trajectory by using a well defined geometric formation, called virtual structure. The simulations, employing the aircraft model, show that the aerial vehicles do not loose formation neither collide, even when noise in the measured signals is considered. The time complexity analysis of the simulation algorithm shows its efficiency is not time critical. An optimization algorithm is used to tune the controller parameters. It is proved that the parallel algorithm is optimal if the controller tuning optimization procedure efficiency is optimal.*

Keywords: *Automatic Control, Autonomous Systems, Analytic Models of Simulation, Parallel Processing*

1. INTRODUCTION

There is growing interest for unmanned aerial vehicles that can be used for solving public security problems. Some examples are the monitoring of forest fires, Martínez-de Dios *et al.* (2007), and the convoy protection, Ding *et al.* (2009).

The unmanned aerial vehicles can be remotely guided by humans, Nestmeyer *et al.* (2013), or they can be autonomous. Autonomous aircraft can work alone or together with other autonomous aerial vehicles, forming an autonomous aircraft squadron, as in Chao *et al.* (2012). Autonomous aircraft squadrons is a distributed system, and has advantages, such as a better use of sensors, Chao *et al.* (2012), reduced costs, Verma *et al.* (2003) and increased system robustness, Min *et al.* (2009). As another advantage, let georeferencing be the problem to be solved by autonomous aircraft squadrons. Then, the images captured by the squadron can be mosaicked, increasing the vision range of the squadron. The quality of the images is improved, since the aerial vehicles fly at low altitudes, and they can solve problems that are difficult to be solved by satellites, Martínez-de Dios *et al.* (2007), like the processing of images with high number of people.

In the case the autonomous aircraft squadrons are used to capture images to be mosaicked, the squadron formation has to be precise. This can be done by defining well defined geometric formations for the squadrons. So, control strategies have to be implemented so that the aerial vehicles can follow these references.

Simulations are frequently used to define control strategies and to predict the behavior of autonomous agents. Simulations can become inefficient with the aerial vehicles quantity growth, since thousands of time steps have to be defined for each aircraft in the squadron. The parallel processing can improve the efficiency.

Parallel architectures can be centralized or distributed. Centralized architectures typically has a good processor use rate and are easy to manage. Distributed architectures are formed by many computers that communicates by networking, and usually has an attractive cost, Tanenbaum (1992). It is possible to find parallel architectures with processors quantity compatible with aerial vehicles quantities frequently considered in recently autonomous aircraft squadrons researches. Thus, the efficiency can be controlled by increasing simultaneously the number of processors and aerial vehicles, and the parallel processing can improve costs and reduce risks when used to simulate autonomous aircraft squadrons.

Therefore, we can summarize the problem addressed in this work: to use a control model of autonomous aircraft squadrons that can cooperate together to achieve the squadron formation, without losing formation, even when noise in the measured signals is considered. The simulation has to be executed efficiently, making the process development more efficient, interactive and productive.

This problem can be divided into several parts: the simulation efficiency, the formation achievement and the control strategy used by the aerial vehicles to follow their references. Each of these sub-problems are considered in the literature, separately or not, with different objectives.

Several approaches found in the literature attempt to solve one or more of the aforementioned sub-problems. The control problem is addressed by the multiagent controller approach of Salichon and Tumer (2012), but formation is not considered. The formation and control problems are addressed by the leader-follower approach of You and Shim (2011) and Gu *et al.* (2009), by the sliding mode approach of Galzi and Shtessel (2006), by the formation command and control management approach of Verma *et al.* (2003), by the formation reconfiguration approach of Venkataramanan and Dogan (2004), by the multi-uav collaboration approach of Maithripala and Jayasuriya (2008), by the Q-Learning fuzzy controller approach of Rui (2010), by the formation by energy preserving approach of Veth *et al.* (1995), by the decentralized information structure approach of Bauso *et al.* (2003) and by the constraint forces approach of Zou and Pagilla (2009).

The well defined geometric formation problem is not the main objective in the particle filter approach of Alejo *et al.* (2009), and in the the potential artificial fields approach of Roussos and Kyriakopoulos (2012). None of the aforementioned studies pursued the simulation efficiency issue.

Other researches, such as the particle swarm optimization of Fu *et al.* (2012), the genetic algorithm-inspired path planner approach of Cheng and Leung (2012) and the minimum-time trajectory planning approach of Xu *et al.* (2012), deal with path planning, but not with formation, and the efficiency is determined numerically.

Some researches deal with well defined formations and control problems, such as the nonlinear model predictive control approach of Chao *et al.* (2012) and Shin and Kim (2009), the virtual structure approach of Min *et al.* (2009) and the adaptable approach of Sattigeri and Calise (2003), but the efficiency is not treated by Min *et al.* (2009) and Sattigeri and Calise (2003) and is investigated only numerically by Shin and Kim (2009). The model predictive control has its efficiency improved by Jansen and Ramirez-Serrano (2011) and Shin and Kim (2009), but there is no proof in Chao *et al.* (2012), Jansen and Ramirez-Serrano (2011) and Shin and Kim (2009) about the optimization algorithm convergence, what is undesirable for real-time applications. The cooperative-game based control approach of Lluch (2002) and Anderson and Robbins (1998) does not deal with well defined geometric formations. The main contributions of this work are:

1. A navigation scheme that employs a defined set of 3D maneuvers created by algorithms using parameters like radius, distance, velocity, initial position and direction angles. With these maneuver~~s~~ specifications, algorithms create position and velocity references. The references are translated for each aircraft in the squadron. A well defined geometric formation is easily implemented with these references, and a better formation stability is achieved.

2. Validation of the proposed algorithms, by means of a case study, considering three aerial vehicles in formation. The new serial and parallel algorithms and the control strategy proposed are used to perform this validation, by using the new 3D maneuvers scheme.

2. METHODOLOGY

In this work, the control strategy proposed by Anderson and Robbins (1998) is reviewed and improved, since it is simple and well tested, and can be used to simulate autonomous aerial vehicles. The analytical model used by them is a three degree-of-freedom, point-mass model. Models of this type are commonly used to investigate optimal maneuvers and aircraft performance.

