Design and implementation of distributed path planning algorithm for a fleet of UAVs

Adel Belkadi, Hernan Abaunza, Laurent Ciarletta, Pedro Castillo Garcia, Didier Theilliol

▶ **To cite this version:**

Adel Belkadi, Hernan Abaunza, Laurent Ciarletta, Pedro Castillo Garcia, Didier Theilliol. Design and implementation of distributed path planning algorithm for a fleet of UAVs. IEEE Transactions on Aerospace and Electronic Systems, 2019, 55 (6), pp.2647-2657. 10.1109/TAES.2019.2906437 . hal-02102487

HAL Id: hal-02102487
https://hal.science/hal-02102487v1

Submitted on 12 Jul 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Design And Implementation Of Distributed Path Planning Algorithm For A Fleet Of UAVs

Adel Belkadi*,**, Hernan Abaunza***, Laurent Ciarletta**, Pedro Castillo***, and Didier Theilliol*

***Abstract*—This paper presents the development of a controller for a fleet of Unmanned Arial Vehicles (UAVs) based on a distributed path planning strategy under a multi agent systems framework. The issue, treated as an online optimization problem, is solved using a Particle Swarm Optimization Algorithm (PSO). The proposal was validated in experiments, considering different scenarios like fixed and mobile targets, external disturbances, and the loss of an agent. The proposed PSO is implemented independently in each vehicle in order to determine, by minimizing a cost function, the best paths that ensure the fleet formation control, target tracking and collision avoidance.**

***Index Terms*—Fleet control, Multi Agent Systems, PSO algorithm, Path planning, Defective Agents**

I. Introduction

In recent years, fleet control of autonomous vehicles has attracted increasing attention from researchers around the world. The technological developments such as the miniaturization of the electronic and mechanical components, constantly improving wireless communication devices, the increasing capabilities of the storage and computation allow the conception and operational implementation of multi-agent systems (MAS) [1][2].

In the literature, many problems related to the control of fleets of autonomous vehicles are discussed in [3]. The strategies which are implemented for solving these issues can be classified in three categories: centralized, decentralized and distributed. Each of them has its own advantages and drawbacks. The choice of a strategy depends mainly on resource limitations, computational and communications constraints, the working environment and especially the operational objective [4]. Centralized approaches are generally considered as the simplest way to control multi-agent systems, [5][6], but it requires a considerable computational effort and becomes inappropriate when the MAS dimension increases and makes it very sensitive to communication faults [7].

An effective solution to overcome these disadvantages is to use decentralized or distributed approaches. In the case of distributed control the fleet shares a common algorithm, which is computed by strategically distributing it and sharing information between the agents, arriving to a consented control action for every one of them [8]. Whereas in the case of a decentralized approach, every agent computes an individual algorithm using its own state data and exchanging information with other agents [9]. In order to benefit from the advantages of both approaches, the robustness and fault tolerance of a distributed control, with the scalability and extensibility of the decentralized case, we propose in this paper a new structure based on a decentralized control architecture coupled with a distributed path planning scheme in which a single optimization problem is solved by computing the corresponding terms of every agent in its individual internal processor.

In addition to the different control architectures used to control a fleet of autonomous vehicles, there are also different theoretical tools used to synthesize the control action. A first class of methods proposes to use a consensus protocol aimed to reach a defined objective within the fleet [10]. Rendezvous algorithms, for example, are used in the case where its objective is a common position to be reached by the agents [11][12]. Other algorithms, known as Flocking algorithms in which the goal would be for agents to keep a common direction and speed while moving as a compact group and avoiding collisions [13]-[15]. It should be noted that in this method, algebraic graph theory is the main theoretic tool used to solve the control problem. Recently, some authors have used the design based method [16][17], which focuses formally, on how to design a robust controller for a fleet of autonomous vehicles in order to satisfy specified requirements and achieve output consensus. Game theory is one of the tools used in this context [18].

Another class of methods, called Optimization-based approaches, which interests us particularly in this paper, covers a wide range of control synthesis methods [19][20]. In these approaches, the control objective is formulated as an optimization problem, where the optimal solution is found using some optimization tools and considering given constraints. Model Predictive Control (MPC), which uses the system's model to generate trajectories, is one of the most used methods in this class and has yielded many successful results [21][22].

All the methods above mentioned are often designed for a specific category of systems, where the dynamics are often linear. The application of one of these approaches is therefore strongly related to the types of agents or vehicles that form the fleet. Also, the consideration of perturbations and the heterogeneity of the group are a major problem in the proofs of stability, consensus and others.

- Research supported by the Conseil Regional de Lorraine, the Ministére de L'Education Nationale, de l'Enseignement Supérieur et de La Recherche, and the National Network of Robotics Platforms (ROBOTEX), from France

*,**Universite de Lorraine, CRAN, CNRS, UMR 7039, BP 70239, 54506 Vandoeuvre Cedex, France (e-mail: adel.belkadi@univ-reims.fr) (e-mail: didier.theilliol@univ-lorraine.fr).

**Mines Nancy, Universite de Lorraine, Loria, UMR 7503, France (e-mail: laurent.ciarletta@loria.fr).

***Sorbonne Universites, Universite de Technologie de Compigne, CNRS UMR 7253, Compiegne, France (e-mail: (castillo,habaunza)@hds.utc.fr).

We propose in this paper a control approach partially independent from the general dynamics of the agents, by only taking into account the translation of the vehicles to compute optimal trajectories. The general architecture is a combination of decentralized control and distributed path planning techniques. The advantage of this combination is to dissociate the local control part of each vehicle with its training counterpart. Vehicle trajectories are generated online and locally as a result of a distributed optimization algorithm. A positive scalar potential function is constructed in such a way that its minimum is obtained when the agents reach convergence to the desired configuration, target tracking, collision avoidance, and fault tolerance. Since each vehicle is provided with the information of the position of all the agents in its neighborhood thanks to vision techniques and/or wireless communication capabilities.

This strategy was introduced in our previous work. In [23], the objective function has been minimized in a simplified manner, the goal was to choose among a number of reachable points from a vehicle, the position that minimized at best the objective function. In order to improve this approach we have chosen to use in [24] an optimization method that is more developed and responsive to command our flee. This method is derived from meta-heuristics and is called Particle Swarm Optimization (PSO), which has demonstrated optimal performance for searching minimal terms, with a minimum calculation time and a relatively easy implementation [25]-[29]. In [30], The method has been implemented on a linear system modeled by a double integrator, where a polynomial trajectory generation block has been attached to the optimization part.

In this paper, the PSO algorithm is rehabilitated for controlling of a real fleet of quadrotors governed by non-linear dynamics and operating in a 3-dimensional space. Part of the contribution of this work is the consideration of real world constraints and computational limitations to implement the algorithm in real UAVs. The dynamics of the quadrotors are taken into account in the choice of parameters of the PSO algorithm. A quaternion-based model and a control law is implemente to ensure local stability of the closed-loop of each UAV. This controller was based on previous work from [31]-[33]. Large disturbances and defective agent case are also considered to illustrate the robustness of the presented approach. The proposed algorithm is validated in real-time experiments for a fleet of 3 quadrotors and compared with other techniques in real-time emulations.

