An Open-Source Architecture for Control and Coordination of a Swarm of Micro-Quadrotors

M. Furci, G. Casadei, R. Naldi, R.G. Sanfelice and L. Marconi

***Abstract*— This work presents the design of an open-source control architecture specifically tailored to the rapid development and testing of control and coordination algorithms on micro quadrotors. The proposed design extends an existing open-source and open-hardware quadrotor project, i.e., the Crazyflie nano quadrotor, by adding the guidance, navigation and control layers required to accomplish autonomous flight. The control layer, in particular, is based on a cascade control algorithm able to globally stabilize the position and the attitude of the vehicle. The global property guaranteed by the algorithm allows the quadrotors to perform aggressive maneuvers. The guidance layer is designed to coordinate simultaneously multiple vehicles by generating suitable reference trajectories. Experiments demonstrate the effectiveness of the proposed design.**

I. Introduction

A. Background and Motivation

Quadrotor aerial vehicles are currently employed successfully in a large variety of applications including, among others, surveillance, aerial photography and search and rescue operations [1]. One reason for this large success is the high level of maneuverability [2], [3], [4] which allows to safely perform flight missions even in densely populated environments [5] or to perform advanced robotic operations [6]. All of these vehicles are under-actuated mechanical systems, namely the number of available control inputs is less than the number of degrees-of-freedom (d.o.f.). As a consequence, to achieve the level of agility required by real-world applications, the feedback control design plays a central role.

Several contributions in the literature document the effectiveness of rapid prototyping control frameworks for the development and testing of coordination and control algorithms [7], [8]. Most of these frameworks are based on motion tracking systems in which ad-hoc quadrotor platforms can be employed to perform advanced maneuvers. While the control algorithms are often released by the authors, the overall control architecture may not be directly available to other research groups. More recently, an open-source and open-hardware project, the Crazyflie [9], has proposed a miniature low-cost quadrotor platform. The vehicle can be easily piloted by a human operator though a remote computer by means of a standard joystick interface. However, the currently available software architecture does not include the guidance, control and navigation layers required to test closed-loop control algorithms and to perform advanced operations.

This research has been conducted in part under the collaborative project SHERPA supported by the European Community under the 7th Framework Programme.

M. Furci, G. Casadei, R. Naldi and Lorenzo Marconi are with C.A.SY. – DEI, University of Bologna, Bologna, Italy ; R.G. Sanfelice is with Department of Computer Engineering, UCSC. Mail: michele.furci@unibo.it, g.casadei@unibo.it, roberto.naldi@unibo.it, lorenzo.marconi@unibo.it, ricardo@ucsc.edu.

B. Contribution

The goal of this paper is to develop an open source control framework to allow the Crazyflie nano quadrotors to be employed as a test bench for the development of advanced control and coordination algorithms. In particular, the guidance, control, and navigation layers have been designed to let a group of nano quadrotors to be controlled simultaneously using a motion tracking system. The control layer is based on a novel trajectory tracking controller [10] able to globally stabilize the position and the attitude of the vehicle. In particular, the proposed algorithm overcomes the topological obstruction [11] affecting continuous state-feedback stabilizers by relying upon hybrid control techniques. This feature allows to exploit the agility of the selected nano quadrotors so as to perform acrobatic maneuvers as well as to effectively recover the vehicle from an arbitrary initial configuration. The proposed algorithm relies on a cascade control structure based in which an inner attitude control loop plays the role of a virtual input for the outer position control loop. This structure is exploited to distribute the computation on the ground and the onboard embedded processor. More specifically, the attitude loop is implemented on the onboard processor while the outer position loop as well as the overall guidance layer are implemented on a remote ground station, which consists of a PC. The guidance layer is in charge of generating the reference position and orientation to be tracked by the cascade controller and also to coordinate multiple nano quadrotors.

The proposed control architecture is then validated by means of an experiment in which two different vehicles are required to perform a coordinated maneuver. The experiment also shows the capability of the proposed control layer to recover the vehicle from an arbitrary initial configuration.

C. Paper Organization

The paper is organized as follows. In Section II, we briefly introduce the Crazyflie quadrotors used to implement the architecture and to perform the experiments. In Section III, we describe the control architecture and all the components involved in the design, while in Section IV, we describe briefly the control law implemented.

Finally, in Section V, we display the experimental setup used to test the proposed architecture, and show experimental results.

D. Notation and Definitions

Throughout this paper, $\mathcal{F}_i$ and $\mathcal{F}_b$ denote, respectively, an inertial reference frame and a reference frame attached to the center of gravity of the vehicle. With $I_n \in \mathbb{R}^{n\times n}$ we denote the n-dimensional identity matrix. With e_1, e_2 and e_3 we denote the unit vectors $e_1 := [1,\, 0,\, 0]^T$, $e_2 := [0,\, 1,\, 0]^T$ and $e_3 := [0,\, 0,\, 1]^T$. For any $x \in \mathbb{R}^3$, we let

$$S(x) := \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

be a skew-symmetric matrix and we denote with $^\wedge$ the inverse operator such that $S(x)^\wedge = x$. With S_n we denote the n-dimensional unit sphere defined as $S_n := \{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$. A unit quaternion $q \in S_3$ is defined as a pair $q = [\eta,\, \epsilon^T]^T$ in which $\eta \in \mathbb{R}$ and $\epsilon \in \mathbb{R}^3$ are denoted respectively as the scalar and vector part. Given unit quaternions $q_1 = [\eta_1,\, \epsilon_1^T]^T$ and $q_2 = [\eta_2,\, \epsilon_2^T]^T$, the standard quaternion product is defined as

$$q_1 \otimes q_2 = \begin{bmatrix} \eta_1 & -\epsilon_1^T \\ \epsilon_1 & \eta_1 I_3 + S(\epsilon_1) \end{bmatrix} \begin{bmatrix} \eta_2 \\ \epsilon_2 \end{bmatrix}.$$

Rotations can be parameterized by means of a unit quaternion $q \in S_3$ through the mapping $\mathcal{R} : S_3 \to SO(3)$ known as Rodrigues formula [12] and defined as

$$\mathcal{R}(q) = I + 2\eta S(\epsilon) + 2S(\epsilon)^2\,.$$

The mapping $\mathcal{R}$ is such that $\mathcal{R}(q) = \mathcal{R}(-q)$, namely the two quaternions q and $-q$ corresponds to the same rotation matrix.

We refer to a *saturation function* as a mapping $\sigma : \mathbb{R}^n \to \mathbb{R}^n$ such that, for $n = 1$,

1) $|\sigma'(s)| := |d\sigma(s)/ds| \leq 2$ for all s,
2) $s\sigma(s) > 0$ for all $s \neq 0$, $\sigma(0) = 0$,
3) $\sigma(s) = \mathrm{sgn}(s)$ for $|s| \geq 1$,
4) $|s| < |\sigma(s)| < 1$ for $|s| < 1$.

For $n > 1$, the properties listed above are intended to hold componentwise.

II. Crazyflie Platform

The Crazyflie nano-quadrotor is an open-source, open-hardware project developed by the company Bitcraze AB [9]. The available components [1] include the airframe, the onboard avionics hardware, a wireless communication device and the software packages that allow a human pilot to govern the vehicle by means of a standard joystick connected to a ground PC. The following two subsections describe the available hardware and software components, respectively.

[1]We have considered the 6 DoF version of Crazyflie nano-quadrotor.

A. Hardware

- *Crazyflie Quadrotor:* Because of its dimensions - 90 mm from rotor to rotor - and flight time - up to 7 minutes - the Crazyflie falls under the nano-quadrotor category [13]. Its small weight, 19 grams, and the low-cost of all the components make it suitable to safely perform experiments without the risk of damaging expensive hardware. Moreover, the high thrust-to-weight ratio (the maximum thrust is more than 35 grams) and torque-to-inertia ratio make it suitable to perform aerobatic maneuvers. The core of the airframe is given by the Printed Circuit Board (PCB), which includes the microprocessor (an ARM Cortex $M3$), the Inertial Measurement Unit (IMU) sensor - 3-axis gyros and 3-axis accelerometers integrated in a single $MPU6050$ chip - and the power circuit for the motors. Each of the four DC current motors drives a fixed-pitch propeller having a diameter of 1.8 inches.
- *Crazyflie Wireless Communication Device:* The Crazyflie setup includes a wireless radio for bidirectional communication between the quadrotor and a ground station. The radio device, denoted as *Crazyflie Dongle*, is characterized by a frequency of 2.4 GHz and it is based on a Nordic Semiconductor $nRF24LU1$ chip. The wireless communication is based on the *Enhanced ShockBurst* protocol. The wireless device supports approximately 100 different channels simultaneously. Accordingly, the hardware architecture is suitable to employ a large number of different nano quadrotors simultaneously.

B. Software

The original software released by the company includes a PC application, the *Crazyflie Client*, and the on-board firmware. The two components, which are described hereafter, are specifically designed to let the vehicle be easily piloted by a human operator.

- *Crazyflie Client:* the main goal of the *Crazyflie Client* is to let the Crazyflie nano quadrotor communicate with a ground station PC. From a hardware point of view, the communication is obtained by means of the wireless device described in the previous subsection. The *Crazyflie Client* includes a user interface for manual set-up and operations with the quadrotor. In particular, the user interface presents to the operator a number of useful information (battery status, current motor thrust, etc.) and it allows to change the flight parameters in real-time. In the standard flight mode, the *Crazyflie Client* reads the position of the four axis of a standard joystick device attached to the ground PC in order to produce, as output, the set of commands to be sent to the onboard avionics. In particular, the commands are the attitude roll and pitch angle, the yaw angular speed, and the resultant thrust to be produced by the four propeller.

- *Crazyflie Firmware:* the firmware is the code running on the on-board microcontroller. The firmware runs on a Real-Time Operative System (FreeRTOS) able to handle multiple threads. The most important components include the sensor drivers, the attitude estimation filter, the attitude control law, the communication layer, and the motor control. The attitude estimator, in particular, is based on a complementary filter proposed in [14]. The attitude control law is based on a simple PID control loop that has been implemented to stabilize the three Euler angles, i.e., the roll, the pitch and the yaw, parameterizing the attitude of the vehicle. As a consequence, the available attitude control algorithm suffers from singularities deriving from the attitude parameterization and it is not able to globally stabilize a desired angular position.

III. Motivation and Control Architecture

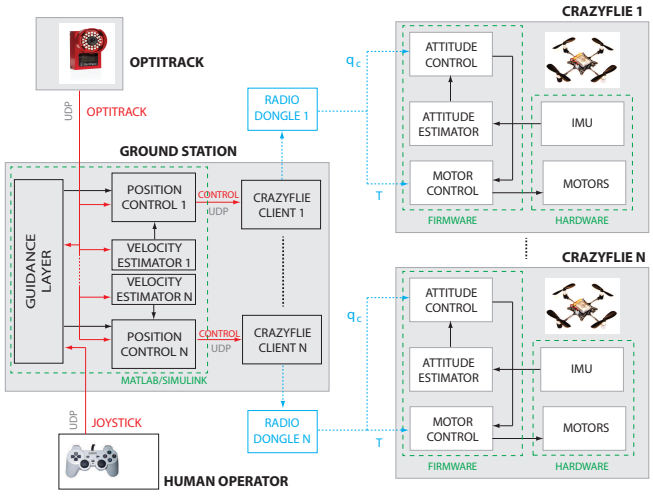

Fig. 1: Control Framework

A. Motivation and Architecture

The idea behind this work is to define a *distributed layered control architecture*. By taking advantage of the Crazyflie platform described in Section II, this work derives a control architecture suitable for experimental tests of a swarm of quadrotors. The architecture can be divided into these layers:

- Control Layer is designed to achieve:
  - a cascade closed loop control scheme with decentralized computation;
  - the controller to be global asymptotically stable, which means that for any possible initial condition the position and attitude error is steered to zero asymptotically;
- Navigation Layer is designed to obtain information about the state of the system: it relies on sensors (IMU, gyroscopes, Optitrack System) and estimation filters (filters for the attitude and velocity estimation);
- Guidance Layer: define a coordination protocol to control a swarm of quadrotors.

The overall control architecture is depicted in Figure 1 where the main interactions between hardware and software components, necessary to achieve the proposed goals, are shown. The core elements of this architecture, which are detailed in the following, are given by the Crazyflie quadrotor, the Optitrack System, the human-machine interface and the ground station.

- *Optitrack System:* is an off-the-shelf real-time motion tracking solution based on infrared cameras [15]. The set of camera with the legacy software package Tracking Tools, are able to provide attitude and position of rigid bodies at a frequency of 100 Hz. A set of infra-red reflective markers needs to be pinned to the rigid-body to be tracked (in our setup, 3 markers for each Crazyflie quadrotor). A flight arena, in which 12 infrared cameras are employed, has been used in the experiments. The resulting tracking volume is approximately given by a box of $4 \times 4 \times 2$ meters. Any tracking system could be substituted to Optitrack by sending appropriate UDP packets following the protocol described later;
- *Human-Machine Interface:* the human operator (pilot) can communicate with the ground station through a joystick. By means of the joystick the operator can interact with the guidance layer trajectory, and guarantee a safe flight termination. From a software point of view, the commands generated by the human via the joystick are processed by a control algorithm implemented using Matlab/Simulink;
- *Ground Station:* it consists of a desktop PC running Matlab/Simulink with RTWTK (Real Time Windows Target Kernel) and Crazyflie Clients. The Simulink software is employed to rapidly implement and validate all the control and coordination algorithms. An open source library has been developed to interpret the joystick commands and data received from the Optitrack System and transmit them, via UDP protocol, to the Simulink controller.

1) Control Layer: A cascade control structure is chosen to reflect the intrinsic physical connection between attitude and position of the quadrotor model (See Section IV). This strategy allows a simple and intuitive tuning of the attitude and position controllers separately.

The cascade control structure requires frequency separation between the two loops, in which the attitude is referred as the inner loop while the position is the outer loop. Thus, the attitude controller is running on-board to fast compute attitude sensors data, while the position control is running on the ground station PC using Optitrack position data.

The outer loop layer (position) is running on the ground station PC at 100 Hz, while the inner loop layer (attitude) is running on-board on the Crazyflie quadrotor at 250 Hz. The communication between these layers is described in Section III-B, while the control law is defined in Section IV.

2) Navigation Layer: The Optitrack measured yaw is sent on-board to allow yaw angle control, since the Crazyflies PCB does not embed a magnetometer and thus the on-board

estimation diverges. The Optitrack System measures only the position. Hence, a high gain filter for each quadrotor is implemented on Simulink to estimate the linear velocity. The linear velocity estimation is necessary to implement the control law described in Section IV.

Also, with the aforementioned control law in mind, the original on-board attitude estimation filter is slightly modified to evaluate the attitude in terms of quaternion instead of Euler Angles.

3) Guidance Layer: All the high level logic (state machines, trajectory generation, swarm algorithms, etc.) can be implemented directly on Simulink, depending on the specific experiment to perform. The *Guidance Layer* is defined as the layer that aims to control the overall behavior of the system, being a single quadrotor or a swarm. In therms of output, the Guidance Layer generates reference trajectories to be tracked by the Control Layer: thus, from a functional point of view, it can be seen as a complex reference generator.

B. Communication Protocols for Architecture Design

On the Ground Station PC, the communication between the layers is obtained by means of UDP protocols while the communication from the Ground Station PC to the the quadrotors is performed by means of the Crazyflie standard radio communication.

"Stream Input" and "Stream Output" Simulink blocks (Real Time Windows Target Toolbox) are used to receive input data from Optitrack and Joystick into Simulink and to send output data from Simulink to the Client, respectively. The Crazyflie standard radio communication protocol is used to send the desired attitude and thrust from the Crazyflie Client to the quadrotors on-board. Three types of messages can be defined (see Figure 2):

- Optitrack Message: Used to send UDP data from the Optitrack System to Simulink; a single packet is used for every quadrotor, and the ID of the vehicle is distinguished using different UDP port for every agent;
- Joystick Message: Used to send UDP data from Joystick to Simulink;
- Control Message: Used to send UDP data from Simulink to the Client.

Message Type	FIELDS											
OPTITRACK	LEN	TYPE	**X**	**Y**	**Z**	**Q0**	**Q1**	**Q2**	**Q3**	ERR	T	CRC
JOYSTICK	LEN	TYPE	**A**	**B**	**C**	**D**	**BUT**	T	CRC	-	-	-
CONTROL	LEN	TYPE	**R**	**P**	**Y**	**THR**	**YAW**	T	CRC	-	-	-

Fig. 2: Messages Protocol Definition

Fields description:

- X, Y, Z: x, y, z positions from optitrack of the tracked quadrotor
- Q0, Q1, Q2, Q3: the four component of the attitude of the tracked quadrotor, expressed with quaternion
- A, B, C, D: the values from the four axis of joystick (from 0 to 1024)
- BUT the value of the button of joystick
- R, P, Y, THR: roll, pitch, yaw and thrust reference sent to the client.
- YAW: the actual yaw measured by the Optitrack System, to be sent to the client
- T: timestamp (data not used in the current architecture)
- ERR: mean tracking error of optitrack system (data not used in the current architecture)
- CRC: cyclic redundancy check for packets integrity check

To allow the use of multiple quadrotors with the proposed architecture, multiple Clients should be used on the Ground Station. Each quadrotor is associated to one Crazyflie Client, each one reading from a different UDP ports and communicating data on-board via a private Dongle. The client is modified to allow the scan for multiple Dongles and to be able to link every dongle to a quadrotor.

A preliminary code example, including the modified on-board firmware, the modified Client and the Simulink control scheme for multiple Crazyflies can be found in [16].

IV. Global Trajectory Tracking Control Law

Following [10], we shortly describe the implemented control law.

A. Quadrotor Dynamical Model

The dynamical model of the Crazyflie nano quadrotor can be described by considering the so called *vectored-thrust* approximation [17]:

$$\begin{aligned} M\ddot{p} &= -TRe_3 + Mge_3 \\ \dot{R} &= RS(w) \\ J\dot{\omega} &= S(J\omega)\omega + \tau \end{aligned} \tag{1}$$

where M and J are the mass and the moment of inertia matrix of the vehicle, $p = [x,\, y,\, z]^T \in \mathbb{R}^3$ is the position in inertial frame $\mathcal{F}_i$, $R \in SO(3)$ is the rotation matrix expressing the rotation of the body frame with respect to inertial frame, $\omega \in \mathbb{R}^3$ is the angular speed of the vehicle expressed in the body frame $\mathcal{F}_b$, $T \in \mathbb{R}_{\geq 0}$ and $\tau \in \mathbb{R}^3$ are the thrust and the torques generated by the four propellers, respectively.

B. The Control Goal

Goal of the controller to be designed is to track a desired time reference trajectory

$$t \mapsto p_R(t),\, t \mapsto R_R(t) \tag{2}$$

with $p_R(t)$, $R_R(t)$ sufficiently smooth functions of time. Due to the under-actuation of the dynamical model (1), the reference attitude is required to satisfy some constraints in order to guarantee the feasibility of the position tracking objective. More specifically, with an eye on (1), the following relation must be satisfied

$$R_R e_3 = \frac{v_R}{|v_R|} \tag{3}$$

with $v_R := -M\ddot{p}_R + Mge_3$ satisfying $|v_R(t)| > 0$ for all $t \geq 0$. The above constraint is requiring the spin axis of the propellers to be oriented so as to produce the force required

to track the desired position trajectory. Let us assume that the class of references is such that $e_3^T v_R > 0$, namely the desired vertical acceleration is lower bounded by the gravity acceleration. From (3) we can compute the desired reference rotation matrix as $R_R = R_x(\phi_R)R_y(\theta_R)R_z(\psi_R)$, where R_i, $i \in \{x, y, z\}$, are elementary rotation matrices around the body axis, $\phi_R := \arctan\left(-e_2^T v_R / e_3^T v_R\right)$, $\theta_R := \arcsin(e_1^T v_R)$, and, finally, $\psi_R \in \mathbb{R}$ is a degree-of-freedom corresponding to the desired heading orientation that can be chosen arbitrarily without affecting the position tracking objective. In summary, by taking into account for the under-actuation of the vehicle, the control goal consists of tracking the reference position and heading angle trajectories given by

$$t \mapsto p_R(t),\, t \mapsto \psi_R(t) \tag{4}$$

where the position trajectory satisfies $e_3^T v_R(t) > \underline{v}_z > 0$ for all $t \geq 0$. Note that the above constraint also implies that $|v_R(t)| > 0$ so as (3) is well defined.

C. The Cascade Control strategy

In order to address the control problem derived in the previous section, we consider a cascade control structure having the attitude and the position control loops playing the role of the *outer* and *inner* loop, respectively.

1) Position control law: Let us consider the following control vector

$$v_c(\eta_p, \tilde{p}, \dot{\tilde{p}}) := v_R + \kappa(\tilde{p}, \dot{\tilde{p}}, \eta_p)$$

in which $\tilde{p} := p - p_R$ denotes the position error coordinates, and

$$\kappa(\cdot) := \lambda_3 \sigma\left(\frac{k_3}{\lambda_3}\left(\dot{\tilde{p}} + \lambda_2\sigma\left(\frac{k_2}{\lambda_2}\tilde{p} + \lambda_1\sigma\left(\frac{k_1}{\lambda_1}\eta_p\right)\right)\right)\right) \tag{5}$$

with λ_i, k_i, $i \in \{1, 2, 3\}$, control parameters to be tuned, and where η_p denotes the state of the following integrator

$$\dot{\eta}_p = \tilde{p}\,.$$

The integrator is included to be robust to constant disturbances due to non perfect parameters estimation. In order to apply the control vector to the position, following the same arguments as in the previous section, we design the control thrust as $T = |v_c|$ and the control attitude R_c as

$$R_c e_3 = \frac{v_c}{|v_c|} \tag{6}$$

with $v_c(0, 0, 0) \equiv v_R$. As a consequence, if $\eta_p = \tilde{p} = \dot{\tilde{p}} = 0$, the desired position trajectory is tracked and $R_c \equiv R_R$, i.e., the desired heading angle ψ_R is stabilized. Essentially, from [10] the control law was slightly modified to introduce an integral term to be more robust to constant or slow varying disturbances (as battery discharge) or to wrong parameters estimation (for example the mass).

Remark 1 *It is worth to point out that the above position controller is essentially a saturated PID control law. Saturations are introduced to guarantee that the desired control vector v_c is non vanishing and the rotation R_c can be computed as proposed in Section IV-B.*

2) Attitude: Goal of the attitude control loop is to stabilize the desired control attitude R_c so as, on one hand, to allow the position controller to track the desired trajectory $p_R(t)$ and, on the other, to track the desired heading angle $\psi_R(t)$ given in (4).

Let $q_c = [\eta_c, \epsilon_c^T]^T$ be the control quaternion that is computed from from the rotation matrices R_c applying the path-lifting mechanism proposed in [18]. Inspired by [19], [10], we consider the following hybrid attitude control law:

$$\tau_c^\star = \tau_c^{FF}(\bar{q}, w_c, \dot{w}_c) + \tau_c^{FB}(\bar{q}, \bar{w}, \bar{h}) \tag{7}$$

having defined

$$\bar{q} = q_c^{-1} \otimes q,\, \bar{\omega} := \omega - \bar{\omega}_c, \tag{8}$$

with $\bar{\omega}_c := \mathcal{R}(\bar{q})^T \omega_c$ and ω_c, $\dot{\omega}_c$ as

$$\begin{bmatrix} \omega_{c,x} \\ \omega_{c,y} \end{bmatrix} := W_{xy} R_c \frac{d}{dt}\frac{v_c}{|v_c|}\,, \tag{9}$$

$$\begin{bmatrix} \dot{\omega}_{c,x} \\ \dot{\omega}_{c,y} \end{bmatrix} := W_{xy}\left(-S(\omega_c)R_c\frac{d}{dt}\frac{v_c}{|v_c|} + R_c\frac{d^2}{dt^2}\frac{v_c}{|v_c|}\right) \tag{10}$$

where $W_{xy} \in \mathbb{R}^{2\times 3}$ is the matrix with the first and second rows given by $[0, -1, 0]$ and $[1, 0, 0]$, respectively. And where

$$\begin{array}{rcl} \tau_c^{FF} & = & J_m \mathcal{R}(\bar{q})^T \dot{w}_c - S(J\bar{w}_c)\bar{w}_c \\ \tau_c^{FB} & = & -k_p \bar{h}\bar{\epsilon} - k_d \bar{w} \end{array} \tag{11}$$

with k_p, k_d positive gains and where $\bar{h} \in \{-1, 1\}$ is obtained through the following hybrid system

$$\mathcal{H}_c \quad \left\{ \begin{array}{ll} \dot{\bar{h}} = 0 & \bar{h}\bar{\eta} \geq -\delta \\ \bar{h}^+ \in \overline{\text{sgn}}(\bar{\eta}) & \bar{h}\bar{\eta} \leq -\delta \end{array} \right. \tag{12}$$

where $\delta \in (0, 1)$ is the hysteresis threshold and $\overline{\text{sgn}} : \mathbb{R} \rightrightarrows \{-1, 1\}$ is the outer-semicontinous set-valued map

$$\overline{\text{sgn}} = \left\{ \begin{array}{ll} \text{sgn}(s) & |s| > 0 \\ \{-1, 1\} & s = 0\,. \end{array} \right.$$

Remark 2 *Note that the proposed attitude controller is essentially composed of a nonlinear PD feedback controller, a feed-forward control action deriving from the desired references and able to decouple the inner from the outer loop, and a hybrid hysteresis mechanism. The hybrid control algorithm is required to remove the topological obstruction, while the hysteresis mechanism allows to obtain robustness also in the presence of non negligible measurement noise [19].*

D. