Distributed Integral Control of Multiple UAVs: Precise Flocking and Navigation

Osamah Saif[1,4], Isabelle Fantoni[2,4], Arturo Zavala-Río[3]

Abstract—**Our interest in this work is to perform a precise real-time flocking of multiple UAVs. A consensus-based flocking algorithm that ensures a precise security distance between UAVs is proposed. By using Lyapunov theoretical analysis, we propose a flocking algorithm that ensures the ultimate boundedness of multiple-UAV system solutions. Moreover, this algorithm is enhanced by a distributed integral control that renders the inter-distances between UAVs more precise. Finally, experimental results are provided to prove and show the efficiency of these algorithms.**

I. Introduction

Recently, relatively cheap Unmanned Aerial Vehicles (UAVs), equipped by sensing and actuation capabilities, are emerging rapidly. Moreover, with the miniaturization revolution, these devices are able to perform local decision-making as well as short-range communications. These novelties motivate scientists and researches to raise different questions about the techniques that could be introduced to coordinate and control these devices. One of the biggest challenges is to define local interaction rules between these devices that lead to a global desired behaviour

Autonomous formation control is a field that addresses the control of multiple robots in order to realize geometrical patterns. Inspired from biology, these geometrical patterns could be regular, such as the V-shape seen in migratory birds, lattice or irregular as seen in a flock of birds. In the existing literature, we can distinguish three formation control structures: *Leader-follower*, *Virtual structure* and *Behavioural-based*.

In the *leader-follower* structure, individuals in the formation follow one agent (or airplane) which is designated as a leader. Several theoretical and experimental works using this structure are found in [1] [2] [3] [4] [5] [6] [7] [8] [9] and [10]. One of the main drawbacks of this structure is that the entire formation depends on one agent, so if there is a problem with the leader, the whole formation will be affected.

In the *virtual structure*, each agent in the formation has its own trajectory to follow. The overall trajectories form the desired formation. Examples of experimental works of such control strategy can be found in [11] and [12]. The centralization of control and the absence of interaction between robots are two drawbacks of this structure.

In the *behavioural-based* structure, each agent follows some rules to achieve the formation. Among the first technical work on this structure is the distributed behavioural model introduced by *Reynolds* [13]. He considered that each individual in a formation should follow three rules in order to perform a behavioural-based structure. These rules are: 1) Collision Avoidance; 2) Velocity Matching, and 3) formation Centring The *behavioural-based* is found in the literature, for example in the works of [14] [15] [16] [17] [18] and [19]. The main advantages of the behavioural-based structure are the self-organization aspect of agents in the formation, scalability and their distributed control. However, the most of control laws in these works are designed and implemented for linear second-order models. Moreover, the precision of separation distances between agents is not dealt with. In our work, we use the *behavioural-based* structure.

Using a second-order agent for modelling dynamical systems is a good choice for multiple-agent applications. In fact, models of several systems such as, drones, rovers, vessels etc., that are used for multiple-agent applications, could be simply approximated as a second-order systems. Several studies have used this approach for modelling and designing control algorithms for multiple-agent systems [9] [14] [18] [19] [20] [21].

In [9], the guaranteed-performance consensus problem for second-order multi-agent systems with time delays is solved using state space decomposition, linear matrix inequality and Lyapunov stability theory with some sufficient conditions. A group consensus protocol is proposed in [19] to solve the problem of group consensus for heterogeneous multi-agent systems modeled as discrete-time first- and second-order agents. In [20], the authors propose a non-linear protocol for second-order multi-agent systems to achieve a robust fixed-time consensus tracking using sliding mode control and Lyapunov theory.

However, when developing an algorithm to control these systems, it is important to consider the validation step on a more complex model than second-order in order to test its efficiency. In fact, in most of the previous studies, where the authors use second-order system, they both design and validate their control law on the same system. On the contrary, in our work, we design the control law based on the second-order system, and we validate it on a nonlinear model of UAVs in simulation and on real drones. This disparity between design and validation rises the degree of trust in the application of our control laws compared to the other works.

[1] Scalian Eurogiciel, 417 L'Occitane, CS 77679, 31676 Labège cedex, France. aldakamy@gmail.com

[2] LS2N, UMR CNRS 6004, 1 rue de la Noë, 44321 Nantes, France. isabelle.fantoni@ls2n.fr

[3] Instituto Potosino de Investigacion Cientifica y Tecnologica, Camino a la Presa San Jose 2055, Lomas 4a. Seccion 78216, San Luis Potosi, Mexico. azavala@ipicyt.edu.mx

[4] Former affiliation: Sorbonne universités, Université de Technologie de Compiègne (UTC) - CNRS, UMR 7253 Heudiasyc, 60200, Compiègne, France.

In the works of [14], the author proposes flocking algorithm for multiple agents with double-integrators models. When we applied the proposed algorithm on nonlinear model, for a comparison purpose, we noticed oscillating behaviour of the inter-distances between agents (cf. section VI-C). Moreover, the algorithms were designed and applied on a linear model without considering perturbations.

In [18] and [21], the authors propose a flocking algorithms mainly based on the work of [14], with a pinning approach that considers less informed agents. These pinning algorithms have the same drawbacks as in [14]. Moreover, informed agents selection algorithms are centralized, which means that these algorithms are not scalable.

Our interest in this paper is to design control laws for a real-time autonomous navigation and flocking of multiple quadrotors that ensure a precise security inter-distances between them. We propose algorithms based on behavioural control structure. The contribution of this work is threefold: 1) We propose a flocking algorithm that ensures the ultimate boundedness of the solution of multiple-UAV system. 2) We enhance this algorithm by a distributed integral control term to ensure a precise inter-distance between UAVs. 3)We validate these algorithms in simulation on a nonlinear model of UAVs, in experimentation on real drones and show that they are robust against perturbations.

The paper is organized as follows. Firstly, preliminaries on quadrotors non linear model and graph-based modelling of multiple-UAV are presented in section II. Then, the control architecture of quadrotors in the flocking context is presented in section III. After that, our proposed algorithms are introduced and analysed in sections IV and V. Finally, simulations, experimental results and a conclusion about the paper and future works are given in sections VI, VII and VIII.

II. Preliminaries

A. Multiple-UAV modelling using graph theory

In this section, we use graph theory to describe the topology of a multiple-quadrotor system. A multiple-quadrotor system is represented by an undirected graph $G=(\mathscr{V},E)$, where $\mathscr{V}$ is a set of nodes $\mathscr{V}=\{1,2,...,M\}$, and E is a set of edges $E\subseteq\{(i,j):i,j\in\mathscr{V},i\neq j\}$. Every node represents a quadrotor and edges depict the sensing between quadrotors. An adjacency matrix $\mathscr{A}$ is an $M\times M$ matrix with elements $a_{ij}=1$ if $(i,j)\in E$ and $a_{i,j}=0$ otherwise. For more information about graph theory, the reader can consult [22].

Before working on the dynamics of quadrotors, we need to represent our multiple-quadrotor system in the Euclidean space. Therefore, to every node i in the graph, a position vector $q_i\in\mathbb{R}^f$ is associated, where f is the dimension of the space (example: $f=2,3$). The configuration of all nodes of the graph is defined by the vector $q=col(q_1,...,q_n)\in\mathbb{R}^{fM}$.

A set of spacial neighbours of a quadrotor i is defined by:

$$N_i=\left\{j\in\mathscr{V}:\|q_j-q_i\|<c\right\} \tag{1}$$

where $\|.\|$ is the Euclidean norm, and c is the interaction range. A position-induced graph $G(q)=(\mathscr{V},E(q))$ is called a proximity net and is defined by $\mathscr{V}$ and the set of edges $E(q)=\left\{(i,j)\in\mathscr{V}\times\mathscr{V}:\|q_j-q_i\|<c,j\neq i\right\}$.

The desired formation of multiple quadrotors in a flock can be written as follows:

$$\|q_j-q_i\|=d \quad \forall j\in N_i(q) \tag{2}$$

where d is the desired inter-distance. A proximity net that ensures the objective in (2) is defined as an "*α-Lattice*" [14]. However, implicit inaccuracies give rise to an *α-Lattice* with some edge-length uncertainty. This type of proximity net is called a "*quasi-α-Lattice*" [14], and it is described by the following inequality:

$$-\delta<\|q_j-q_i\|-d<\delta \quad \forall(i,j)\in E(q) \tag{3}$$

where δ is the edge-length uncertainty.

B. Quadrotor modelling

A quadrotor is modelled as a rigid body that evolves in 3D space. Several studies have dealt with the modelling of quadrotors as in [23], [24], [25] and [26]. A quadrotor is composed of four rotors and motors located at corners of a square. We begin by defining the coordinate frames. Let $I=(I_x,I_y,I_z)$ be the global inertial frame, and let $B=(B_x,B_y,B_z)$ be the body-fixed frame.