The rest of paper is organized as follows. The details of the problem formulation are discussed in Section II. In Section III, we present the proposed distributed path planning solution, while the quadrotor controller is introduced in Section IV Then, a comparison with other methodologies is presented in Section V using real-time emulated tests. Section VI illustrates the performance of the method based on various experiments. Finally, a conclusion and perspectives are presented in the last section.

II. Preliminaries and problem statement

In general, an agent refers to a dynamic system. In this paper, the term 'agent' is interchangeable with "Autonomous vehicle", which corresponds to a type of non-linear dynamic system. Each agent is equipped with a local controller that ensures its local stability. A fleet consists of a set of agents $N = 1, ..., n$ where n is the number of vehicles. Information exchanged among agents are modeled by means of graph theory.

Describe the system by a weighted graph $G(N, \epsilon(t), A(t))$ where N and $\epsilon(t) \in n \times n$ are the set of nodes and the set of arcs respectively of the graph $G(t)$ and $A(t)$ a matrix of $M_n(R)$ of elements in $\mathbb{R}$ as $a_{ij}(t) > 0$ if : $(j, i) \in \epsilon(t)$. At each sample i is an incoming neighbor of j and j is an outgoing neighbor of i.

G is assumed to be an undirected graph such as : $\forall(i, j) \in \epsilon(t) \Rightarrow (j, i) \in \epsilon(t)$. In this case the matrix $A(t)$ is symmetrical with $a_{ij}(t) = a_{ji}(t)$, where the entries represent the influence between agents i and j, which depend on the sets of elements $d_{ij}(t)$ that represent the distance between two agents i and j, and form a matrix defined as $\partial(t) \in \mathbb{R}^{n\times n}$. Similarly, the desired distances between agents are defined by d^d_{ij}, and are used to construct the matrix ∂_d., while the safety distance is denoted as c (where $c < l$) and l symbolizes the scope of the neighborhood.

Define for each i node a set of neighbors $\Xi_i(t)$ such that:

$$\Xi_i(t) = \{j \in \mathbb{N} : \|x_j(t) - x_i(t)\| \leq l\}, \tag{1}$$

where $\Xi_i(t)$ is called the metrical neighborhood of the robot i, and $x_i(t)$ symbolizes its position. Then, consider a second type of neighborhood called topological neighborhood, in this case only λ elements of the set $\Xi_i(t)$ from the closest node i are considered. The λ nodes represent a new $\Xi_i(t)'$ set with $\Xi_i(t)' \subset \Xi_i(t)$.

The control objectives are then:

1) To bring the fleet from an initial geometrical configuration $\partial(t_0)$ to a desired geometrical configuration ∂_d

$$\lim_{t\to+\infty} \partial(t) = \partial_d. \tag{2}$$

2) Ensuring target points tracking, which position is defined as P_d

$$\forall i \in N, \lim_{t\to+\infty} d_{ip}(t) = d^d_{ip}. \tag{3}$$

where $d_{ip}(t)$ and d^d_{ip} are respectively the real and desired distance between agent i and the target point.

3) To avoid collisions between interacting agents

$$\forall i, j \in N, i \neq j : \|x_i(t) - x_j(t)\| > c. \tag{4}$$

4) Ensuring the continuation of the mission in the event of a loss of an agent .

The issue is formulated as an online optimization problem where a positive scalar cost functions $\Lambda(t)$ is constructed and divided amongst all the agents in such a way that its minimum is obtained when the fleet reaches the defined objectives.

The main idea is to find at each step time the best displacement $h_i^*(t)$ which yields the minimum cost function:

$$h_i^*(t) \rightarrow \min(\Lambda(t)), \tag{5}$$

and

$$Y_i(t+\tau) = x_i(t) + h_i^*(t). \tag{6}$$

where $Y_i(t+\tau)$ is the desired reference input of agent i, representing a position reference in the global inertial frame.

Figure 1 illustrates the general control scheme for the proposed algorithm, note the state information is exchanged between UAVs, and a ground station is used only to define the desired fleet formation references.

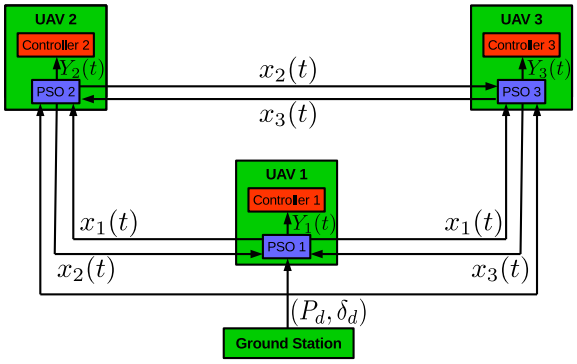


Fig. 1: Distributed path planning structure

III. Distributed path planning design

Most fleet controllers are designed for a specific category of vehicles, where the dynamics of the agents are taken into account in the design of the control laws [35][36]. The results are therefore valid only for a particular type of system and are generally not usable for other applications.

A technique is here introduced, based on a distributed path planning where each vehicle is equipped with a part of an optimization algorithm. The trajectories are therefore not computed in a central unit but in a distributed manner on a vehicle team, contrary to controllers that are completely decentralized.

The novelty of the proposed fleet control is the usage of a PSO algorithm for computing trajectory points in real time, such that each particle represents a displacement on a plane at a time t. Then, the optimization of the proposed cost function makes it possible to find the best displacement that guarantees different objectives of the mission.

Another advantage of the proposed approach is that collision avoidance and loss of agents from the fleet are managed directly in the optimization and path planning modules. An application on a group of real UAVs is illustrated in the last part of the paper.

A. Control structure architecture

As previously explained, an agent is considered as a non-linear dynamic system with a local controller. The stability of agent i, is therefore ensured locally by a controller, which is dependent only on the agent's own state information. Each agent is connected uniquely with its closest neighbors such that its individual optimization block uses the position information its neighbors to compute its trajectory according to certain constraints resulting from its dynamics.

Therefore, the communication between agents is only used to share their positions and the location of a point in space, which is considered as a target to be tracked, each UAV then computes its own trajectory, thus resulting in a distributed path planning, which is tracked using individual controllers for every agent.

B. Construction of the Cost function

A positive cost function is designed for the fleet of drones, and then distributed amongst all of them. In the context of motion planning, such function is built by taking into account the distance between agents and the location of an element considered as a target for the fleet point P_d. Its construction also takes into account information to ensure collision avoidance. A positive cost function $\Lambda(t)$ is proposed which ensures the fleet control objectives by minimizing it its minimization at each step time.

First, let us define a cost function proposed in this paper $\Lambda(t)$, which is divided amongst the agents such that each agent i optimizes a part of the function using its own information and of its neighbors j such that:

$$\begin{array}{l} \Lambda(t) = \sum_{i=1}^{n} \Lambda_i(t), \\ \Lambda_i(t) = \rho\Big(\|P_d - [x_i(t) + h_i(t)]\| - d_{ip}^d\Big) \\ + \sum\limits_{j=1}^{k} a_{ij}(t)\Big(\|x_j(t) - [x_i(t) + h_i(t)]\| - d_{ij}^d\Big) \end{array}, \tag{7}$$

where

$$a_{ij}(t) = 1 + exp\left(\frac{c - d_{ij}(t)}{\sigma}\right), \tag{8}$$

with $i \neq j$, $k = \text{card}(\Xi_i'(t))$, $\rho >> 1$ and $\sigma \neq 0$ denote constant gains, while d_{ip}^d represents the desired distance between agent i and target point.

The choice of ρ depends essentially on the number and type of neighbors of each agent and plays a role on the rate of convergence towards the target,while σ depends on how strongly the agents must react to avoid collisions. A compromise should be found between target monitoring, collision avoidance, and training control. Since an homogeneous fleet is considered in this work, ρ has the same value for all the vehicles.

The main goal is to find the best vector $h_i^*(t)$ for each vehicle i minimizing the cost function $\Lambda_i(t)$ such that

$$\forall i \in N : \lim_{t\to\infty} \Lambda_i(t) = 0 \Rightarrow \tag{9}$$

$$\Rightarrow \left\{ \begin{array}{l} \lim_{t\to\infty} \partial(t) = \partial_d \\ \forall i \in N, \lim_{t\to+\infty} d_{ip}(t) = d_{ip}^d \\ \forall i, j \in N, i \neq j : \|x_i(t) - x_j(t)\| > c \end{array} \right. ,$$

where $h_i^*(t)$ represents the desired displacement relative to the local frame of agent i between two instants t and $t+\tau$. The reference trajectory at time $t+\tau$ for the agent i becomes

$$h_i^*(t) = Y_i(t+\tau) - x_i(t). \tag{10}$$

During the evolution of the vehicles in R^2 space, non-collision of the agents should be ensured. This constraint is introduced in the cost function $\Lambda_i(t)$ through function $a_{ij}(t)$ which is all the greater when $d_{ij}(t) < c$ and $a_{ij}(t) \to 1$ when $a_{ij}(t) >> c$. So, each agent i is more likely to favor the values of $h_i(t)$ which avoids the need for the distances $d_{ij}(t) < c$ and minimizes the cost function $\Lambda_i(t)$, thanks to the PSO algorithm.

In order to have a more stable behavior of the fleet in the proximity of the minimum, a minimum value of the cost function $\Lambda_{i\mathrm{MIN}}$ is considered satisfactory. In this case, when $\Lambda_i(t) < \Lambda_{i\mathrm{MIN}}$, then the value of $h_i^*(t)$ is defined as the zero vector, since the function will be optimized in real-time, the quadrotors will not move only when the fleet is very close to the desired formation.

The choice of the step time τ as well as the search intervals for the displacements $h_i(t)$ of each agent are obtained empirically, and are considered as optimization constraints, such that:

$$h_{i\mathrm{MIN}}(t) < h_i(t) < h_{i\mathrm{MAX}}(t). \tag{11}$$

The choice of $h_i(t)$ thus depends on the choice of τ.

C. Faulty agent case

Assuming every quadrotor is capable of communicating if it becomes defective (actual fault detection algorithms are out of the scope of this work), it is considered that each agent can take into account the presence of a defective neighbor in $\Xi_i'(t)$, then let $K(t)$ be a diagonal matrix $n \times n$ of elements $\delta_i(t)$, where:

1) $\delta_i(t) = 1$ Agent i is free of faults.
2) $\delta_i(t) = 0$ Agent i loses its effectiveness totally and its output is stuck at zero.

The new adjacency matrix $A'(t)$ is defined as

$$A'(t) = A(t) \times \mathrm{diag}(\delta_1(t), \delta_2(t), ..., \delta_n(t)). \tag{12}$$

A defective agent can be then modeled in the cost function as

$$\Lambda_i(t) = \rho\Big(\|P_d - [x_i(t) + h_i(t)]\| - d_{ip}^d\Big) + \sum_{j=1}^{q} \delta_j(t) \times a_{ij}(t)\Big(\|x_j(t) - [x_i(t) + h_i(t)]\| - d_{ij}^d)\Big). \tag{13}$$

Note the zero value of $\delta_j(t)$ makes it possible to cancel $a_{ij}(t)$ and thus eliminates the influence of the defective agent j on agent i. The loss of an agent is therefore taken into account in the cost function, which enables the generation of the correct trajectories for achieving the mission.

D. Particle Swarm Optimization

The Particle Swarm Optimization (PSO) algorithm is initialized by a random population of potential solutions $x(t)$, interpreted as particles moving in the search space where $V(t)$ represents their speed. Each particle is attracted to its best position discovered in the past, noted P_1 as well as to the best position of the particles discovered by its neighborhood P_2. The algorithm includes several setting parameters to act on the compromise Exploration - Exploitation.

The classic PSO algorithm is written as [34]:

$$\begin{aligned} V(t+1) &= aV(t) + b_1 r_1\big(P_1(t) - x(t)\big) \\ &\quad + b_2 r_2\big(P_2(t) - x(t)\big) \\ x(t+1) &= x(t) + V(t+1) \end{aligned}, \tag{14}$$

where a is the coefficient of inertia, b_1 and b_2 represent the intensity of attraction and r_1,r_2 denote two random values between 0 and 1. The best particles $P_1(t)$ and $P_2(t)$ are selected from random samples by evaluating the cost function (7) and selecting the values that give a minimum result.

To simplify the study, we consider the deterministic version of the algorithm, where random numbers are replaced by their average values$(1/2)$, then the algorithm can be rewritten as

$$V(t+1) = aV(t) + b\big(P(t) - x(t)\big) \tag{15}$$

$$x(t+1) = x(t) + V(t+1), \tag{16}$$

where $P(t) = \dfrac{b_1 P_1(t) + b_2 P_2(t)}{b_1 + b_2}$ and $b = \dfrac{b_1 + b_2}{2}$.

To make a dynamic analysis of the algorithm, the equations (15) and (16) are rewritten in the matrix form, the equations of the algorithm can written as

$$\begin{pmatrix} x(t+1) \\ V(t+1) \end{pmatrix} = A_{\mathrm{PSO}} \begin{pmatrix} x(t) \\ V(t) \end{pmatrix} + P(t), \tag{17}$$

where $\begin{pmatrix} x(t) \\ V(t) \end{pmatrix}$ is the state of the system, consisting on the position of the particle and its velocity, P denotes the system input, $A_{\mathrm{PSO}} = \begin{bmatrix} 1-b & a \\ -b & a \end{bmatrix}$ represents the dynamical matrix, and $\begin{bmatrix} b \\ b \end{bmatrix}$ symbolizes the input matrix.

The equilibrium point of the system is such that the particle is positioned in $x(t) = P$, and has zero speed $V(t) = 0$.

The behavior of the particle depends on the eigenvalues λ of the matrix $A_{\mathrm{PSO}}(t)$ and are the solutions of : $\det(\lambda I - A_{\mathrm{PSO}}) = 0$, i.e

$$\lambda^2 - (a - b + 1)\lambda + a = 0. \tag{18}$$

The convergence of the algorithm depends on parameters a and b. The analysis of equation (18) leads to determining the area of convergence of the PSO according to their values. Algorithm 1 shows in more detail, the most important steps considered in the proposed strategy to manage the fleet of autonomous vehicles.

Algorithm 1 Trajectory generation

. Initialize parameters
. Define the desired configuration for the UAVs with the matrix ∂_d
repeat {at each step time}
. Calculate the matrix $\partial(t)$
if $\Lambda_i(t) \leq \Lambda_{i\text{MIN}}$ **then**
. $h[h_x, h_y] = [0, 0]$
. Calculate the next position (x_{id}, y_{id}) at time $t + \tau$:
$x_{id}(t + \tau) = x_i(t)$
$y_{id}(t + \tau) = y_i(t)$
else
. Find all topological neighbors for quadrotor i by set $\Xi_i(t)$
. Minimization of the cost function $\Lambda_i(t)$ by PSO Algorithm
. Chose the vector $h[h_x, h_y]$ that minimizes the cost function $\Lambda_i(t)$
. Calculate the next position (x_{id}, y_{id}) at time $t + \tau$:
$x_{id}(t + \tau) = x_i(t) + h_x$
$y_{id}(t + \tau) = y_i(t) + h_y$
end if
until End of experiment

In order to validate the proposed PSO algorithm in real flight tests, an appropriate position control law needs to be selected such that the Y_i from (10) will be correctly followed.

IV. Quadrotor quaternion dynamic model

The developed method can be tested on different arranges of robotic agents, in this case, a fleet of quadrotors was considered to be operating in the 3-dimesional space.

Many different controllers can be chosen for proving the PSO technique, in this case, a quaternion approach was chosen due to its robustness and mathematical simplicity. The quaternion-based dynamic model of a quadrotor is defined as

$$\frac{d}{dt}\begin{bmatrix} p \\ \dot{p} \\ \boldsymbol{q} \\ \omega \end{bmatrix} = \begin{bmatrix} \dot{p} \\ \boldsymbol{q} \otimes \dfrac{bF_{th}}{m} \otimes \boldsymbol{q}^* + \overline{g} \\ \dfrac{1}{2}\,\boldsymbol{q} \otimes \omega \\ J^{-1}\,(\tau - \omega \times J\,\omega) \end{bmatrix}, \tag{19}$$

where $p \in \mathbb{R}^3$ and $\dot{p} \in \mathbb{R}^3$ are respectively the vehicle position and velocity, F_{th} defines the thrust generated by the motors and $b = [\,0\;\;0\;\;1\,]^T$ its direction, m and $\overline{g}$ represent the vehicle mass and gravity vector, while $\boldsymbol{q}$ describes its attitude as a quaternion, and ω denotes its angular velocity, J introduces the inertia matrix, and τ is the torque vector caused by the action of the motors.

A position feed-back control law is proposed to regulate the translational dynamics from (19) to desired reference

$$u_{\text{pos}} = -K_{\text{pos}}\,(x_{\text{pos}} - x_{\text{pos}_d}), \tag{20}$$

such that (19) is regulated to a desired translational reference.

Where $x_{\text{pos}} = [\,p\;\;\dot{p}\,]^T$, $K_{\text{pos}} \in \mathbb{R}^3$ denotes a diagonal gain matrix with all positive entries, and $x_{\text{pos}_d} = [\,Y(t)\;\;0\,]^T$ represents the desired reference vector, in which $Y(t)$ is the computed PSO trajectory extracted from (10).

The vehicle vertical thrust vector is then rotated to coincide with the required force w.r.t the inertial frame from (20). The required rotation can be computed by

$$\begin{aligned} \boldsymbol{q}' &= (b \cdot u_{\text{pos}} + ||u_{\text{pos}}||) + b \times u_{\text{pos}} \\ \boldsymbol{q}_d &= \frac{\boldsymbol{q}'}{||\boldsymbol{q}'||} \\ F_{th} &= ||u_p|| \end{aligned}, \tag{21}$$

where $\boldsymbol{q}_d$ denotes the required rotation defined as a quaternion. In order to stabilize the quadrotor rotational dynamics $\boldsymbol{q} \to \boldsymbol{q}_d$ an attitude controller is defined as

$$u = -K_{\text{att}}\,(x_{\text{att}} - x_{\text{att}_d})\,, \tag{22}$$

where $K_{\text{att}} \in \mathbb{R}^3$ denotes the control gains, $x_{\text{att}} = [\,(2\ln \boldsymbol{q})^T\;\;\omega^T\,]^T$, and the attitude reference is given by $x_{\text{att}_d} = \big[\,(2\ln \boldsymbol{q}_d)^T\;\;(\frac{d}{dt}2\ln \boldsymbol{q}_d)^T\,\big]^T$. For more details, please refer to [31]-[33].

V. Emulated tests

To demonstrate the effectiveness of the proposed method. The selected approaches were Artificial Potential Fields (APF), based on the work presented in [1], and Model Predictive Control (MPC) [19], using an adaptation of [37].

Using the Fl-Air framework, developed at the Heudiasyc laboratory [38], 20 tests were performed in a real-time emulation environment for each one of the three techniques, considering initial conditions generated at random for every simulation. The objective was to take three quadrotors from their random initial positions to a symmetrical formation around a fixed target.

Figures 2 and 3 depict the position of the vehicles during the simulations, note their initial positions are scattered throughout the plots. The desired distance the fleet must maintain towards the target is represented by a circle.

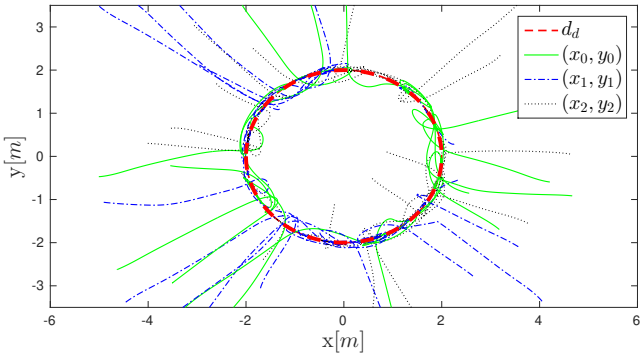
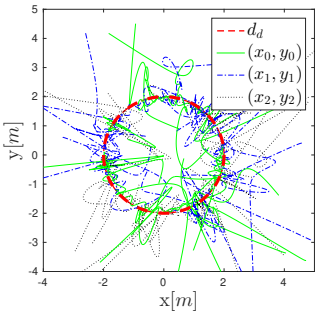
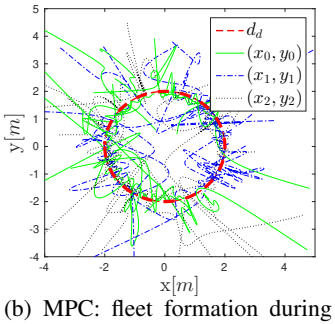


Fig. 2: PSO: Horizontal position of the quadrotors in real-time simulations, presenting a robust convergence with small oscillations.



(a) APF: real-time feet formation, 20 simulations.



(b) MPC: fleet formation during 20 simulations.

Fig. 3: Quadrotor trajectories for 40 simulations of APF and MPC algorithms for fleet formatio, notice how strong oscillations happen, and the vehicles come dangerously close to the center in some cases.

Note although the three approaches stabilize the vehicles at the desired distance, the convergence of the proposed PSO algorithm is smoother, presents less oscillations, and the computed trajectories respect a safer distance between the quadrotors and the target.