2.1 The Aircraft Model and Control

The aircraft state space model, see Anderson and Robbins (1998) for details, is given by

$$\frac{dV}{dt} = g \cdot \left[\frac{(T-D)}{W} - sin(\gamma)\right] \tag{1}$$

$$\frac{d\gamma}{dt} = \frac{g}{V} \cdot [n \cdot cos(\mu) - cos(\gamma)] \tag{2}$$

$$\frac{d\chi}{dt} = \frac{g \cdot n \cdot sin(\mu)}{V \cdot cos(\gamma)} \tag{3}$$

$$\frac{dx}{dt} = V \cdot cos(\gamma) \cdot cos(\chi) \tag{4}$$

$$\frac{dy}{dt} = V \cdot cos(\gamma) \cdot sin(\chi) \tag{5}$$

$$\frac{dh}{dt} = V \cdot sin(\gamma) \tag{6}$$

where the state variables are: airspeed (V), flight path angle (γ), flight path heading (χ), and the position variables (x, y, h). It is not possible to change bank angle (μ), load factor (n) or thrust (T) instantaneously, then it is necessary to represent the airplane rotational dynamics by using first-order linear differential equations, namely,

$$\frac{dT}{dt} = \frac{T_c - T}{\tau_t} \tag{7}$$

ISSN 2176-5480

$$\frac{d\mu}{dt} = \frac{\mu_c - \mu}{\tau_b} \tag{8}$$

$$\frac{dn}{dt} = \frac{n_c - n}{\tau_n} \tag{9}$$

The aircraft velocity vector is determined by commanded airspeed (V_c), commanded flight path (γ_c), and commanded flight path heading (χ_c), using the controller bandwidths w_v, w_γ and w_χ, according to

$$\frac{dV_c}{dt} = w_v \cdot (V_c - V) \tag{10}$$

$$\frac{d\gamma_c}{dt} = w_\gamma \cdot (\gamma_c - \gamma) \tag{11}$$

$$\frac{d\chi_c}{dt} = w_\chi \cdot (\chi_c - \chi) \tag{12}$$

The dynamic inversion control law is determined by setting the commanded rates in Eq. (10), (11) and (12) equal to the associated actual rates in Eq. (1), (2) and (3) and solving for thrust, bank angle and load factor. The calculated thrust, bank angle, and load factor become the commanded values used in Eq. (7), (8) and (9). By setting Eq. (10) equal to (1) and resolving for thrust, it follows that

$$T_c = D + w_v \cdot W \cdot \frac{V_c - V}{g} + W \cdot sin(\gamma) \tag{13}$$

Similarly for Eq. (2) and (11) and for Eq. (3) and (12), one obtains

$$a \equiv n_c \cdot cos(\mu_c) = w_\gamma \cdot V \cdot \frac{(\gamma_c - \gamma)}{g} + cos(\gamma) \tag{14}$$

$$b \equiv n_c \cdot sin(\mu_c) = w_\chi \cdot V \cdot (\chi_c - \chi) \cdot \frac{cos(\gamma)}{g} \tag{15}$$

Equations (14) and (15) are used to determine the required bank angle and load factor commands, i.e.,

$$n_c^2 = a^2 + b^2 \tag{16}$$

$$\mu_c = atan\left(\frac{b}{a}\right) \tag{17}$$

The drag force is determined by considering a single drag polar model as

$$D = 0.5 \cdot \rho \cdot V^2 \cdot S \cdot C_{Do} + 2 \cdot k \cdot n^2 \cdot \frac{W^2}{\rho \cdot V^2 \cdot S} \tag{18}$$

The load factor command is restricted so that the aircraft does not exceed structural limits or stall, via

$$n_{stall} = 0.5 \cdot \rho \cdot V^2 \cdot S \cdot \frac{C_{Lmax}}{W} \tag{19}$$

The Anderson and Robbins (1998) work is here improved by using different aircraft references. As can be seen in Eq. (10), (11) and (12), Anderson and Robbins (1998) use direction angles and velocity as aircraft references, but here are used the velocity and the position. Then, the position references have to be converted to the Anderson and Robbins (1998) references by making

$$u_x = k_{px} \cdot (x_c - x) + k_{ix} \cdot \int_0^t (x_c - x) \cdot dt \tag{20}$$

$$u_y = k_{py} \cdot (y_c - y) + k_{iy} \cdot \int_0^t (y_c - y) \cdot dt \tag{21}$$

$$u_h = k_{ph} \cdot (h_c - h) + k_{ih} \cdot \int_0^t (h_c - h) \cdot dt \tag{22}$$

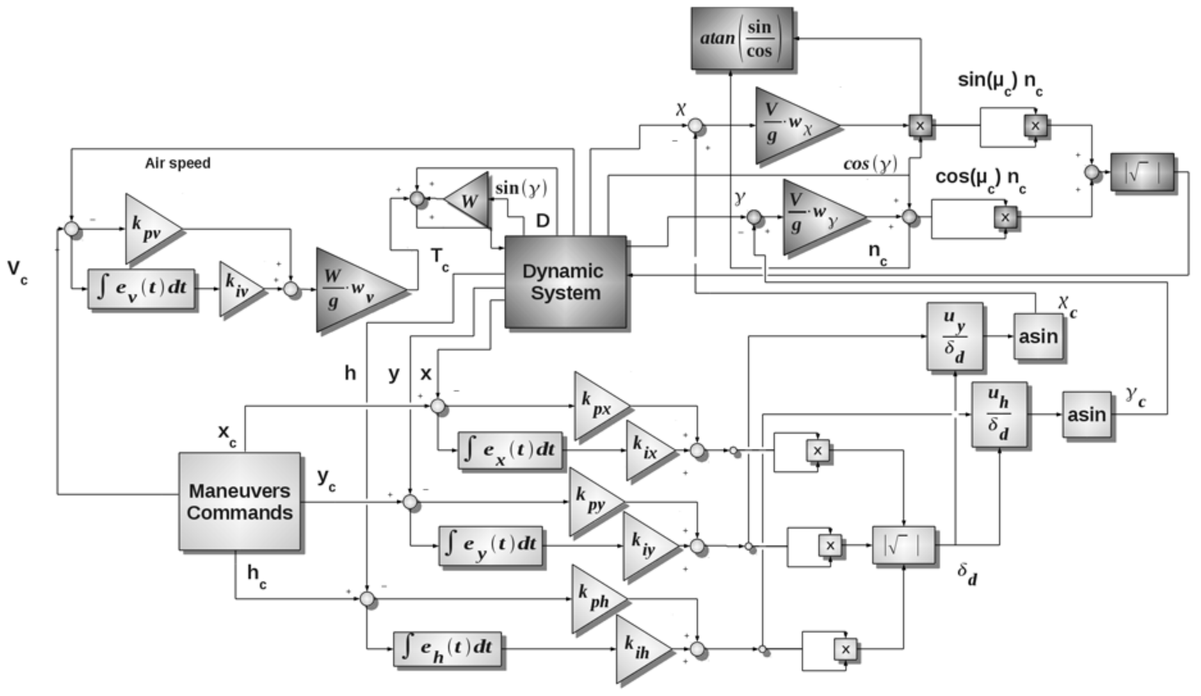
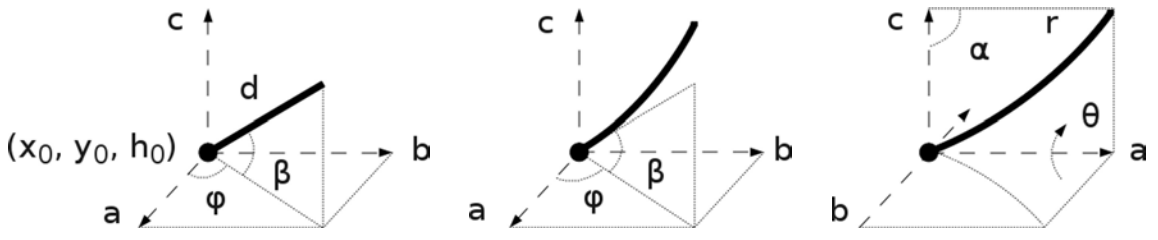