The nonlinear model of a quadrotor i is given as follows:

$$\dot{\xi}_i=v_i \tag{4a}$$

$$m\ddot{\xi}_i=G+\mathbf{R}_i\,U_i \tag{4b}$$

$$\dot{\eta}_i=W\Omega_i \tag{4c}$$

$$J\dot{\Omega}_i=-\Omega_i\times J\Omega_i+\tau_i \tag{4d}$$

where $\xi_i=[x_i,y_i,z_i]^T$ is the position of the centre of mass of the quadrotor in the I frame, $v_i=[v_{x_i},v_{y_i},v_{z_i}]$ is the vector of linear velocities in the I frame, $\eta_i=[\phi_i,\theta_i,\psi_i]^T$ is the vector of Euler angles named as *Roll*, *Pitch* and *Yaw* respectively. $\Omega=[w_{B_{xi}},w_{B_{y_i}},w_{B_{zi}}]^T$ is the vector of the angular velocities in the B frame, m is the mass of the quadrotor, $G=[0,0,-g]^T$ with g is the gravitational acceleration, $U_i=[0,0,F_i]^T$ is the thrust vector and $\tau_i=[\tau_{\phi_i},\tau_{\theta_i},\tau_{\psi_i}]^T$ is the torque vector.

J is the moment of inertia diagonal matrix where J_x J_y and J_z are diagonal components. W is a transformation matrix between the angular velocities and the derivatives of Euler angles. $\mathbf{R}_i$ is the rotation matrix from the B frame to the I frame. The reader can refer to [26] for detailed expressions of W and $\mathbf{R}_i$.

III. Modelling and flocking of Multiple quadrotors

In this work, we introduce a new architecture to control multiple quadrotors in a flocking context. We separate the control problem of multiple UAVs in two parts. The first part is the control of internal dynamics of each UAV. We mean by internal dynamics, the altitude z_i and the rotational dynamics of each UAV. This part will not be involved in the algorithm of flocking. The second part is the control of the x_i,y_i translation and flocking dynamics of multiple UAVs.

In fact, we specify a fixed desired altitude z_i and heading ψ_i. Moreover, outputs of x_i-y_i translation and flocking controllers are feedforwarded to the inputs of controllers of roll and pitch $\phi_i-\theta_i$ angles. Figure 1 shows the overall control architecture.

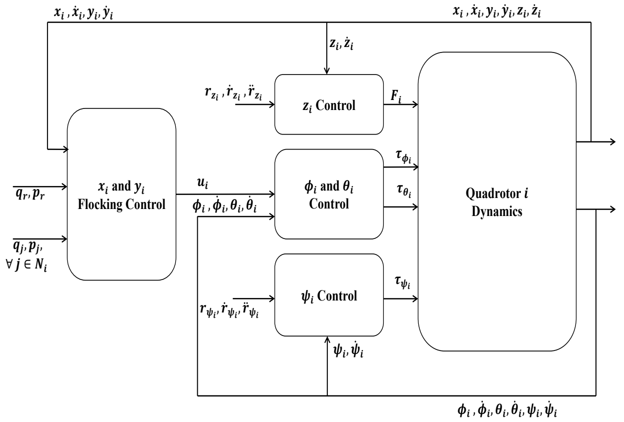
$x_i, \dot{x}_i, y_i, \dot{y}_i$
$x_i, \dot{x}_i, y_i, \dot{y}_i, z_i, \dot{z}_i$
$z_i, \dot{z}_i$
$r_{z_i}, \dot{r}_{z_i}, \ddot{r}_{z_i}$
z_i Control
F_i
τ_{ϕ_i}
τ_{θ_i}
q_r, p_r
x_i and y_i Flocking Control
u_i
$\phi_i, \dot{\phi}_i, \theta_i, \dot{\theta}_i$
ϕ_i and θ_i Control
Quadrotor i Dynamics
$q_j, p_j, \forall j \in N_i$
ψ_i Control
τ_{ψ_i}
$r_{\psi_i}, \dot{r}_{\psi_i}, \ddot{r}_{\psi_i}$
$\psi_i, \dot{\psi}_i$
$\phi_i, \dot{\phi}_i, \theta_i, \dot{\theta}_i, \psi_i, \dot{\psi}_i$

Fig. 1: Control architecture of quadrotor in a flocking perspective. Internal dynamics of $\phi_i - \theta_i$ are controlled separately from the $x_i - y_i$ translation and flocking dynamics. The z_i and ψ_i controller track a user-defined references r_{z_i}, r_{ψ_i} and their derivatives

A. Control of quadrotor internal dynamics

Before starting the description of the control strategy of a UAV in a flocking perspective, we begin by linearising the nonlinear model (4) about the origin ($\xi_i = \mathbf{0}, \dot{\xi}_i = \mathbf{0}, \eta_i = \mathbf{0}, \dot{\eta}_i = \mathbf{0}, F_i = 0, \tau_i = 0$).

The result of linearisation about the origin is given as follows:

$$\begin{array}{llll}
\dot{x}_i = v_{x_i} & \text{(5a)} & \dot{\phi}_i = w_{B_{xi}} & \text{(6a)} \\
\dot{y}_i = v_{y_i} & \text{(5b)} & \dot{\theta}_i = w_{B_{y_i}} & \text{(6b)} \\
\dot{z}_i = v_{z_i} & \text{(5c)} & \dot{\psi}_i = w_{B_{zi}} & \text{(6c)} \\
\dot{v}_{x_i} = (1/m)\theta_i & \text{(5d)} & \dot{w}_{B_{xi}} = (1/J_x)\tau_{\phi_i} & \text{(6d)} \\
\dot{v}_{y_i} = -(1/m)\phi_i & \text{(5e)} & \dot{w}_{B_{y_i}} = (1/J_y)\tau_{\theta_i} & \text{(6e)} \\
\dot{v}_{z_i} = F_i & \text{(5f)} & \dot{w}_{B_{zi}} = (1/J_z)\tau_{\psi_i} & \text{(6f)}
\end{array}$$

Equations (5a) through (5f) represent the translational dynamics of the UAV and equations (6a)-(6f) represent the rotational dynamics.

The control of the UAV will be as follows. Firstly, we use PID controllers to control the z_i dynamics (equations (5c) and (5f)) and the ψ_i dynamics (equations (6c) and (6f)).

Secondly, the rotational dynamics of the angles $\phi_i - \theta_i$, in equations (6a), (6b), (6d) and (6e)), will be controlled using nested saturation controllers [27], [28], [29] and [30]. Here, we suppose that these controllers will ensure that the $\phi_i - \theta_i$ rotational dynamics follow $\phi_{id} - \theta_{id}$ desired angles.

B. Translational dynamics

To control the $x_i - y_i$ dynamics, in equations (5a), (5b), (5d), (5e), we use an approach similar to the backstepping technique which is widely used in the control of quadrotors [26]. In fact, we consider the ϕ_{id} and θ_{id} as the control inputs of the $x_i - y_i$ dynamics and we can rewrite them as follows:

$$\begin{aligned} \dot{q}_i &= p_i \\ \dot{p}_i &= u_i + \delta_i \end{aligned} \tag{7}$$

with $q_i = [x_i \ y_i]^T$, $p_i = [v_{x_i} \ v_{y_i}]^T$, and $u_i = [\frac{1}{m}\theta_{id} \ \frac{-1}{m}\phi_{id}]^T$. The term $\delta_i \equiv \delta_i(q_i, p_i) \in \mathbb{R}^f$ is a perturbation vector that represents both external disturbances, for instance, wind and uncertainties in the tracking performance of ϕ_{id} and θ_{id} in the rotational dynamics of $\phi_i - \theta_i$.

Assumption 1. *The perturbation term δ_i is an unknown function, but it has a known upper bound such that $\|\delta_i\| \leq \kappa$ and $\kappa > 0$.*

This assumption is realistic, since in practical and real-time problems we cannot estimate perturbation signals, and their effect on the system dynamics is generally limited.

C. Flocking of multiple quadrotors

Flocking is a collective motion of multiple agents toward a common objective. In this article, in order to achieve the flocking behaviour, the UAVs move toward a unique desired trajectory with a time varying position q_r and velocity p_r. In fact, knowing the desired destination and the current state of the UAV, q_r and p_r are generated on each UAV using a trajectory generator. Hence, we define a moving frame that has its origin in (q_r) and we assume here that the velocity is bounded such that $\|p_r\| \leq p_{r_{max}}$. Therefore, positions and velocities of UAVs in the moving frame are represented as follows:

$$q_i^r = q_i - q_r; \quad p_i^r = p_i - p_r \tag{8}$$

It is easy to see that the relative positions and velocities are the same in both the inertial and the moving frames. That is:

$$\begin{aligned} q_j^r - q_i^r &= q_j - q_i \\ p_j^r - p_i^r &= p_j - p_i \end{aligned}$$

Therefore, the agent model in (7) will be written as:

$$\begin{aligned} \dot{q}_i^r &= p_i^r \\ \dot{p}_i^r &= u_i^r + \delta_i^r \end{aligned} \tag{9}$$

with $u_i^r = u_i(q_i^r, p_i^r) - \dot{p}_r$, and $\delta_i^r = \delta_i$.