Figures 4 and 5 illustrate the behavior of the simulated fleet by displaying the average horizontal distance between the quadrotors and the averaged distance towards the target. The PSO algorithm displays a smoother convergence towards the fleet formation with more robustness to unfavorable initial conditions than the other two approaches.

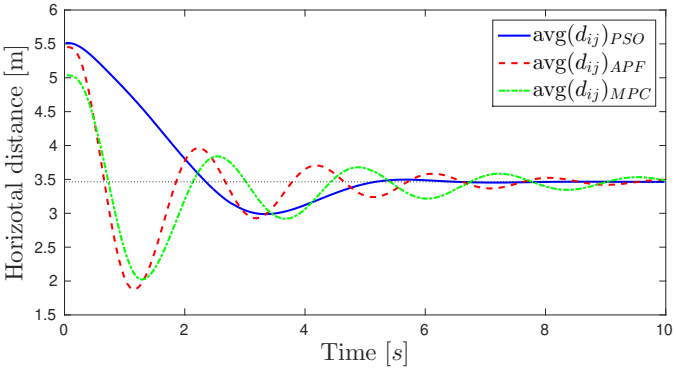


Fig. 4: Averaged separation between quadrotors for 20 real-time simulations for PSO, APF, and MPC approaches.

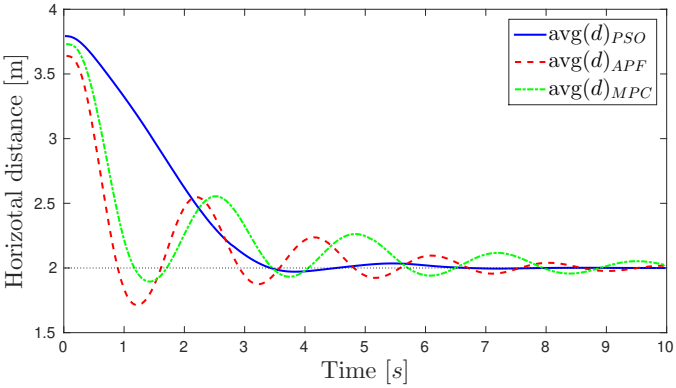


Fig. 5: Average distance from UAVs to target for each algorithm, converging to $d_d = 2m$.

VI. Experimental validation

The proposed algorithm was coded and tested in real-time experiments on a fleet of three quadrotors. All the drones were programmed with the same code. The position of the agents is estimated using an OptiTrack Motion Capture system, and broadasted to all the UAVs.

Each quadrotor computes its own trajectory, such that an uniform fleet is formed around a given target while avoiding collisions amongst each other. A triangular formation is expected under this configuration.

The parameters considered in for the PSO algorithm are depicted in Table I.

$\rho = 5$	$h_{xi}, h_{xi} \in [-0.4m, 0.4m]$
$\sigma = 0.4$	$\tau = 0.1s$
$c = 0.5m$	$d_{ij}^d = 1.732$
$\Lambda_{i\text{MIN}} = 0.1$	$d_d = 3.464$
$\|\|h_{i\text{MAX}}\|\| = .02$	$\|\|h_{i\text{MAX}}\|\| = 0.7$

TABLE I: PSO considered parameters

The selected UAVs were Parrot ARDrone2 with a modified firmware using a framework created by members of the Heudiasyc Laboratory at the Universite de Technologie de Compigne. All the results presented in the following are derived from the actual experiments.

The computational resources of this drone are quite limited, which is expected due to its low cost. The number of particles for $\Xi_i(t)$ was set to be 80, due to a compromise between the trajectory precision and speed for all the other computing required to fly.

The optimization constraints also consider a bounded region for generating random particles, defined by the values $h_{i\text{MIN}}$ $h_{i\text{MAX}}$, ensuring the particles are generated close to the current position of the UAV.

A. Fleet of three quadrotors with fixed target and no disturbances

In this first experiment, the desired configuration for the UAVs from the matrix ∂_d is fixed. The elements $d_{ij}(t)$ are defined such that the desired distance between UAVs is homogeneous, thus arriving to a triangle configuration around the target point, as illustrated in Figure 6.

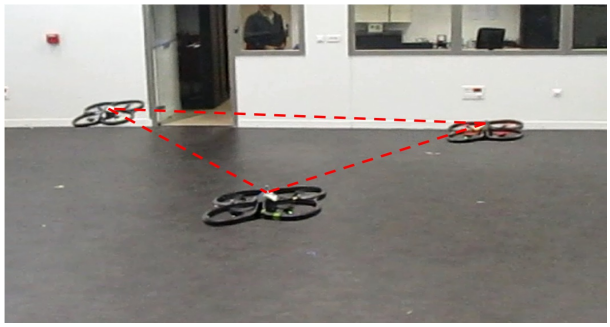


Fig. 6: Three quadrotors in triangular formation.

Four punctual coordinates were considered as targets which switched one to another with a fixed lapse. The optimization algorithm computes the desired position, forming a trajectory which makes the fleet arrive at the desired objectives.

Figure 7 represents the fleet translational behavior in a 2D space. Note the triangular formation is broken when the target changes from one place to another due to the distance from the current location of the fleet, but it is recovered during the evolution of the PSO algorithm.

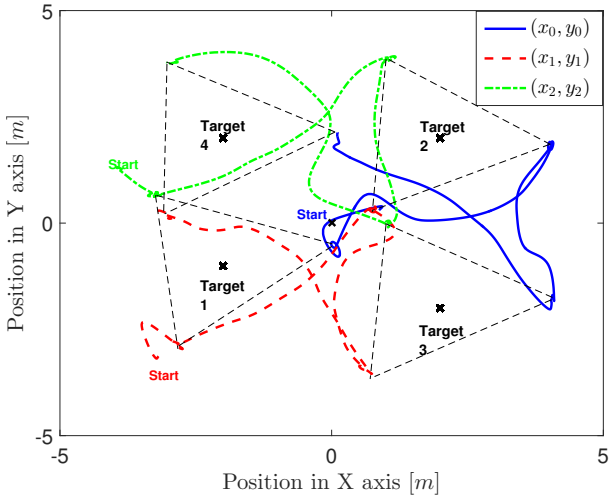


Fig. 7: 2D targets, trajectory and drone positions

An important issue that is addressed in the fleet formation problem is the stabilization of the distance between agents. In this case, a uniform triangular formation is expected with a desired distance of $1.9m$ towards the target, using simple geometry, this implies that the desired distance between agents is $d_{\text{agents}} = 2(1.9m)\cos(\pi/6) = 3.3m$.

The computed references for the agents, ensure the convergence of the distance between every pair of agents to the desired value, this behavior is illustrated in Figure 8.

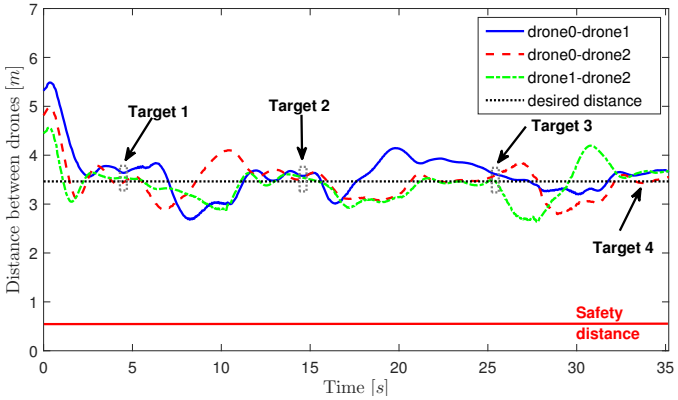


Fig. 8: Desired distance between agents in a real-time experiment, the terms UAV and drone are considered to be equivalent in the context of this work.

Figure 9 represents the desired distance between each agent and the target, note its smooth convergence to the desired value even when abrupt changes occur.

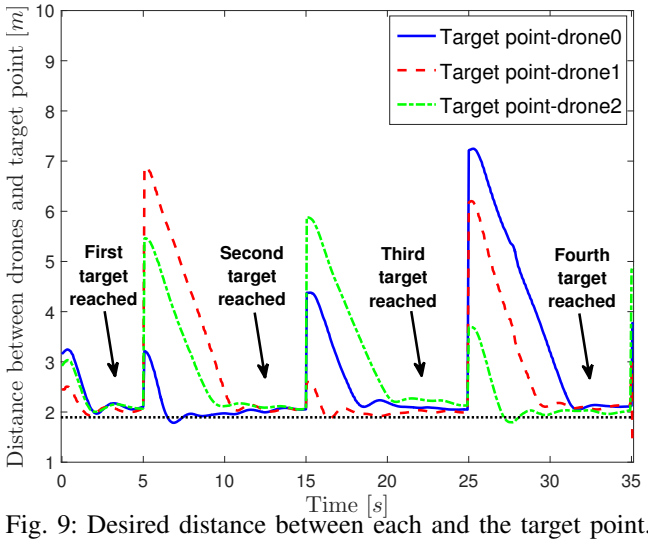


Fig. 9: Desired distance between each and the target point.

B. *Fleet of three quadrotors with fixed target and disturbances*

In the second experiment, the same flight configuration was considered, but unknown disturbances were added. Here, one of our team members pulled one of the agents as it flew towards the target (see Figure 10), the PSO algorithm compensated the disturbance and corrected the agent path.

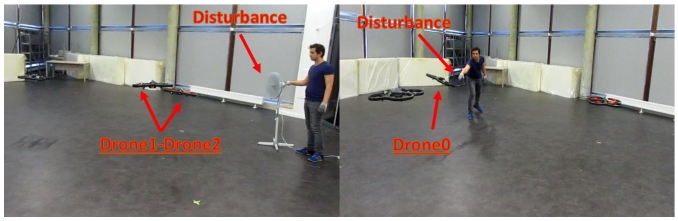


Fig. 10: Disturbances induced to the fleet during experiments.

Distances between agents are illustrated in Figure 11, note that strong disturbances occur at some instants, but the PSO algorithm recovers the position of the affected UAV in a smooth manner, returning to the desired value.

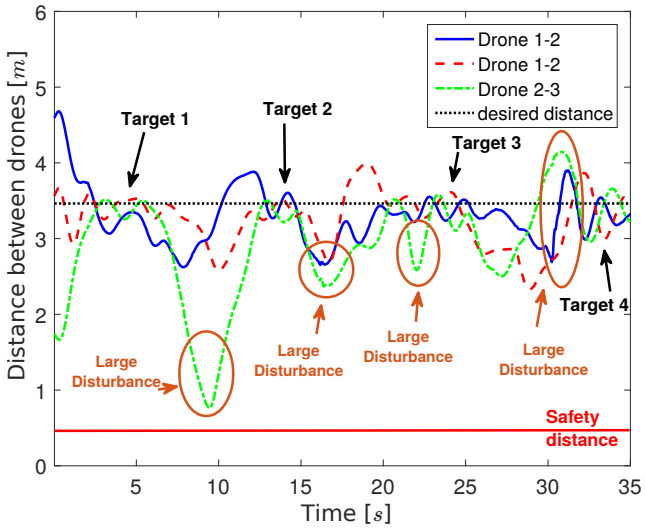


Fig. 11: Desired distance between agents in the presence of strong perturbations.

C. Fleet of three quadrotors with mobile target and disturbances

In this experiment, the movement of the target was controlled using a joystick connected to a ground station, such that the trajectories for every UAV take the fleet towards the moving reference. Additionally, strong disturbances were applied as depicted in Figure 12.

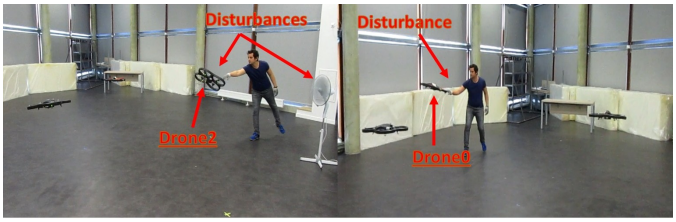


Fig. 12: Inducing disturbances on the moving fleet.

Figure 13 illustrates the distance between agents, while Figure 14 represents the distance between each agent and the target, letters (a,b,c,d) correspond to each disturbance. Note the convergence towards the desired values, even when disturbances are present.

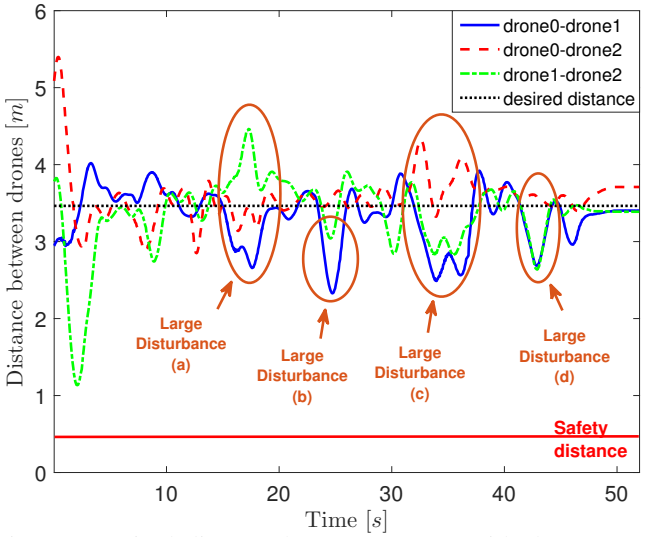


Fig. 13: Desired distance between agents, with the presence of a moving target and disturbances

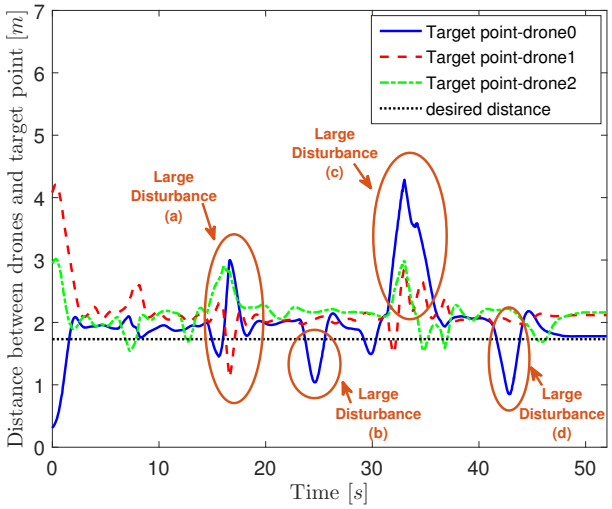


Fig. 14: Distance between agents and moving target point

D. Fleet of three quadrotors with fixed target and defective agent

In this last experiment, the same scenario as in section VI-A was considered, but in this case, one agent was set to act as defective (Figure 15), and aborted its mission and landed while the fleet was moving to the desired reference.

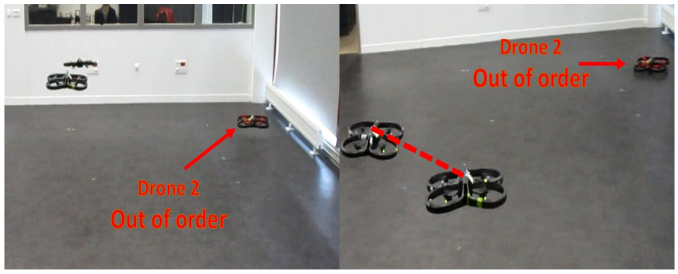


Fig. 15: Defective agent during real-time experiments

The optimization algorithm computes the optimized trajectory for the remaining agents, continuing the mission and arriving to the desired target. Figure 16 represents the fleet position tracking towards the desired references, note the remaining agents arrive to the target while the defective one remains in the ground.

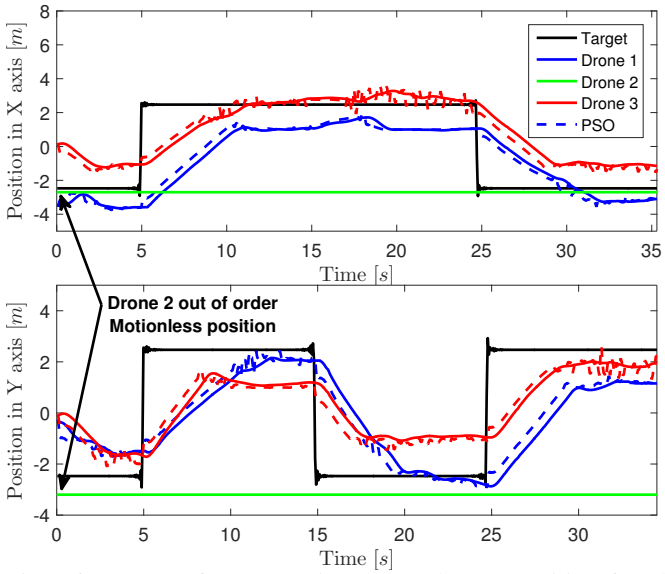


Fig. 16: Target reference, trajectory, and UAV position for all UAVs

All the performed experiments can be watched at the following link: `https://youtu.be/VD3vbGZNhqM`

VII. Conclusion

This paper has presented a distributed path planning approach for controlling multi agent systems consisting on a fleet of autonomous vehicles with real-time validations. The objective has been formulated as an optimization problem based on a Particle Swarm Optimization algorithm. The proposed control approach is partially independent from the general dynamics of the agents, decoupling the translational

movement from the rotational dynamics for computing optimal trajectories, where the stability of each vehicle is ensured locally. The implementation of this approach is then done only in the trajectory generation bloc, which makes it easier to use.

Experimental results on a fleet of real quadrotors have shown unambiguously that the proposed method is effective for training control, target tracking and collisions avoidance. The combination between local control law and distributed path planning approaches allows a good robustness against disturbances. The loss of an agent is directly taken into account in the cost function by decreasing its impact on its neighbors.

REFERENCES

[1] MERHEB, Abdel-Razzak, GAZI, Veysel, et SEZER-UZOL, Nilay. Implementation Studies of Robot Swarm Navigation Using Potential Functions and Panel Methods. IEEE/ASME Transactions on Mechatronics, 2016, vol. 21, no 5, p. 2556-2567.

[2] Miad Moarref and Luis Rodrigues. An optimal control approach to decentralized energy efficient coverage problems. In IFAC World Congress, Cape Town, South Africa, 2014.

[3] PANG, Sze Kim, LI, Jack, et GODSILL, Simon J. Detection and tracking of coordinated groups. IEEE Transactions on Aerospace and Electronic Systems, 2011, vol. 47, no 1, p. 472-502.

[4] WORTHMANN, Karl, KELLETT, Christopher M., BRAUN, Philipp, et al.Distributed and decentralized control of residential energy systems incorporating battery storage. IEEE Transactions on Smart Grid, 2015, vol. 6, no 4, p. 1914-1923.

[5] Pia L Kempker, Andre CM Ran, and Jan H van Schuppen. Controllability and observability of coordinated linear systems. Linear Algebra and its Applications, 437(1):121-167, 2012.

[6] Pia L Kempker, Andre CM Ran, and Jan H Van Schuppen. A formation ying algorithm for autonomous underwater vehicles. In CDC-ECE, pages 1293-1298, Orlando, FL, UASA, 2011.

[7] Nader Meskin and Khashayar Khorasani. Fault detection and isolation: Multi-vehicle unmanned systems. Springer Science & Business Media, 2011.

[8] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. Industrial Informatics, IEEE Transactions on, 9(1):427-438, 2013.

[9] Herbert G Tanner, Ali Jadbabaie, and George J Pappas. Flocking in teams of nonholonomic agents. In Cooperative Control, pages 229-239. Springer, 2005.

[10] Reza Olfati Saber Richard M Murray. Consensus protocols for networks of dynamic agents. In Proceedings of the 2003 American Controls Conference, 2003.

[11] OLFATI-SABER, Reza, FAX, J. Alex, et MURRAY, Richard M. Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE, 2007, vol. 95, no 1, p. 215-233.

[12] REN, Wei, CHAO, Haiyang, BOURGEOUS, William, et al. Experimental validation of consensus algorithms for multivehicle cooperative control. IEEE Transactions on Control Systems Technology, 2008, vol. 16, no 4, p. 745-752.

[13] OLFATI-SABER, Reza. Flocking for multi-agent dynamic systems: Algorithms and theory. IEEE Transactions on automatic control, 2006, vol. 51, no 3, p. 401-420.

[14] S. M. Ahn, H. Choi, S.-Y. Ha, and H. Lee. On collision avoiding initial configurations to Cucker-Smale type ocking models. Communications in Mathematical Sciences, 2012, vol. 10, p. 625-643.

[15] BELKADI, Adel, THEILLIOL, Didier, CIARLETTA, Laurent, Ponsart, J. C. Robust flocking control design for a feet of autonomous agents. In : 3rd Conference on Control and Fault-Tolerant Systems, SysTol 2016, Barcelona, Spain.

[16] Semsar-Kazerooni, E., and Khorasani, K. (2008), Optimal Consensus Algorithms for Cooperative Team of Agents Subject to Partial Information, Automatica, 44, 27662777.

[17] JIANG, Yulian, LIU, Jianchang, TAN, Shubin, et al. Robust consensus algorithm for multi-agent systems with exogenous disturbances under convergence conditions. International Journal of Systems Science, 2014, vol. 45, no 9, p. 1869-1879.

[18] Semsar-Kazerooni, E., and Khorasani, K. (2009), Multi-Agent Team Cooperation: A Game Theory Approach, Automatica, 45, 22052213.

[19] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. Automatica, 36(6):789, 814, 2000.

[20] NEDIC, Angelia et OZDAGLAR, Asuman. Distributed subgradient methods for multi-agent optimization. IEEE Transactions on Automatic Control, 2009, vol. 54, no 1, p. 48-61.

[21] Eduardo F Camacho and Carlos Bordons. Model predictive control. Springer, 2013.

[22] Sorin Olaru, Alexandra Grancharova, and Fernando Lobo Pereira. Developments in Model- Based Optimization and Control: Distributed Control and Industrial Applications, volume 464. Springer, 2015.

[23] BELKADI, A., CIARLETTA, L., et THEILLIOL, D. UAVs team flight training based on a virtual leader: Application to a fleet of Quadrirotors. In : Unmanned Aircraft Systems (ICUAS), 2015 International Conference on. IEEE, 2015. p. 1364-1369.

[24] BELKADI, Adel, CIARLETTA, Laurent, et THEILLIOL, Didier. Particle swarm optimization method for the control of a fleet of Unmanned Aerial Vehicles. In :Journal of Physics: Conference Series. IOP Publishing, 2015. p. 012015.

[25] KENNEDY, James. Particle swarm optimization. In : Encyclopedia of Machine Learning. Springer US, 2010. p. 760-766.

[26] ALFI, Alireza et MODARES, Hamidreza. System iden-

tification and control using adaptive particle swarm optimization. Applied Mathematical Modelling, 2011, vol. 35, no 3, p. 1210-1221.

[27] EBERHART, Russ C., KENNEDY, James, et al. A new optimizer using particle swarm theory. In : Proceedings of the sixth international symposium on micro machine and human science. 1995. p. 39-43.

[28] GUO, Jinchao, WANG, Xinjin, et ZHENG, Xiaowan. Trajectory planning of redundant robot manipulators using QPSO algorithm. In : Intelligent Control and Automation (WCICA), 2010 8th World Congress on. IEEE, 2010. p. 403-407. Jinan, China

[29] CHYAN, Goh Shyh et PONNAMBALAM, S. G. Obstacle avoidance control of redundant robots using variants of particle swarm optimization. Robotics and Computer-Integrated Manufacturing, 2012, vol. 28, no 2, p. 147-153.

[30] BELKADI, Adel, CIARLETTA, Laurent, et THEIL-LIOL, Didier. UAVs fleet control design using distributed particle swarm optimization: A leaderless approach. In : Unmanned Aircraft Systems (ICUAS), 2016 International Conference on. IEEE, 2016. p. 364-371. Arlington, USA.

[31] P. Castillo, L. E. Munoz, P. Garcia, Indoor Navigation Strategies For Aerial Autonomous Systems, ISBN:780128051894, Fev. 2017, Elsevier

[32] GUERRERO, Eusebia, ABAUNZA, Hernan,CASTILLO, Pedro, LOZANO, Rogelio, GARCIA,Carlos. Quadrotor Energy-Based Control Laws: AUnit-Quaternion Approach, Journal of Intelligent and Robotic Systems, December 2017, Vol. 88, Issue 2-4,pp 347-377, Online ISSN 1573-0409

[33] ABAUNZA, Hernan, IBARRA, Efrain, CASTILLO, Pedro, VICTORINO, Alessandro. Quaternion based control for circular UAV trajectory tracking, following aground vehicle: Real-time validation, 20th IFAC World-Congress, Toulouse, France, 2017.

[34] SHI, Yuhui et EBERHART, Russell C. Empirical study of particle swarm optimization. In : Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. IEEE, 1999.

[35] LIAO, Fang, TEO, Rodney, WANG, Jian Liang, et al. Distributed formation and reconfiguration control of VTOL UAVS. IEEE Transactions on Control Systems Technology, 2017, vol. 25, no 1, p. 270-277.

[36] GUERRERO-CASTELLANOS, J. F., VEGA-ALONZO, A., MARCHAND, N., et al. Real-time event-based formation control of a group of VTOL-UAVs. In : Event-Based Control, Communication and Signal Processing (EBCCSP), 2017 3rd International Conference on. IEEE, 2017. p. 1-8.

[37] BETANCOURT-VERA, Julio, CASTILLO, Pedro, LOZANO, Rogelio, and VIDOLOV, Boris, Robust control scheme for trajectory generation and tracking for quadcopters vehicles: Experimental results. In : Unmanned Aircraft Systems (ICUAS), 2018 International Conference on. IEEE.

[38] SANAHUJA, G., et al. Framework Libre Air (FL-AIR), 2018. URL *https://uav.hds.utc.fr/software-flair/*

Adel Belkadi received the Ph.D. degree in automatic control from Lorraine-University, Nancy, France, in 2017. Since September 2017, he is a Temporary Teacher and Researcher (ATER) in Reims-University, IFTS-Charleville-Mezieres. He was born in Bejaia, Algeria, on February 21, 1989. He received the B.S. degree in electronics engineering from university of houari boumediene (USTHB), Algers, in 2010 and MS degree in robotics in 2012. He obtained in 2014 a MS degree in complex system and intelligence control from the University of Paris 12, Creteil, France. His research interests cover control theory and application and multi-agent systems.

Hernan Abaunza was born in Morelos, Mexico, on October 1, 1990. He received the B.S. degree in mechatronic engineering from the Instituto Tecnologico y de Estudios Superiores de Monterrey (ITESM), Mexico, in 2012, the M. Sc. degree in autonomous systems for aerial and submarine navigation from the Centro de Investigacin y de Estudios Avanzados (CINVESTAV), Mexico, in 2014, he is currently working to obtain his Ph.D. degree in Automatic Control from the University of Technology of Compigne, France. His research topics include real-time control applications, quaternion-based modeling and control, and cooperative navigation between unmanned aerial and ground vehicles.

Laurent Ciarletta is ssociate professor in Information Technology and Computer Science, Mines-Nancy, Lorraine University- Ecole des Mines de Nancy (ENSMN). His current research interests include Ambient Computing / Pervasive Computing / Ubiquitous Computing. He focus of his research activities and to UAV ad the Internet of Mobile and Smart Things

Pedro Castillo was born in Morelos, Mexico, on January 8, 1975. He received the B. S. degree in electromechanic engineering from the Instituto Tecnologico de Zacatepec, Morelos, Mexico, in 1997, the M. Sc. degree in electrical engineering from the Centro de Investigacin y de Estudios Avanzados (CINVESTAV), Mexico, in 2000, and the Ph.D. degree in automatic control from the University of Technology of Compigne, France, in 2004. His research topics include real-time control applications, nonlinear dynamics and control, aerospace vehicles, vision, and underactuated mechanical systems

Didier Theilliol received the Ph.D. degree in control engineering from Nancy-University, Vandoeuvre-ls-Nancy, France, in 1993.Since 2004, he is a full Professor with the Research Centre for Automatic Control of Nancy (CRAN), Nancy University, where he coordinates and leads national, European, and international R&D projects in steel industries, wastewater-treatment plant, or aerospace domain. He is the Chair of Intelligent Control and Diagnosis working group where different French and German research teams are involved. His current research interests include model-based fault-diagnosis method synthesis and active fault-tolerant control system design for linear time-invariant, linear parameter-varying, and multilinear systems.