Figure 1. Aircraft Control Scheme.



(a) Go forward with directions (b) To turn with directions (c) To turn with radius

Figure 2. The set of basic maneuvers with $\vec{a} \parallel \vec{x}$, $\vec{b} \parallel \vec{y}$ and $\vec{c} \parallel \vec{h}$.

$$\delta_d = \sqrt{u_x^2 + u_y^2 + u_h^2} \tag{23}$$

$$\chi_c = asin\left(\frac{u_y}{\delta_d}\right) \tag{24}$$

$$\gamma_c = asin\left(\frac{u_h}{\delta_d}\right) \tag{25}$$

The dynamic inverse control used by Anderson and Robbins (1998) is considered an inner control loop. It is not always possible for the aerial vehicles to achieve formation stability after performing obstacles avoidance when only this strategy is used, since the position usually drifts. Therefore, in order to improve formation stability, an outer control loop is proposed here. As is presented in Fig. 1, this is implemented by using a proportional-integral (PI) controller, where the light blocks represent the PI controller and also convert the direction angles and velocity references into the velocity and position references generated by the 3D maneuvers scheme.

2.2 The Maneuvers Scheme

The maneuvers scheme uses a set of elementary maneuvers in 3D space to build trajectories. For achieving this goal, two basic maneuvers are used: to turn and go forward, as showed in Fig. 2. The to turn maneuvers are specified by the direction angles φ, θ and β, the radius r, the initial position (x_0, y_0, h_0) and the α angle. The go forward maneuver is specified by the distance d, the initial position (x_0, y_0, h_0) and the direction angles φ and β.

Algorithms are used to create trajectory references for the leader aircraft using the ~~maneuvers~~ specifications. The algorithm that creates the to turn maneuver is presented in Alg. 1. It uses the function $rotate(w, \vec{s}, p)$, that rotates the point p w *rads* around the vector $\vec{s}$, and it also uses the function $rectangular(angle_1, angle_2, module)$, where $angle_1$ and $angle_2$ are similar to α and β in Figure 2, and translates the 3D point $(angle_1, angle_2, module)$ to its rectangular representation (x, y, h).

Algorithm 1 To turn maneuver creator

```
1:  procedure TO_TURN($x_0, y_0, h_0, \varphi, \beta, \theta, \alpha, r, vel, T$)
2:      if $|T| = 0$ then
3:          $t \leftarrow 0$
4:      else
5:          $t \leftarrow last(T)$                                    ▷ does not modify $T$
6:      end if
7:      $arc \leftarrow vel * \Delta_t$
8:      $\Delta_{arc} \leftarrow arc/r$
9:      $i \leftarrow \pi/2 + \Delta_{arc}$
10:     while $i \leqslant \pi/2 + \alpha$ do
11:         $h \leftarrow 0$
12:         $x \leftarrow -cos(i) \cdot r$
13:         $y \leftarrow (sin(i) - 1) \cdot r$
14:         $point \leftarrow (x, y, h)$
15:         $point \leftarrow rotate(\theta, -\vec{x}, point)$
16:         $point \leftarrow rotate(\phi, \vec{z}, point)$
17:         $\vec{r} \leftarrow rectangular(\phi - \pi/2, 0, 1)$               ▷ from polar to rectangular
18:         $point \leftarrow rotate(\beta, \vec{r}, point)$
19:         $point \leftarrow point + (x_0, y_0, h_0)$
20:         $Ref \leftarrow Ref \cup \{point\}$
21:         $t \leftarrow t + \Delta_t$
22:         $T \leftarrow T \cup \{t\}$                                ▷ externally visible
23:         $i \leftarrow i + \Delta_{arc}$
24:     end while
25:     return $Ref$
26: end procedure
```

The maneuvers presented in Fig. 2 are translated for each aircraft in the squadron, since a virtual structure is kept by the aerial vehicles.

2.3 The Virtual Structure

The to turn and go forward algorithms are applied for the leader aircraft in the virtual structure, and a simple algorithm is used to translate the generated leader aircraft references to each follower aircraft references. This scheme is slightly different from the one presented by Min *et al.* (2009), since it does not use a virtual leader.

The virtual structure is formed by a leader aircraft and by one or more follower aerial vehicles. As presented in Fig. 3, the squadron forms a well defined body, where the follower aircraft tries to keep a fixed distance from the leader. Therefore, three distances have to be specified: the forward distance Δ_A, the lateral distance Δ_B and the vertical distance Δ_C between the leader and each follower aircraft. These parameters and the leader aircraft references are used by the translator algorithm.

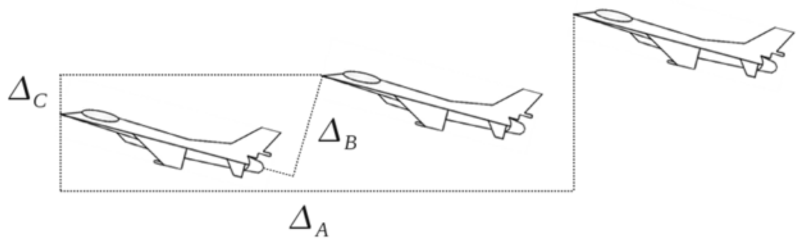


Figure 3. The virtual structure.

The trajectories produced by the translator algorithm are used by the optimization scheme to find appropriated values for the controller parameters in Fig. 1.

ISSN 2176-5480

2.4 The Static Optimization Scheme

The parameters k_{pv}, k_{iv}, k_{px}, k_{ix}, k_{py}, k_{iy}, k_{ph} and k_{ih}, presented in Fig. 1, are tuned by the static optimization scheme. Whereas analytical design procedures would typically demand linear model, or piece-wise linearization, an advantage of the optimization scheme is that it can be used with non-linear aircraft models, such as the one used in this work, with relatively simplicity. The parameters are tuned by the optimization scheme before the aerial vehicles flight. Hence, this scheme is said to be static. It is implemented by using the objective function showed in Alg. 2 with the algorithm of Nelder and Mead (1965).

The inputs of Alg. 2 are the parameters k_{pv}, k_{iv}, k_{px}, k_{ix}, k_{py}, k_{iy}, k_{ph} and k_{ih}, the system initial state X_0 and the position references R, that is a set of (x, y, h) coordinates. The Alg. 2 simulates the aircraft flight by using input parameters and returns the maximum error between the simulated trajectory, stored in X, and the reference trajectory R. The simulation is made by using the proposed PI control scheme. The dynamic inverse control scheme of Anderson and Robbins (1998) is implemented inside the *runge_kutta_4* function. The *runge_kutta_4* function also integrates the system differential equations and obtains the velocity and direction angle references from the position references, and apply them to the dynamic inversion control scheme of Anderson and Robbins (1998).

The main objective of Alg. 2 and the algorithm of Nelder and Mead (1965) is to minimize the cost, that is, to find the parameters that minimize the maximum distance between the simulated trajectory and the aircraft reference.

Algorithm 2 The objective function

```
 1: procedure OBJECTIVE_FUNCTION($k_{px} \cdots k_{iv}$, $X_0$, $R$)
 2:     $X \leftarrow \varnothing$
 3:     $t \leftarrow 0$
 4:     $distance \leftarrow 0$
 5:     $X \leftarrow X \cup \{X_0\}$
 6:     for $i \leftarrow 1, |R|$ do
 7:         $t \leftarrow t + \Delta_t$
 8:         $x_c \leftarrow R(i, 0)$
 9:         $y_c \leftarrow R(i, 1)$
10:         $h_c \leftarrow R(i, 2)$
11:         $X_1 \leftarrow back(X)$                                  ▷ does not remove $x_1$
12:         $u_x \leftarrow k_{px} \cdot (x_c - X_1(3) + k_{ix} \cdot \int_0^t (x_c - X_1(3)) \cdot dt$
13:         $u_y \leftarrow k_{py} \cdot (y_c - X_1(4)) + k_{iy} \cdot \int_0^t (y_c - X_1(4)) \cdot dt$
14:         $u_h \leftarrow k_{ph} \cdot (h_c - X_1(5)) + k_{ih} \cdot \int_0^t (h_c - X_1(5)) \cdot dt$
15:         $\Delta_x \leftarrow R(i, 0) - R(i - 1, 0)$
16:         $\Delta_y \leftarrow R(i, 1) - R(i - 1, 1)$
17:         $\Delta_h \leftarrow R(i, 2) - R(i - 1, 2)$
18:         $v_c \leftarrow \left(\sqrt{\Delta_x^2 + \Delta_y^2 + \Delta_h^2}\right) / \Delta_t$
19:         $u_v \leftarrow k_{pv} \cdot (v_c - X_1(0)) + k_{iv} \cdot \int_0^t (v_c - X_1(0)) \cdot dt$
20:         $X_1 \leftarrow runge\_kutta\_4(u_x, u_y, u_h, u_v, X_1)$
21:         $X \leftarrow X \cup \{X_1\}$
22:         $\Delta_x \leftarrow x_c - X_1(3)$
23:         $\Delta_y \leftarrow y_c - X_1(4)$
24:         $\Delta_h \leftarrow h_c - X_1(5)$
25:         $a \leftarrow \sqrt{\Delta_x^2 + \Delta_y^2 + \Delta_h^2}$
26:         if $a > distance$ then
27:             $distance \leftarrow a$
28:         end if
29:     end for
30:     return $distance$                                            ▷ the maximum error
31: end procedure
```

The Alg. 2 uses as input the references R, that are produced by a set of maneuvers. Here, the references R are produced by a maneuvers specification set, as shown in Fig. 4.

2.5 The Maneuvers Combination Scheme

It is possible to define a test trajectory, as shown in Fig. 4, by using a maneuvers combination set formed by the basic maneuvers of Fig. 2. This same maneuvers combination scheme can be used by Alg. 2 to find the optimum control parameters.

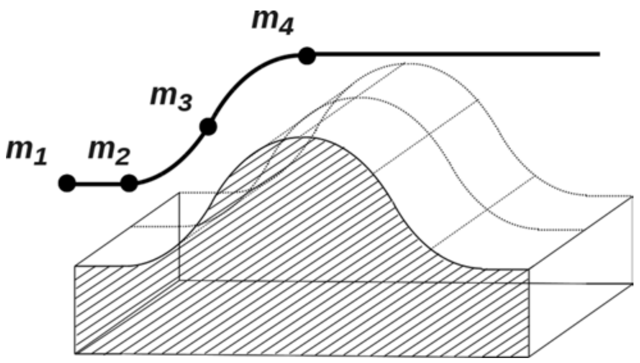


Figure 4. The maneuvers combination set.

A to turn maneuver m is specified by the set $(p, r, \varphi, \beta, \theta, \alpha, vel)$, where p is the (x_0, y_0, h_0) initial position, r is the arc radius, φ, β, θ and α are specified as in Fig. 2, and vel is the aircraft airspeed. The go forward maneuver is specified by $(p, d, \varphi, \beta, vel)$, where p is the initial position (x_0, y_0, h_0), d is the distance and φ and β are the direction angles.

This formalism is used to define the maneuvers of Fig. 4, in the follow way: $N = \{m_1, m_2, m_3, m_4\}$, where $m_1 = (p_1, d, 0, 0, 122)$, $m_2 = (p_2, r, 0, 0, \pi/2, \pi/4, 122)$, $m_3 = (p_3, r, 0, \pi/4, -\pi/2, \pi/4, 122)$ and $m_4 = (p_4, d, 0, 0, 122)$.

The same maneuvers combination set showed in Fig. 4 is used to find the optimal control parameters k_{pv}, k_{iv}, k_{px}, k_{ix}, k_{py}, k_{iy}, k_{ph} and k_{ih}, by using the simulation algorithm described in the next section.

2.6 The Serial Simulation Algorithm

The serial simulation algorithm receives as input a M set of maneuvers, a set of sets of initial states, called X_0 , with one set of initial states for each aircraft, and the quantity n of aerial vehicles, as shown in Alg. 3.

The first part of Alg. 3 uses the trajectory algorithm, showed in Alg. 4. The trajectory algorithm receives as input the same parameters as the serial algorithm. Basically, the trajectory algorithm calculates the references for the leader aircraft by using the maneuvers set M, and calculates the references for the follower aerial vehicles, by using the translator algorithm with the references of the leader aircraft.

2.7 The Parallel Simulation Algorithm

The parallel and serial algorithms are similar, but the parallel algorithm was made to be more efficient, since the parallel algorithm does not have an external loop to iterate over the aerial vehicle's index, because it uses one processor for each aircraft.

As showed in Fig. 5, the parallel algorithm has three rows, where the second row represents the leader aircraft process and the first and third rows represent the follower aircraft processes. The light rectangles represent the send and receive procedures of the message passing scheme.

3. SIMULATION RESULTS

The simulation is made by using the C++ programming language and the algorithm of Nelder and Mead (1965), that is available in the GNU Scientific Library of Galassi *et al.* (2010). The simulation considers one leader and two followers aerial vehicles. Figure 6 shows the dynamics of the aerial vehicles when the test trajectory is applied.

This simulation considers that the initial position (x_0, y_0, h_0) of the aerial vehicles are $(915, 305, 3050)$, $(610, 0, 6100)$ and $(610, 610, 6100)$ for the leader, for the follower aircraft in the left, and for the follower aircraft in the right, respectively. The graph shows all unities in meters. The initial distance between each follower aircraft and the leader is 3080.35 *m*. The initial velocity is 30.5 *m/s* for all aerial vehicles in the squadron and all the other states have negligible initial states.

The minimum distance between the leader aircraft and the left aircraft is 3080.35 *m*, and the same distance is kept between the leader aircraft and the right aircraft. The maximum error between the references and the dynamics for all aerial vehicles is 234.95 *m*. As shown in Fig. 7, the velocity stabilizes at 122.00 *m*, with 0.25% of noise introduced in the velocity sensors, and 1.00% of noise considered for both gama and heading angles. Therefore, the squadron keeps the

Algorithm 3 The serial simulation algorithm

```
1:  procedure SERIAL_SIMULATOR($M$, $X_0$, $n$)
2:      $N \leftarrow an\_initial\_part\_of(M)$
3:      $Traj_1 \leftarrow trajectory(N, X_0, n)$
4:      $Traj_2 \leftarrow trajectory(M, X_0, n)$
5:      for $a \leftarrow 0, n-1$ do
6:          $opt(k_{px}(a) \cdots k_{iv}(a), X_0, Traj_1(a))$                ▷ $k's$ are optimized
7:      end for
8:      for $a \leftarrow 0, n-1$ do                                        ▷ $a$ is the aircraft index
9:          $X(a) \leftarrow X(a) \cup \{X_0(a)\}$
10:         for $i \leftarrow 0, |Traj_2(a)| - 1$ do
11:             $X_1 \leftarrow back(X(a))$                                 ▷ X is not modified
12:             $x_c \leftarrow Traj_2(a, i, 0)$
13:             $y_c \leftarrow Traj_2(a, i, 1)$
14:             $h_c \leftarrow Traj_2(a, i, 2)$
15:             $u_x \leftarrow k_{px} \cdot (x_c - X_1(3) + k_{ix} \cdot \int_0^t (x_c - X_1(3)) \cdot dt$
16:             $u_y \leftarrow k_{py} \cdot (y_c - X_1(4)) + k_{iy} \cdot \int_0^t (y_c - X_1(4)) \cdot dt$
17:             $u_h \leftarrow k_{ph}(h_c - X_1(5)) + k_{ih} \cdot \int_0^t (h_c - X_1(5)) \cdot dt$
18:             $\Delta_x \leftarrow Traj_2(a, i, 0) - Traj_2(a, i-1, 0)$
19:             $\Delta_y \leftarrow Traj_2(a, i, 1) - Traj_2(a, i-1, 1)$
20:             $\Delta_h \leftarrow Traj_2(a, i, 2) - Traj_2(a, i-1, 2)$
21:             $v_c \leftarrow \left(\sqrt{\Delta_x^2 + \Delta_y^2 + \Delta_h^2}\right) / \Delta_t$
22:             $u_v \leftarrow k_{pv} \cdot (v_c - X_1(0)) + k_{iv} \cdot \int_0^t (v_c - X_1(0)) \cdot dt$
23:             $X_1 \leftarrow runge\_kutta\_4(u_x, u_y, u_h, u_v, X_1)$
24:             $X \leftarrow X \cup \{X_1\}$
25:         end for
26:     end for
27:     return $X$
28: end procedure
```

Algorithm 4 The trajectory algorithm

```
1:  procedure TRAJECTORY($Q$, $X_0$, $n$)
2:      for $i \leftarrow 0, |Q| - 1$ do                                    ▷ $Q$ is a set of maneuvers
3:          $traj(0) \leftarrow ref(Q(i))$                                  ▷ get references
4:          for $a \leftarrow 1, n-1$ do                                    ▷ Aircraft 0 is not considered
5:              $\Delta_A \leftarrow X_0(a, 3) - X_0(0, 3)$
6:              $\Delta_B \leftarrow X_0(a, 4) - X_0(0, 4)$
7:              $\Delta_C \leftarrow X_0(a, 5) - X_0(0, 5)$
8:              $traj(a) \leftarrow translator(traj(0), b, \Delta_A, \Delta_B, \Delta_C, V_T)$
9:              $b \leftarrow back(traj(0))$                                ▷ traj is not modified
10:         end for
11:         for $a \leftarrow 0, n-1$ do
12:             $Traj(a) \leftarrow Traj(a) \cup \{traj(a)\}$
13:         end for
14:     end for
15:     return $T$
16: end procedure
```

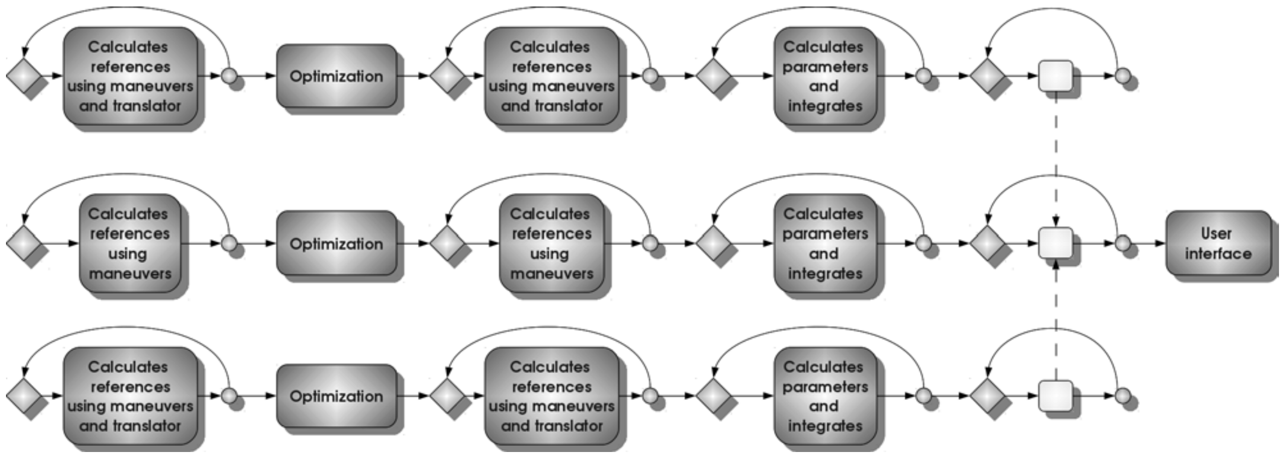

Calculates references using maneuvers and translator
Optimization
Calculates references using maneuvers and translator
Calculates parameters and integrates
Calculates references using maneuvers
Optimization
Calculates references using maneuvers
Calculates parameters and integrates
User interface
Calculates references using maneuvers and translator
Optimization
Calculates references using maneuvers and translator
Calculates parameters and integrates

Figure 5. An parallel and efficient algorithm.

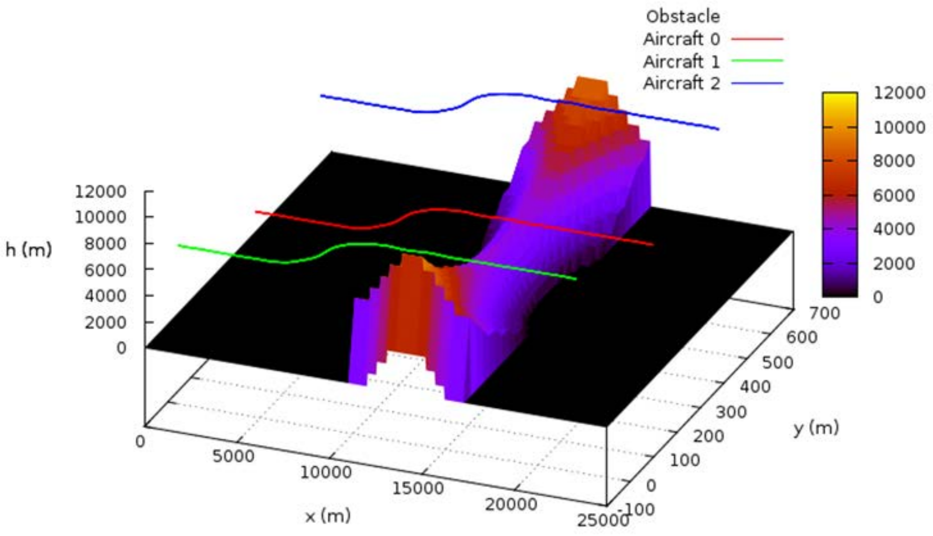


Figure 6. Aircrafts height dynamics

formation, and there is no collision, even when the noise is introduced in the sensors.

4. EFFICIENCY ANALYSIS

The parallel algorithms in Fig. 5 was used to obtain the results presented in Section 3. In this section, the time complexity and the computation time of serial and parallel algorithms are presented and compared. It is assumed that the number of aerial vehicles n is the same as the number of processors.

4.1 The Serial Algorithm

As shown in Fig. 3, the serial algorithm has two main parts: the simulation by integration and the optimization of the k_{pv}, k_{iv}, k_{px}, k_{ix}, k_{py}, k_{iy}, k_{ph} and k_{ih} parameters. The Alg. 4 is used by these two parts. So, the time complexity of Alg. 4 is calculated first, and the time complexity of Alg. 3 is calculated after. The trajectory algorithm iterates over the maneuvers with an external loop, where $|Q|$ is the number of maneuvers. For each maneuver, references are produced by the trajectory algorithm. Be $Q = \{m_0, m_1, \cdots, m_{|Q|-1}\}$ the set of maneuvers, and $R_Q = \{r_0, r_1, \cdots, r_{|Q|-1}\}$ the set of references, with $r_i = ref(m_i)$. Let $S_Q = \{s_0, s_1, \cdots, s_{|Q|-1}\}$ be the set of translated references, with $s_i = translator(r_i)$. Then, the trajectory algorithm has the time complexity given by

$$O\{|Q| \cdot (|r_i| + (n-1) \cdot |r_i| + n \cdot |r_i|)\}$$

$$O\{|Q| \cdot (n \cdot |r_i| + n \cdot |r_i|)\}$$

$$O\{|Q| \cdot 2 \cdot n \cdot |r_i|\}$$

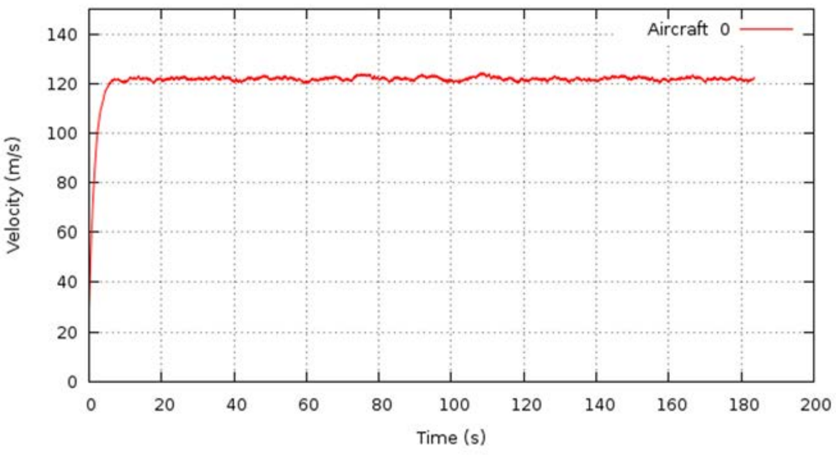


Figure 7. First aircraft airspeed.

$$O(|Q| \cdot |r_i|) = O(|r_0| + \cdots + |r_{Q-1}|) = O(|Traj(0)|)$$

$$O(|Traj(0)| \cdot n) \tag{26}$$

Therefore, the trajectory algorithm has the time complexity proportional to the length of the total of the leader aircraft references times the aircraft quantity. Now, the time complexity of the Alg. 3 is determined. Let C_1 be the time complexity of Nelder and Mead (1965)'s algorithm. Then, the time complexity of the Alg. 3 is

$$O(n \cdot |Traj_2(0)| + n \cdot C_1 + n \cdot |Traj_2(0)|)$$

$$O\{n \cdot (|Traj_2(0)| + C_1)\} \tag{27}$$

The serial algorithm has time complexity proportional to n times the sum between the optimization time complexity and the total of leader references. The simulation part of the serial algorithm can not be executed in less than $O(n \cdot |Traj_2(0)|)$. Therefore, $O(n \cdot |Traj_2(0)|)$ is the lower bound on the time complexity of the simulation problem, when the optimization is not considered. Then, the lower bound on the time complexity of the simulation problem, without the optimization algorithm, is the same as the time complexity of the serial algorithm simulation part. Considering the optimality definition found in AKL (1989), *the serial algorithm is optimal if the efficiency of the optimization algorithm, employed for controller parameter tuning, is optimal.*

4.2 The Parallel Algorithm

A parallelization proposal of the Alg. 3 is the algorithm shown in 5. We can see in Fig. 5 that the lateral column processes have the same time complexity as the middle column process. So, only the time complexity of the middle column process is determined, that is

$$O(|r_0| + |r_1| + \cdots + |r_{|N|-1}| + C_1 + |r_0| + |r_1| + \cdots + |r_{|M|-1}| + |Traj_2(0)|)$$

$$O(C_1 + |r_0| + |r_1| + \cdots + |r_{|M|-1}| + |Traj_2(0)|)$$

$$O(C_1 + |Traj_2(0)| + |Traj_2(0)|)$$

$$O(C_1 + |Traj_2(0)|) \tag{28}$$

Therefore, the time complexity of the parallel algorithm shown in Fig. 5 is n times lower than the time complexity of the serial one. So, if the serial algorithm is optimal, then the parallel algorithm in Fig. 5 is optimal. Since the serial algorithm is optimal if the optimization procedure for controller parameter tuning is optimal, then *the parallel algorithm is optimal if the time complexity on the optimization procedure is optimal.*

4.3 The Computation Time

Although a comparison between the time complexity of serial and parallel algorithm is made in the previous subsection, a more practical comparison between these algorithms can be made by using their computation time.

The serial algorithm spent 5.7401 seconds to simulate the same trajectory processed by the parallel algorithm, that spent 2.2176 seconds. The two algorithms were executed in a laptop that has four cores of 2.53 GHz and 2.8 GB of main memory, running Linux Debian $6.0.7$ with kernel $2.6.32$ 6 686.

The parallel algorithm turned to be 2.59 times faster than the serial one, as far as computation time is concerned. This result is close to that obtained when the time complexity is considered, namely $n = 3$.

Since $O(C_1 + 2 \cdot |Traj_2(0)|) = O(C_1 + |Traj_2(0)|)$, the time complexity analysis is made by considering only the most time demanding expressions of the proposed algorithms. Additionally, the experiment considers a small number of aerial vehicles. These two facts explain the difference between the results. However, this difference is expected to decrease when the number of aerial vehicles and processes increase at the same rate.

5. CONCLUSION

A navigation scheme that employs a defined set of 3D maneuvers was proposed. It was possible to achieve a well defined geometric formation with this scheme by using algorithms that create the references by considering parameters like radius, distance, velocity, initial position and direction angles. This well defined geometric formation was not achieved in Anderson and Robbins (1998).

The proposed control model is implemented with two loops: an inner loop that is formed by the dynamic inverse control employed by Anderson and Robbins (1998) and an new outer control loop that is composed by a proportional-integral controller, which uses position and velocity as references.

Additionally, the efficiency analysis showed that the parallel algorithm time complexity was n times less than the equivalent serial one, being n the same number of aerial vehicles and processors. Equation (28) shows that the time complexity of the parallel algorithm is not a function of n. Then, the same time complexity is kept when the number of aerial vehicles and processors increases. So, it is possible to keep the same efficiency by increasing equally the number of processors and the number of aerial vehicles.

Furthermore, the analysis was made by taking into account C_1, the cost of the optimization algorithm used for tuning the outer control loop. Although C_1 was not determined, the efficiency analysis showed that the parallel algorithm is optimal if the optimization procedure is optimal on its time complexity. Here, the efficiency was analyzed mathematically, in contrast to Shin and Kim (2009), that analyzed the efficiency only numerically. Differently of the dynamic processing found in Chao *et al.* (2012) and Shin and Kim (2009), the optimization algorithm convergence is not critical here, since the optimization is carried out in a static way.

A set of known maneuvers were simulated by the proposed parallel algorithm and the simulation validated the algorithms and control schemes used here. However, the maneuvers were not specified dynamically by the presented algorithms. Therefore, algorithms that specify the maneuvers dynamically would be developed in future researches, and the parallel algorithm would be used to simulate these maneuvers. A proof that there is no collision with the proposed approach can be investigated intuitively by analyzing, for each aircraft pair, the difference between the maximum overshoot of the aircraft positions and the fixed aircraft distance.

6. REFERENCES

AKL, S.G., 1989. *The design and analysis of parallel algorithms*. Prentice-Hall, Inc.

Alejo, D., Conde, R., Cobano, J.A. and Ollero, A., 2009. "Multi-uav collision avoidance with separation assurance under uncertainties". In *Proceedings of the 2009 IEEE International Conference on Mechatronics*.

Anderson, M.R. and Robbins, A.C., 1998. "Formation flight as a cooperative game". *Collection of Technical AIAA Guidance, Navigation, and Control Conference and Exhibit*, Vol. 10, No. 12, pp. 244–251.

Bauso, D., Giarré, L. and Pesenti, R., 2003. "Attitude alignment of a team of uavs under decentralized information structure". In *Proceedings of 2003 IEEE Conference on Control Applications*.

Chao, Z., Zhou, S.L., Ming, L. and Zhang, W.G., 2012. "Uav formation flight based on nonlinear model predictive control". *Mathematical Problems in Engineering*, Vol. 2012, No. 261367, pp. 1 – 15.

Cheng, C. and Leung, H., 2012. "A genetic algorithm-inspired uuv path planner based on dynamic programming". *IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews*, Vol. 42, No. 6, pp. 1128–1134.

Ding, X.C., Rahmani, A. and Egerstedt, M., 2009. "Optimal multi-uav convoy protection". In *Robot Communication and Coordination, 2009. ROBOCOMM'09. Second International Conference on*. IEEE, pp. 1–6.

Fu, Y., Ding, M. and Zhou, C., 2012. "Phase angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for uav". *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, Vol. 42, No. 2, pp. 511–526.

Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Alken, P., Booth, M. and Rossi, F., 2010. *GNU Scientific Library*, 1st edition.

Galzi, D. and Shtessel, Y., 2006. "Uav formations control using high order sliding modes". In *Proceedings of the American Control Conference*.

Gu, Y., Campa, G. and Seanor, B., 2009. *Aherial Vehicles*, InTech, chapter Autonomous Formation Flight - Desing and Experiments, pp. 235–257.

Jansen, F. and Ramirez-Serrano, A., 2011. "Agile unmanned vehicle navigation in highly confined environments". In *IEEE International Conference on Systems, Man, and Cybernetics*. p. 2381–2386.

Lluch, D., 2002. "Building multi-uav simulation methods". In *AIAA Modeling and Simulation Technologies Conference and Exhibit*.

Maithripala, D.H.A. and Jayasuriya, S., 2008. "Feasibility considerations in formation control: Phantom track generation through multi-uav collaboration". In *Proceedings of the 47th IEEE Conference on Decision and Control*.

Martínez-de Dios, J., Merino, L., Ollero, A., Ribeiro, L.M. and Viegas, X., 2007. "Multi-uav experiments: Application to forest fires". In *Multiple Heterogeneous Unmanned Aerial Vehicles*, Springer, pp. 207–228.

Min, H., Sun, F. and Niu, F., 2009. "Decentralized uav formation tracking flight control using gyroscopic force". In *International Conference on Computational Intelligence for Measurement Systems and Applications*.

Nelder, J.A. and Mead, R., 1965. "A simplex method for function minimization". *Computer Journal*, Vol. 7, No. 4, p. 308–313.

Nestmeyer, T., Riedel, M., Lächele, J., Hartmann, S., Botschen, F., Giordano, P.R. and Franchi, A., 2013. "Interactive demo: Haptic remote control of multiple uavs with autonomous cohesive behavior". In *IEEE International Conference on Robotics and Automation*.

Roussos, G. and Kyriakopoulos, K.J., 2012. "Decentralized navigation and conflict avoidance for aircraft in 3-d space". *IEEE Transactions on control systems technology*, Vol. 20, No. 6, pp. 1622 – 1629.

Rui, P., 2010. "Multi-uav formation maneuvering control based on q-learning fuzzy controller". In *2nd International Conference on Advanced Computer Control (ICACC)*.

Salichon, M. and Tumer, K., 2012. "Evolving a multiagent controller for micro aerial vehicles". *IEEE Transactions on Sytems, Man, and Cybernetics – Part C: Applications and Reviews*, Vol. 42, No. 6, pp. 1772–1783.

Sattigeri, R. and Calise, A.J., 2003. "An adaptive approach to vision-based formation control". In *AIAA Guidance, Navigation, and Control Conference and Exhibit*.

Shin, J. and Kim, H.J., 2009. "Nonlinear model predictive control formation flight". *IEEE Transaction on Systems, Man and Cybernetics – Part A: Systems and Humans*, Vol. 39, No. 5, pp. 1116–1125.

Tanenbaum, A.S., 1992. *Modern Operating Systems*. Prentice-Hall, Inc.

Venkataramanan, S. and Dogan, A., 2004. "A multi-uav simulation for formation reconfiguration". In *AIAA Modeling and Simulation Technologies Conference and Exhibit*.

Verma, A., Wu, C.N. and Castelli, V., 2003. "Uav formation command and control management". In *2nd AIAA "Unmanned Unlimited" Conf. and Workshop & Exhibit*.

Veth, M., Pachter, M. and D'Azzo, J., 1995. "Energy preserving formation flight control". In *33rd Aerospace Sciences Meeting and Exhibit*.

Xu, N., Kang, W., Cai, G. and Chen, B.M., 2012. "Minimum-time trajectory planning for helicopter uavs using computational dynamic optimization". In *IEEE International Conference on Systems, Man and Cybernetics*. pp. 2732–2737.

You, D.I. and Shim, D.H., 2011. "Autonomous formation flight test of multi-micro aerial vehicles". *J Intell Robot Syst*, Vol. 61, No. 1-4, pp. 321–337.

Zou, Y. and Pagilla, P.R., 2009. "Distributed formation flight control using constraint forces". *Journal of guidance, control and dynamics*, Vol. 32, No. 1, pp. 112–120.