The configuration of all UAVs in the new reference frame is $q^r = col(q_1^r, \ldots.., q_M^r) \in \mathbb{R}^{fM}$. Therefore we have, $p^r = col(p_1^r, \ldots.., p_M^r) \in \mathbb{R}^{fM}$, $u^r = col(u_1^r, \ldots.., u_M^r) \in \mathbb{R}^{fM}$, and $\delta^r = col(\delta_1^r, \ldots.., \delta_M^r) \in \mathbb{R}^{fM}$. Therefore, we get the following collective dynamics of multiple UAVs :

$$\begin{aligned} \dot{q}^r = & \quad p^r \\ \dot{p}^r = & \quad u^r + \delta^r \end{aligned} \tag{10}$$

The control of this system is studied in the following section.

IV. Flocking control with tuning gains

In this section, we discuss the control of the multiple-UAVs translational system in (10) from a flocking perspective. We begin by introducing the basic principles of controlling a multi-agent system as developed in [14].

Let $A(q^r) = [a_{ij}(q^r)]$ be a spacial adjacency matrix where $a_{ij}(q^r)$ are its elements given as follows:

$$a_{ij}(q^r) = \begin{cases} 0 & \text{if } i = j \\ \rho_h(\|q_j^r - q_i^r\|_\sigma / \|c\|_\sigma) & \text{if } j \neq i \end{cases} \tag{11}$$

with $\rho_h : \mathbb{R}^+ \rightarrow [0,1]$ is a bump function defined as:

$$\rho_h(z) = \begin{cases} 1 & \text{if } z \in [0, h) \\ \frac{1}{2}\left[1 + \cos\left(\pi \frac{z-h}{1-h}\right)\right] & \text{if } z \in [h, 1] \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

and $h \in (0,1)$. Moreover, $\|.\|_\sigma$ is a "σ-norm" $\mathbb{R}^n \rightarrow \mathbb{R}^+$ of a vector $z \in \mathbb{R}^n$, defined as:

$$\|z\|_\sigma = \frac{1}{\varepsilon}\left[\sqrt{1+\varepsilon\|z\|^2}-1\right] \tag{13}$$

where $\varepsilon > 0$ and $\mathbb{R}^+$ is the set of non-negative real numbers. The gradient of σ-norm is defined by:

$$\sigma_\varepsilon(z) \triangleq \nabla_z\|z\|_\sigma = \frac{z}{\sqrt{1+\varepsilon\|z\|^2}} = \frac{z}{1+\varepsilon\|z\|_\sigma} \tag{14}$$

In fact, σ-norm is not a norm but its importance is that it is differentiable everywhere, unlike the Euclidean norm that is not differentiable at $z=0$.

A smooth collective potential function is used to design the flocking algorithm of multiple quadrotors. This function is given as follows:

$$V(q^r) = \frac{1}{2}\sum_i\sum_{j\neq i}\Psi_\alpha(\|q_j^r - q_i^r\|_\sigma) \tag{15}$$

where

$$\Psi_\alpha(z) = \int_{d_\alpha}^{z}\Phi_\alpha(s)ds \tag{16}$$

Φ_α is defined by:

$$\begin{array}{l}\Phi_\alpha(z) = \rho_h(z/c_\alpha)\Phi(z-d_\alpha)\\ \Phi(z) = \frac{1}{2}[(a+b)\sigma_1(z+e)+(a-b)]\\ \sigma_1(z) = z/\sqrt{1+z^2}\end{array} \tag{17}$$

The function $\Phi(z)$ is uneven and sigmoidal, with $0 < a \le b$ and $e = |a-b|/\sqrt{4ab}$ that ensures $\Phi(0) = 0$.

It follows from the above formulas that $\Psi_\alpha(z)$ is a smooth pairwise repulsive/attractive potential function. It has a global minimum at $z = d_\alpha = \|d\|_\sigma$, and it has a finite cut-off at $c_\alpha = \|c\|_\sigma$. The finite cut-off feature of this function is a fundamental source of scalability of the flocking algorithm [14]. Moreover, every local minimum of $V(q^r)$ is an α-lattice. Figure 2 shows the collective potential function in equation (15) for inter-distances between two UAVs varying between 0 and 3 meters.

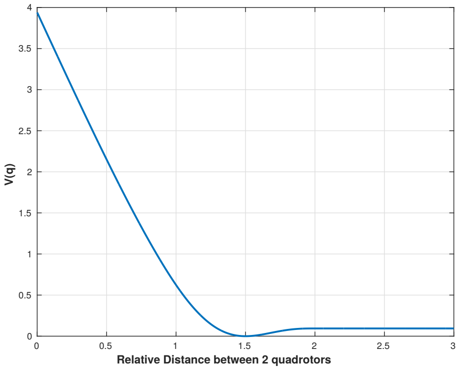

Fig. 2: $V(q^r)$ of two UAVs for inter-distances varying between 0 and 3 meters

The gradient of the potential function $V(q^r)$ is given by:

$$\nabla_{q^r}V(q^r) = col(\ldots, -\sum_{j\in N_i}\Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\mathbf{n}_{ij}, \ldots) \tag{18}$$

with

$$\mathbf{n}_{ij} = \sigma_\varepsilon(q_j^r - q_i^r) \tag{19}$$

as in equation (14).

Assumption 2. *The UAVs initial positions are in the set*

$$\Omega_c = \left\{q^r : d_0 \le \|q_j^r - q_j^r\| \le c_r \;\&\; \|q^r\| \le q_{max} \forall i \in \mathcal{V}\right\}$$

with $c \le c_r \le \infty$, and d_0 is a non collision distance between UAVs, with c is the interaction range between UAVs. Moreover, the initial velocities of UAVs are bounded such that $\|p_i^r\| \le p_{max} \forall i \in \mathcal{V}$.

Assumption 2 indicates that the initial conditions of UAVs are not starting in a collision state.

Lemma 1. *The gradient of the potential function is bounded by*

$$\nabla V_{max} = \frac{b}{\sqrt{\varepsilon}}\sqrt{\sum_{i\in\mathcal{V}}|N_i|^2} \quad \forall q^r \in \mathbb{R}^{fM}.$$

Proof. Taking an element $\nabla_{q^r}V(q^r)_i$ from (18), we have:

$$\nabla_{q^r}V(q^r)_i = \sum_{j\in N_i}\Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\mathbf{n}_{ij} \tag{20}$$

Then,

$$\|\nabla_{q^r}V(q^r)_i\| = \|\sum_{j\in N_i}\Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\mathbf{n}_{ij}\|$$

Therefore, from the expressions of Φ_α, ρ_h and $\mathbf{n}_{ij}$ in (17), (12) and (19) respectively, we can find that the norm of the sigmoidal part of Φ_α is bounded by its maximum b, ρ_h is bounded by 1 and the norm of $\mathbf{n}_{ij}$ is bounded by $\frac{1}{\sqrt{\varepsilon}}$, which leads to:

$$\|\nabla_{q^r}V(q^r)_i\| \le \sum_{j\in N_i}\|\Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\|\|\mathbf{n}_{ij}\| \le \frac{b}{\sqrt{\varepsilon}}|N_i|$$

with $|N_i|$ is the cardinality of the neighbouring set.

Now for all the UAVs in the flock we have:

$$\|\nabla_{q^r}V(q^r)\|^2 = \sum_{i\in\mathcal{V}}\|\nabla_q V(q^r)_i\|^2$$

Therefore:

$$\|\nabla_{q^r}V(q^r)\| \le \nabla V_{max} = \frac{b}{\sqrt{\varepsilon}}\sqrt{\sum_{i\in\mathcal{V}}|N_i|^2} \tag{21}$$

□

Lemma 2. *Suppose the assumption 2 is satisfied. Then, the potential function $V(q^r)$ in (15) is bounded.*

Proof. Let us study $V(q^r)$ at the extremities of Ω_c. For $d_\alpha \le \|q_j^r - q_j^r\| \le c_r$ we have:

$$\Psi_\alpha(c_r) = \int_{d_\alpha}^{c_r}\Phi_\alpha(s)ds = \int_{d_\alpha}^{c}\Phi_\alpha(s)ds + \underbrace{\int_{c}^{c_r}\Phi_\alpha(s)ds}_{=0} \tag{22}$$

The last term is equal to zero since $\Phi_\alpha(s) = 0$ between c and c_r. Taking the maximum of $\Phi_\alpha(s)$ in the interval $d_\alpha \le \|q_j^r - q_j^r\| \le c$, then we can write (22) as follows:

$$\Psi_\alpha(c_r) \le \int_{d_\alpha}^{c}\Phi_{max}\,ds = a\,(c - d_\alpha) \tag{23}$$

On the other hand, for $d_0 \leq \|q_j^r - q_j^r\| \leq d_\alpha$ we have:

$$\Psi_\alpha(d_0) = \int_{d_\alpha}^{d_0} \Phi_\alpha(s)ds \tag{24}$$

Since $\Phi_\alpha(s)$ is uneven function, then:

$$\Psi_\alpha(d_0) = \int_{d_0}^{d_\alpha} |\Phi_\alpha(s)|ds \tag{25}$$

Therefore, taking the absolute maximum of Φ_α in the interval, we get:

$$\Psi_\alpha(d_0) \leq \int_{d_0}^{d_\alpha} b\, ds = b\,(d_\alpha - d_0) \tag{26}$$

From equations (23) and (26) we have:

$$\Psi_\alpha(z) \leq max(a\,(c - d_\alpha), b\,(d_\alpha - d_0)) = \Psi_{max} \tag{27}$$

For $a = b$ and $(d_\alpha - d_0) > (c - d_\alpha)$, we get $\Psi_{max} = a\,(d_\alpha - d_0)$.

Therefore, the potential function is bounded by V_{max} as follows:

$$V(q^r) \leq V_{max} = \Psi_{max} \sum_{i \in \mathscr{V}} N_i \tag{28}$$

□

In this section, we introduce our first flocking control law which is given in the following equation:

$$\begin{aligned} u_i^r = \sum_{j \in N_i} \Big[& K_p \Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\mathbf{n}_{ij} + K_p^{'} a_{ij}(q^r)(q_j^r - q_i^r) \\ & + K_d a_{ij}(q^r)(p_j^r - p_i^r)\Big] + f_i^\gamma(q_i^r, p_i^r) \end{aligned} \tag{29}$$

where $\mathbf{n}_{ij} = \sigma_\varepsilon(q_j^r - q_i^r)$ as in equation (14), and $K_p, K_p^{'}, K_d > 0$ are constant scalar tuning gains and their values depend on the quadrotor device. K_p and K_d are user defined. They give a relative freedom to the user to apply the control law on different quadrotor devices. Moreover, unlike the works in [14] [17], the additional gains will compensate for uncertainties of nonlinear model. $f_i^\gamma(q_i^r, p_i^r) = -c_1 q_i^r - c_2 p_i^r$, $c_1, c_2 > 0$ is a navigational function that depends on aboard generated reference trajectory in each UAV, see equation (8).

This control law is composed of four terms that ensures the rules of *Reynolds* in the behavioural-based control. The first two terms ensure the inter-distance regulation and the attraction/repulsion forces between neighbouring UAVs which leads to a collision-free flocking. The third term is the velocity consensus/matching term, which is analogous to a derivative action in a conventional PD control law. The last term $f_i^\gamma(.)$ is the navigational or the translational feedback control, which leads the whole flock to track a predefined objective trajectory known by every UAV in the flock, which ensures a fragmentation-free flocking and the formation centring

Lemma 3. *[14](Spatial-Order): Every local minimum of $V(q^r)$ is an α-lattice and vice versa.*

An important matrix in graph theory and flocking control is the graph Laplacian. In our graph $G = (\mathscr{V}, E)$ with M UAVs, the graph Laplacian $M \times M$ matrix is defined as follows:

$$L = D(\mathscr{A}(q^r)) - \mathscr{A}(q^r) \tag{30}$$

with $\mathscr{A}(q^r)$ being the spacial adjacency matrix and $D(\mathscr{A}(q^r))$ being a diagonal matrix called, degree matrix of the graph G, and $\sum_{j=1}^{M} a_{ij}(q^r)$ being its diagonal elements. This matrix is positive semidefinite.

The Laplacian matrix satisfies the following sum-of-Square (SOS) property [31] and [14]:

$$S^T L S = \frac{1}{2} \sum_{(i,j) \in E} a_{ij}(S_j - S_i)^2 \;, S \in \mathbb{R}^M \tag{31}$$

An *f-dimensional graph Laplacian* is defined as follows:

$$\hat{L} = L \otimes I_f \tag{32}$$

where $\otimes$ is the Kronecker product and I_f being an $f \times f$ identity matrix. The property in (31) could then be written for $\hat{L}$ as the following:

$$\mathbf{S}^T \hat{L} \mathbf{S} = \frac{1}{2} \sum_{(i,j) \in E} a_{ij} \|\mathbf{S}_j - \mathbf{S}_i\|^2 \;, \mathbf{S} \in \mathbb{R}^{fM} \tag{33}$$

where $\mathbf{S} = col(S_1, S_2, ..., S_M)$ and $S_i \in \mathbb{R}^f$.

Therefore, the multiple-UAVs translational system in (10) that applies the control law in (29) is written as follows

$$\begin{aligned} \dot{q}^r = \;& p^r \\ \dot{p}^r = \;& -K_p \nabla_{q^r} V(q^r) - K_p^{'} \hat{L} q^r - K_d \hat{L} p^r \\ & -c_1 q^r - c_2 p^r + \delta^r \end{aligned} \tag{34}$$

Proposition 1. *Consider the multiple-UAV dynamical system in (34) that applies the distributed control law in (29). Suppose that assumptions (2) and (1) are satisfied. Then the solutions of the multiple-UAV dynamical system (34) are ultimately bounded.*

Proof. In the beginning, we consider the collective system in (34) without perturbations. Taking inspiration from the backstepping technique [32], we distinguish two subsystems in cascade in (34). In the first subsystem:

$$\dot{q}^r = p^r \tag{35}$$

we consider $p^r = U_1$ as a virtual input. Taking the following Lyapunov candidate function:

$$V_1 = \frac{1}{2}\|q^r\|^2 \tag{36}$$

choosing $U_1 = -K_1 q^r$, the derivative of the Lyapunov function is:

$$\dot{V}_1 = -K_1 \|q^r\|^2 \;\leq 0 \tag{37}$$

Then, the origin of the subsystem (35) is asymptotically stable.

Now, considering the second subsystem, without taking in consideration the expression of the control law u^r and the perturbations:

$$\dot{p}^r = u^r \tag{38}$$

Taking the change of variables

$$\mathscr{X} = p^r - U_1 \tag{39}$$

and applying it to the two subsystems in (35) and (38), then we get the following system:

$$\begin{aligned}\dot{q}^r &= -K_1 q^r + \mathscr{Z}\\ \dot{\mathscr{Z}} &= u^r - K_1^2 q^r + K_1\mathscr{Z}\end{aligned}\tag{40}$$

Now, let $H(q^r,\mathscr{Z})$ be a Lyapunov function defined as follows:

$$H(q^r,\mathscr{Z}) = \frac{1}{2}\|q^r\|^2 + \frac{1}{2}\|\mathscr{Z}\|^2 + K_p V(q^r)\tag{41}$$

The time derivative of H gives:

$$\begin{aligned}\dot{H} = \;& -K_1\|q^r\|^2 + q^{rT}\mathscr{Z} + \mathscr{Z}^T u^r\\ & -K_1^2\mathscr{Z}^T q^r + K_1\|\mathscr{Z}\|^2\\ & -K_pK_1\nabla_{q^r}V(q^r)q^r + K_p\nabla_{q^r}V(q^r)\mathscr{Z}\end{aligned}\tag{42}$$

choosing u^r such that:

$$\begin{aligned}u^r = \;& -K_p\nabla_{q^r}V(q^r) - K_d\hat{L}\mathscr{Z}\\ & -(1-K_1^2)q^r - (K_1+K_2)\mathscr{Z}\end{aligned}\tag{43}$$

we get :

$$\dot{H} = -K_1\|q^r\|^2 - K_2\|\mathscr{Z}\|^2 - K_pK_1\nabla_{q^r}V(q^r)q^r - K_d\,\mathscr{Z}^T\hat{L}\mathscr{Z}\tag{44}$$

Now, in the presence of perturbations, that is, $u^r \equiv u^r + \delta^r$, we get:

$$\begin{aligned}\dot{H} = \;& -K_1\|q^r\|^2 - K_2\|\mathscr{Z}\|^2 - K_pK_1\nabla_{q^r}V(q^r)q^r\\ & -K_d\,\mathscr{Z}^T\hat{L}\mathscr{Z} + \mathscr{Z}^T\delta^r\end{aligned}\tag{45}$$

since $\hat{L}$ is positive semi-definite, and using Lemma 1 then:

$$\begin{aligned}\dot{H} \leq \;& -K_1\|q^r\|^2 - K_2\|\mathscr{Z}\|^2\\ & +K_pK_1\nabla V_{max}\|q^r\| + \|\mathscr{Z}\|\kappa\end{aligned}\tag{46}$$

with κ is the upper bound of the perturbation vector δ. Then, for $0 < b_1 < 1$ and $0 < h_1 < 1$ we can write:

$$\begin{aligned}\dot{H} \leq \;& -(1-b_1)K_1\|q^r\|^2 - b_1K_1\|q^r\|^2\\ & -(1-h_1)K_2\|\mathscr{Z}\|^2 - h_1K_2\|\mathscr{Z}\|^2\\ & +K_pK_1\nabla V_{max}\|q^r\| + \|\mathscr{Z}\|\kappa\end{aligned}\tag{47}$$

Therefore,

$$\begin{aligned}\dot{H} \leq \;& -(1-b_1)\beta\|q^r\|^2 - (1-h_1)K_2\|\mathscr{Z}\|^2\\ & \forall\|q^r\| \geq \frac{K_p\nabla V_{max}}{b_1},\ \forall\|\mathscr{Z}\| \geq \frac{\kappa}{h_1K_2}\end{aligned}\tag{48}$$

In order to prove the ultimate boundedness of the solutions of (34) and taking inspiration from boundedness and ultimate boundedness section in [32], we need to formulate (48) and add some inequalities as follows:

From (41) we can find:

$$H(q^r,\mathscr{Z}) \geq \frac{1}{2}\|q^r\|^2 + \frac{1}{2}\|\mathscr{Z}\|^2\tag{49}$$

then, from Lemma 2 and equation (28) we have:

$$H(q^r,\mathscr{Z}) \leq \frac{1}{2}\|q^r\|^2 + \frac{1}{2}\|\mathscr{Z}\|^2 + V_{max}\tag{50}$$

Therefore, using (49) and (50) we can get:

$$c_3\|\Lambda\|^2 \leq H(q^r,\mathscr{Z}) \leq c_4\|\Lambda\|^2 + V_{max}\tag{51}$$

with $\Lambda = [q^{rT}\ \mathscr{Z}^T]^T$, c_3 and c_4 are positive constants, and $c_3\|\Lambda\|^2$, $c_4\|\Lambda\|^2$ are class $\mathscr{K}$ functions.

Considering assumption 2, the condition in (48) can be rewritten as follows:

$$\begin{aligned}\dot{H} \leq \;& -(1-b_1)\beta\|q^r\|^2 - (1-h_1)K_2\|\mathscr{Z}\|^2\\ & \forall\,\Lambda_{min} \leq \|\Lambda\| \leq \Lambda_{max}\end{aligned}\tag{52}$$

with

$$\Lambda_{min} = \sqrt{\left(\frac{K_p\nabla V_{max}}{b_1}\right)^2 + \left(\frac{\kappa}{h_1K_2}\right)^2}$$

and

$$\Lambda_{max} = \sqrt{q_{max}^2 + \mathscr{Z}_{max}^2}$$

with $\mathscr{Z} = K_1q_{max} + p_{max}$.

The proof of ultimate boundedness of the system in (34) is different from the classical one found in [32]. In fact, the proof in [32] needs the Lyapunov function to be upper and lower bounded by class $\mathscr{K}$ functions. However, this condition is deformed here in (51) by the inclusion of V_{max}. Therefore, in this article, we reformulate the proof as follows.

For the multi-UAVs system in (34) to be ultimately bounded, we need to find a set $\Omega_h = \{\Lambda \mid h_{min} \leq H(q^r,\mathscr{Z}) \leq h_{max}\}$, such that, $\Omega_h \subset \Omega_\Lambda = \{\Lambda \mid \Lambda_{min} \leq \|\Lambda\| \leq \Lambda_{max}\}$, with $h_{min}, h_{max} > 0$. Therefore, $\forall\Lambda$ starting in Ω_h the solution of (34) will behave as if the origin is asymptotically stable until entering the set $\Omega_{h_{min}} = \{\Lambda \mid H(q^r,\mathscr{Z}) \leq h_{min}\}$. To find h_{min} and h_{max}, we proceed as follows.

In one hand, from the left inequality of (51) we have:

$$H \leq h_{max} \implies c_3\|\Lambda\|^2 \leq h_{max} \iff \|\Lambda\| \leq \sqrt{\frac{h_{max}}{c_3}}$$

Taking $h_{max} = c_3\Lambda_{max}^2$ implies that
$\|\Lambda\| \leq \Lambda_{max}$ and therefore, $\Omega_{h_{max}} = \{\Lambda \mid H(q^r,\mathscr{Z}) \leq h_{max}\} \subset \Omega_{\Lambda_{max}} = \{\Lambda \mid \|\Lambda\| \leq \Lambda_{max}\}$.

In the other hand, from the right inequality of (51) we have:

$$\forall\Lambda \leq \Lambda_{min} \implies H \leq c_4\Lambda_{min}^2 + V_{max}$$

by taking $h_{min} = c_4\Lambda_{min}^2 + V_{max}$, therefore, we get $\Omega_{\Lambda_{min}} = \{\Lambda \mid \|\Lambda\| \leq \Lambda_{min}\} \subset \Omega_{h_{min}}$, which proves the ultimate boundedness of the solutions of multi-UAVs system in (34) $\forall\Lambda \in \Omega_h$.

Now, to get the ultimate bound on Λ we have from the left inequality of (51):

$$\forall H \leq h_{min} \implies c_3\|\Lambda\|^2 \leq h_{min} \iff \|\Lambda\| \leq \sqrt{\frac{h_{min}}{c_3}}$$

Therefore, we can take $\Lambda_b = \sqrt{\frac{h_{min}}{c_3}} = \sqrt{\frac{c_4\Lambda_{min}^2 + V_{max}}{c_3}}$ as the ultimate bound.

□

Remark 1. *Using the expression of $\mathscr{Z}$ in (39) and replacing it in the control law in (43) we get the control law used in the collective dynamics (34) with:*

$$\begin{aligned} K_p^{'} &= K_d K_1 \\ c_1 &= K_1 K_2 + 1 \\ c_2 &= K_1 + K_2 \end{aligned} \tag{53}$$

V. Flocking with distributed integral control

In the previous section, we proposed our first control law. This control law was tested in a real-time experimental set-up and showed good results, as it will be shown in section VII. However, real-time applications always suffer from perturbations that could not be eliminated easily, and need more effective control laws. Perturbations are generally caused by the wind flown from the rotors of quadrotors and by non modelled dynamics. In real-time experiments, we noticed steady-state errors in the distances between quadrotors. Therefore, in this section, we present an alternative version of (29), intended to eliminate the steady-state errors.

In control theory, one of the ways used to eliminate steady-state errors is to add an integral action to the control law [32]. For ordinary systems, we start by augmenting the system equations with a state that represents the integral of regulation error. However, for our multi-UAVs system, we deal with the regulation of inter-distances with several neighbouring UAVs, which renders the introduction of integral actions challenging.

To overcome this steady-state error problem, we define the distributed regulation error of a single UAV as follows:

$$e_i^r = \sum_{j \in N_i} \Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\, \mathbf{n}_{ij} \tag{54}$$

Then, we augment the single UAV dynamics in (9) by $\dot{\vartheta}^r = e_i$. Therefore, the overall augmented system will be written as follows:

$$\begin{aligned} \dot{\vartheta}^r &= e_i^r \\ \dot{q^r}_i &= p_i^r \\ \dot{p^r}_i &= u_i^r \end{aligned} \tag{55}$$

Therefore, our control law will be written as follows:

$$\begin{aligned} u_i^r = \sum_{j \in N_i} &\Big[K_p \Phi_\alpha(\|q_j^r - q_i^r\|_\sigma)\mathbf{n}_{ij} + K_p^{'} a_{ij}(q)(q_j - q_i) \\ &+ K_d a_{ij}(q^r)(p_j^r - p_i^r) \Big] + K_i \int e_i^r \,\mathrm{d}t \\ &+ f_i^\gamma(q_i^r, p_i^r) \end{aligned} \tag{56}$$

where K_i is the integral-action gain, which is tuned by the user. This alternative control law shows good results in real-time experiments, which complements our first control law in (29). In fact, the steady-state errors are efficiently reduced, and then the solutions of multiple-UAVs system converge to an "*α-Lattice*".

VI. Simulation Results

A. Simulator of Multiple UAVs

As a part of ROBOTEX project, Heudiasyc laboratory is equipped by a fleet of UAVs in order to carry out scientific researches on flight formation control. However, flight formation control could be risky, therefore, the laboratory developed a simulator of fleet of UAVs. The goal of this simulator is to run, on a computer, an identical programme to that one used in the real UAVs, to perform all the test steps safely.

For this purpose, the PC is run under Linux as in the real UAVs. In the simulator, virtual sensors and actuators are connected to a discrete nonlinear model of UAV as in (4). As a result, all UAVs' states are calculated at each instant of time. Each UAV in the fleet simulator is an independent computer process. Moreover, UAVs evolve in a 3-D virtual environment, thanks to Irrlicht engine. The program in the simulator is connected to a Graphical User Interface (GUI) base-station control program. The base station records and draws measurements. Moreover, it is used to start and end simulations and to set the parameters of UAVs and control laws.

B. Flocking control with tuning gains

Figure 3 (**left**) shows an aggregation and circular navigation of 6 UAVs using the flocking control with tuning gains. The UAVs start in their initial positions, marked by diamonds, and converge firstly toward the origin. When the formation is stable, a new phase is activated and the UAVs start to track a circular trajectory, of $10m$ radius, whilst maintaining the formation. Finally, after a certain time, the UAVs quit the circular trajectory and return back to aggregate around the origin by forming a pentagonal quasi-α lattice pattern, with one UAV at the centre.

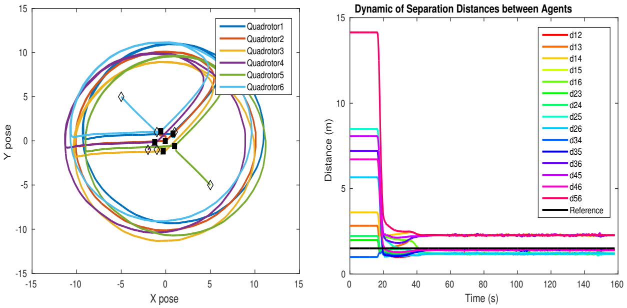

*Fig. 3: **Simulations:** (**left**) Trajectories of 6 UAVs in aggregation and circular navigation scenario using tuning gains control strategy in equation (29). (**right**) Inter-distances between 6 UAVs in aggregation and circular navigation scenario using tuning gains control strategy in equation (29)*

The formation of a quasi-α lattice pattern is confirmed in figure 3 (**right**). The distances between neighbouring UAVs approach the desired inter-distances value $1.5m$. We can notice that the inter-distances do not reach completely the desired reference which means that this control strategy suffers from a steady-state error drawback. However, the scalability of this control law could be seen in figure 3 (**right**). For example, the distance between UAV 5 and 6 does not converge to $1.5m$, since it is greater than the interaction range $2m$. This means that, using this control law, we can add as much UAVs as we need to the fleet and the computational power of each UAV will not be affected, since each UAV interacts with a limited number of UAVs in its neighbouring range.

C. Comparison with other works

In this section, for a comparison purpose, we simulate the flocking algorithm proposed in [14] and reused in [18] and [21].

Figure 4 shows the simulation result of applying the flocking control law, named as *Algorithm 2* in [14], on four quadrotors. The algorithm in [14] is given as follows:

$$
\begin{aligned}
u_i &= \sum_{j\in N_i} [\Phi_\alpha(\|q_j - q_i\|_\sigma)\mathbf{n}_{ij} + a_{ij}(q)(p_j - p_i)] \\
&+ f_i^\gamma(q_i, p_i, q_r, p_r)
\end{aligned} \qquad (57)
$$

This figure shows the distances between quadrotors over the time, and the legend d$_{ij}$ depicts the distance between quadrotor i and j. The intended scenario is as follows: quadrotors start in their initial positions and move toward a predefined q_r position. For the numerical application, we set $a = b = 5$, the desired distance between neighbours is $d = 1$, $c = 1.2d$, $\varepsilon = 0.1$ and $h = 0.2$. In this figure, the distance between quadrotors 1 and 3 does not converge to the desired value because this distance exceeds the interaction range $c = 1.2d$. Moreover, We notice the oscillating behaviour of the quadrotors during flight. This is caused by the absence of tuning gains in the gradient-based and the velocity consensus terms in (57). Since this results are risky, we avoided experimental works with this control law.

Comparing the result of our algorithm in figure 3 with that of figure 4, we can notice clearly that the behaviour of our algorithm is more stable. This result shows that our algorithm is more efficient when the controlled system is nonlinear.

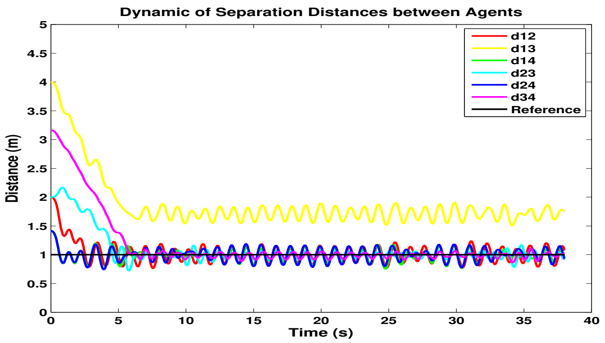

Fig. 4: Inter-distances between 4 quadrotors in the simulator by using Olfati-Saber algorithm (57). Once the control law of flocking is launched in each quadrotors, the interactions start and the distances converge with oscillation

D. Flocking with distributed integral control

In the previous simulation, the main drawback was the ”steady-state errors” in the inter-distances between neighbouring UAVs. In this part, we present the same previous scenario with our distributed integral control. We keep the same desired inter-distance value $1.5m$ and the same interaction range $2m$ to ensure the scalability of this control law. As we will see in the figures, the distributed integral control law overcomes the ”steady-state errors” problem and ensures, almost, an $\alpha-$lattice pattern together in an aggregation and in a navigation scenarios.

Figure 5 (**left**) shows an aggregation and a circular navigation scenario of 6 UAVs. As seen before, the UAVs perform an aggregation toward the origin and form an $\alpha-$lattice pattern. Then, they follow a circular trajectory while keeping the formation. Finally, they quit the circular trajectory and go back in direct navigation toward the origin.

In figure 5 (**right**), we note the precision of this control law. In fact, all the inter-distances between neighbouring UAVs converge to the desired value. This precision continues to persist during different phases of aggregation and navigation scenario. The steady-state errors are completely eliminated and the scalability property is still present as we can see in inter-distances between UAVs 5 and 6, and 4 and 5.

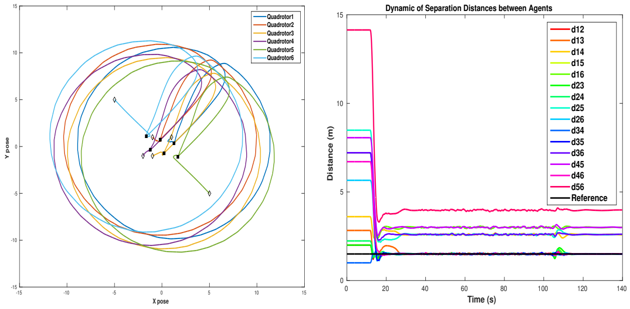

Fig. 5: ***Simulations:*** *(**left**) Trajectories of 6 UAVs in aggregation and circular navigation scenario using distributed integral strategy. (**right**) Inter-distances between 6 UAVs in aggregation and circular navigation scenario using distributed integral control strategy*

VII. Real-Time Experiments

In the following experimental results, our platform is the quadrotor ArDrone2 from Parrot [33]. Using an SDK (Software Development Kit) provided by Parrot, this platform is designed to be controlled, either from a smartphone, a PC through WiFi, or directly by running a program on the UAV via socket. The utilization of SDK prevents us to use our control laws to stabilize the UAV, since the drone has its own control laws designed by Parrot.

We solved this problem through the work of teams from TU Delft University on ArDrone 2 and their autopilot *Paparazzi* [34]. They managed to decode the communication protocols between the main processor of the UAV and its sensors and motors. We used these protocols to directly control the UAV and read the raw data from each sensor. By incorporating these protocols into our own software framework for UAVs, we managed to replace the programs of manufacturer by our own control laws.

The ArDrone 2 is thus mainly used for its material part, whose characteristics are: 1GHz 32 bit ARM Cortex A8 processor, 128 MB RAM, 128MB Flash, WiFi, 3 axis accelerometer, 3 axis gyroscope, 3 axis magnetometer, pressure sensor, ultrasound sensors (altitude $< 6m$) and 4 brushless motors. The inertial sensors are used in a complementary filter [35] to estimate the orientation of the UAV.

Experiments are performed in an indoor environment using Optitrack motion capture system [36]. The system senses the pose of UAVs at 100 Hz. This information is sent to the UAVs through a Cisco router. Each UAV, then, knows the poses of all UAVs in the flock and can then estimate the velocities. In all experiments, we use the Optitrack frame of reference as our global frame I.

We emphasize here that our control laws are run aboard the UAVs. Moreover, our control laws only need the relative distances to the neighbouring UAVs. Since we do not have sensors that measure the relative distances to neighbours, we use the Optitrack system as an alternative to extract inter-distances with neighbours Thus, we calculate the relative distances, aboard on, by using received positions. Moreover, the $a_{ij}(.)$ function in (11) is used to limit the interaction range of UAVs.

A video of interesting real-time experiments with perturbations is available at the link in [37]. Figure (6) shows 6 flying drones in our flight arena during an experiment.

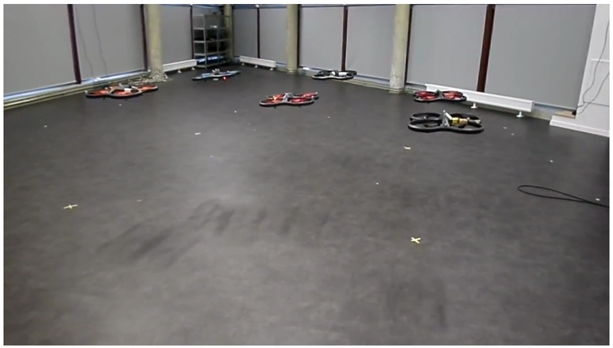

Fig. 6: 6 flying drones during navigation experiment

A. *Flocking control with tuning gains*

In the first experiment, we use four UAVs to form a *quasi-α-lattice* around the origin. We apply our first improved control law in (29). The destination point is defined as the origin of the frame I, $q_r = [0\ 0]^T$. In this experiment, we set $K_p = 0.25, K_d = 0.3, c_1 = 0.1, c_2 = 0.2,\ \varepsilon = 0.1,\ h = 0.2,\ c = 2$, the desired distance between neighbours is $d = 1.5$ and the parameters $a = b = 1$.

Figure 7 shows the trajectories of UAVs of this experiment. UAVs start at their initial positions designated as black diamonds. Then, they start moving toward the desired destination while they avoid collision with their neighbours. A *quasi-α-lattice* is finally formed.

Figure 8 shows the result of using the first control law in (29) in the first real-time experiment. The figure exhibits the distances between UAVs over time. In the experiment, we note, however, a steady-state error in the inter-distances between UAVs, i.e. the desired inter-distances are not completely reached. This steady-state error could be explained by the presence of continuous perturbations in the real-time experiment. One of the sources of these perturbations is the downwash of rotor blades.

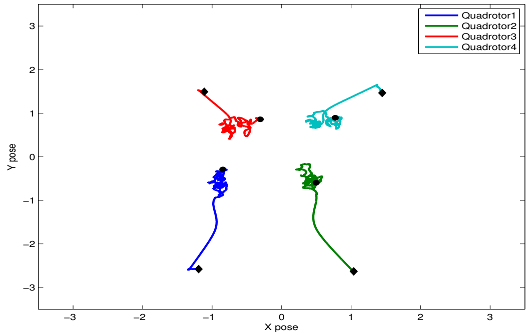

Fig. 7: ***Experiments:*** *Trajectories of 4 UAVs in an aggregation scenario using tuning gains control*

B. *Flocking with distributed integral control*

In the second experiment, we apply our control law in (56) to perform a flocking of multiple UAVs following a circular path. The origin of this circular path is designated to be the origin. The parameters of this control law is set as $K_p = 0.25$, $K_d = 0.3$, $K_i = 0.09$, $c_1 = 0.1$, $c_2 = 0.2$, $\varepsilon = 0.1$, $h = 0.2$, $c = 2$, the desired distance between neighbours is $d = 1.5$ and the parameters $a = b = 1$.

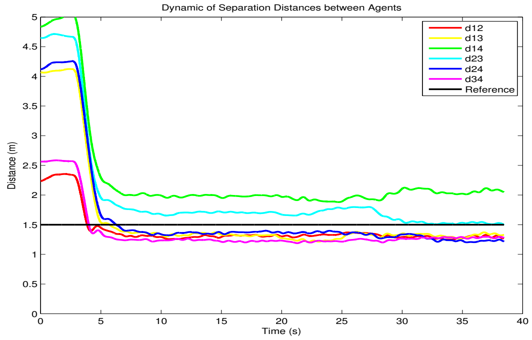

Fig. 8: ***Experiments:*** *Inter-distances between 4 UAVs in an aggregation scenario using tuning gains control*

Figure 9 shows an aggregation and a circular navigation scenario of 6 UAVs. The UAVs perform an aggregation toward the origin and form an $\alpha-$lattice pattern. Then, they follow a circular trajectory while keeping the formation. Unlike the simulation results, because of the experimental room size, the radius of circular path is limited to $1.5m$, this explains the intersections between UAVs trajectories. Figure 10 shows the inter-distances between UAVs that converge to the desired value $1.5m$ even with the perturbations caused by the reflected wind from rotors.

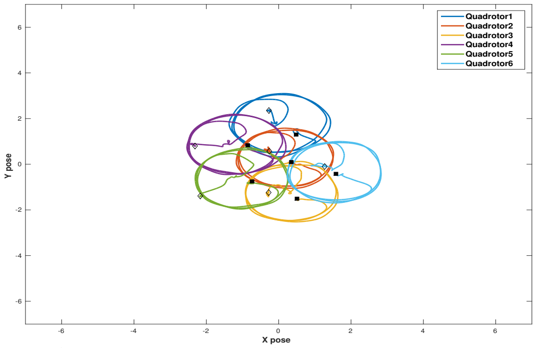

Fig. 9: ***Experiments:*** *Trajectories of 6 UAVs in an aggregation and circular navigation scenarios using distributed integral control*

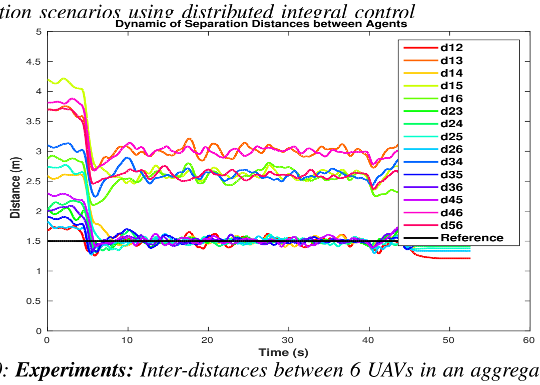

Fig. 10: ***Experiments:*** *Inter-distances between 6 UAVs in an aggregation and circular navigation scenarios using distributed integral control*

Another experimental result using the control law in (56) is shown in figures 11, 12, 13 and 14. In this experiment, we perform only the aggregation scenario toward the origin and we apply external perturbations on UAVs. In figure 13, a push perturbation is applied to UAV 5 and we notice that UAVs resist the perturbation and the inter-distances between neighbouring UAVs return back to the desired value, see figure 12 at $t \approx 50s$ and $55s$. In figure 14, we pull UAV 5 out of the formation, then it returns back to the flock and its inter-distances with the neighbouring UAVs converge again to the desired value, see figure 12 at $t \approx 70s$. Another push is performed at $t \approx 80s$.

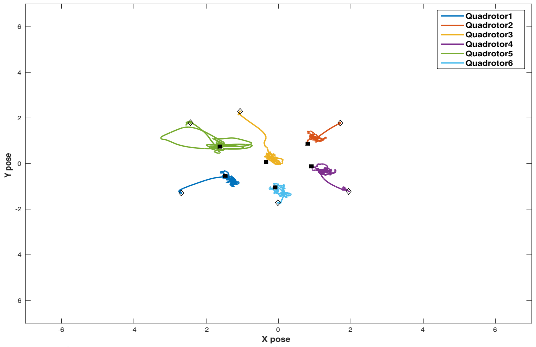

Fig. 11: ***Experiments:*** *Trajectories of 6 UAVs in an aggregation scenario with external perturbations using distributed integral control*

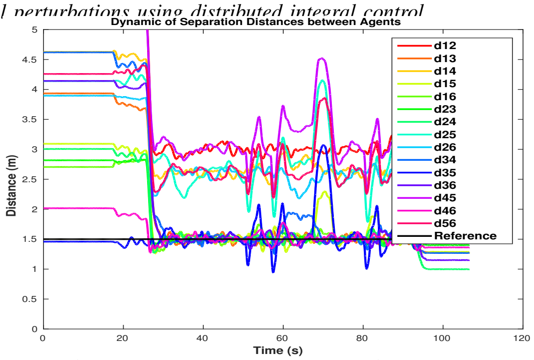

Fig. 12: ***Experiments:*** *Inter-distances between 6 UAVs in an aggregation scenario with external perturbations using distributed integral control*

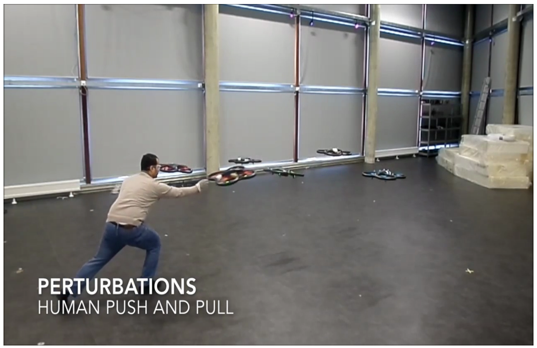

Fig. 13: Experiment with push perturbation

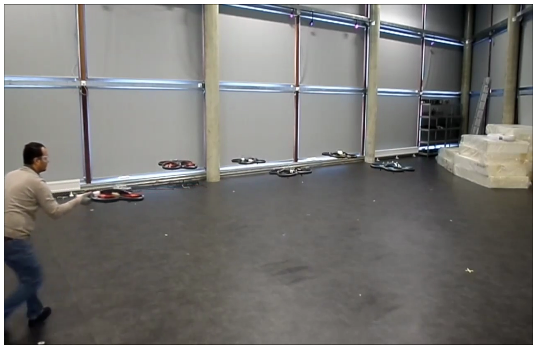

Fig. 14: Experiment with pull perturbation

VIII. Conclusion

In this work, we have proposed two flocking algorithms of multiple-UAV system. The first algorithm ensures the ultimate boundedness of the solutions of multiple-UAV system . The second algorithm improves the first one by adding a distributed integral control term ensuring a precise inter-distance between UAVs. The algorithms were validated through theoretical analysis as well as experimental tests. Moreover, we showed that algorithm is more efficient when applied to nonlinear model compared to other works in the literature.

In this work, we have limited our study on a linearized model of the UAV dynamics to simplify the design of the control laws. Moreover, in our experiments we kept the state of the UAVs around the equilibrium points (rotational angles around zeros). In a future work, we will incorporate the nonlinear model and study how we can design an appropriate control law for the flocking problem.

Our future works will also focus on the improvement of these algorithms and our experimental platform for outdoor experiments. In addition, we will work on the navigation of multiple quadrotors with obstacle avoidance.

Acknowledgement

This work was carried out and funded in the framework of the Labex MS2T (Reference ANR-11-IDEX-0004-02) and the ROBOTEX Equipment of Excellence (Reference ANR-10-EQPX-44), which were supported by the French Government, through the program Investments for the future managed by the National Agency for Research. Moreover, The authors would like to thank Guillaume Sanahuja, who is research engineer in Heudiasyc laboratory, for his efforts in developing the Simulator and the experimental platform and for his help in implementing the control laws in the quadrotors.

References

[1] Giulietti, F., Pollini, L., Innocenti, M.: 'Autonomous formation flight', *IEEE Control Systems*, 2000, **20**, (6), pp. 34–44

[2] Hou, Z., Fantoni, I. 'Distributed leader-follower formation control for multiple quadrotors with weighted topology'. In: 10th System of Systems Engineering Conference (SoSE), 2015. pp. 256–261

[3] Guerrero, J.A., Lozano, R.: 'UAV flight formation control'. (John Wiley-ISTE, 2012). Available from: http://hal.archives-ouvertes.fr/hal-00660317

[4] Chiaramonti, M., Giulietti, F., Mengali, G. 'Formation control laws for autonomous flight vehicles'. In: 14th Mediterranean Conference on Control and Automation. MED 2006. pp. 1–5

[5] Vasarhelyi, G., Viragh, C., Somorjai, G., Tarcai, N., Szorenyi, T., Nepusz, T., et al. 'Outdoor flocking and formation flight with autonomous aerial robots'. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2014. pp. 3866–3873

[6] Roldo, V., Cunha, R., Cabecinhas, D., Silvestre, C., Oliveira, P.: 'A leader-following trajectory generator with application to quadrotor formation flight', *Robotics and Autonomous Systems*, 2014, **62**, (10), pp. 1597 – 1609.

[7] Jasim, W., Gu, D.: 'Robust team formation control for quadrotors', *IEEE Transactions on Control Systems Technology*, 2018, **26**, (4), pp. 1516–1523

[8] Hou, Z., Fantoni, I.: 'Interactive leader-follower consensus of multiple quadrotors based on composite nonlinear feedback control', *IEEE Transactions on Control Systems Technology*, 2018, **26**, (5), pp. 1732–1743

[9] Yu, S., Yu, Z., Jiang, H., Hu, C.: 'Leader-following guaranteed performance consensus for second-order multi-agent systems with and without communication delays', *IET Control Theory Applications*, 2018, **12**, (15), pp. 2055–2066

[10] Shi, H., Wang, L., Chu, T.: 'Virtual leader approach to coordinated control of multiple mobile agents with asymmetric interactions', *Physica D: Nonlinear Phenomena*, 2006, **213**, (1), pp. 51 – 65.

[11] Kushleyev, A., Mellinger, D., Powers, C., Kumar, V.: 'Towards a swarm of agile micro quadrotors.', *Autonomous Robots*, 2013, **35**, pp. 287–300

[12] Schollig, A., Augugliaro, F., Lupashin, S., D'Andrea, R. 'Synchronizing the motion of a quadrocopter to music'. In: IEEE International Conference on Robotics and Automation (ICRA), Anchorage, Alaska, 2010. pp. 3355–3360

[13] Reynolds, C.W. ‘Flocks, herds, and schools: A distributed behavioral model’. In: Computer Graphics. 1987. pp. 25–34

[14] Olfati.Saber, R.: ‘Flocking for multi-agent dynamic systems: algorithms and theory’, *IEEE Transactions on Automatic Control*, 2006, **51**, (3), pp. 401–420

[15] Tanner, H.G., Jadbabaie, A., Pappas, G.J.: ‘Flocking in fixed and switching networks’, *IEEE Transactions on Automatic Control*, 2007, **52**, (5), pp. 863–868

[16] Antonelli, G., Arrichiello, F., Chiaverini, S.: ‘Flocking for multi-robot systems via the null-space-based behavioral control’, *Swarm Intelligence*, 2010, **4**, (1), pp. 37–56

[17] Li, Z., Hovakimyan, N., Stipanovic, D. ‘Distributed multi-agent tracking and estimation with uncertain agent dynamics’. In: American Control Conference (ACC) 2011. pp. 2204–2209

[18] Gao, J., Xu, X., Ding, N., Li, E.: ‘Flocking motion of multi-agent system by dynamic pinning control’, *IET Control Theory Applications*, 2017, **11**, (5), pp. 714–722

[19] Feng, Y., Zheng, W.X.: ‘Group consensus control for discrete-time heterogeneous first- and second-order multi-agent systems’, *IET Control Theory Applications*, 2018, **12**, (6), pp. 753–760

[20] Chu, X., Peng, Z., Wen, G., Rahmani, A.: ‘Robust fixed-time consensus tracking with application to formation control of unicycles’, *IET Control Theory Applications*, 2018, **12**, (1), pp. 53–59

[21] Su, H., Wang, X., Lin, Z.: ‘Flocking of multi-agents with a virtual leader’, *IEEE Transactions on Automatic Control*, 2009, **54**, (2), pp. 293–307

[22] Diestel, R.: ‘Graph Theory’. vol. 173 of *Graduate Texts in Mathematics*. 3rd ed. (Springer-Verlag, Heidelberg, 2005).

[23] Lozano, R.: ‘Unmanned Aerial Vehicles Embedded Control’. (John Wiley-ISTE Ltd, 2010).

[24] Mahony, R., Kumar, V., Corke, P.: ‘Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor’, *IEEE Robotics Automation Magazine*, 2012, **19**, (3), pp. 20–32

[25] Fresk, E., Nikolakopoulos, G. ‘Full quaternion based attitude control for a quadrotor’. In: 2013 European Control Conference (ECC), 2013. pp. 3864–3869

[26] Bouabdallah, S., Siegwart, R. ‘Full control of a quadrotor’. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), California, USA. 2007. pp. 153–158

[27] Teel, A.R.: ‘Global stabilization and restricted tracking for multiple integrator with bounded control’, *Systems and Control Letters*, 1992, **18**, pp. 165–171

[28] Johnson, E.N., Kannan, S.K. ‘Nested saturation with guaranteed real poles’. In: Proceedings of the 2003 American Control Conference, Colorado, USA. vol. 1. 2003. pp. 497–502 vol.1

[29] Sanahuja, G. ‘Commande et localisation embarquée d’un drone aérien en utilisant la vision’. Université de technologie de compiègne, 2010

[30] Kendoul, F., Lara, D., Fantoni, I., Lozano, R.: ‘Real-time nonlinear embedded control for an autonomous quadrotor helicopter’, *Journal of Guidance, Control, and Dynamics*, 2007, **30**, pp. 1049–1061

[31] Godsil, C., Royle, G.: ‘Algebraic graph theory’. vol. 207 of *Graduate Texts in Mathematics*. (New York: Springer-Verlag, 2001).

[32] Khalil, H.K.: ‘Nonlinear Systems’. (Prentice Hall, 2002).

[33] ParrotArdrone2. http://ardrone2.parrot.com/.

[34] TUDelft. http://wiki.paparazziuav.org/wiki/AR_Drone_2/getting_started

[35] Mahony, R., Hamel, T., Pflimlin, J.M.: ‘Nonlinear complementary filters on the special orthogonal group’, *IEEE Transactions on Automatic Control*, 2008, **53**, (5), pp. 1203–1218

[36] Optitrack. ‘Optitrack motion capture systems’. https://www.naturalpoint.com/optitrack/.

[37] Experimental results: https://youtu.be/ylKDuHQBwv8.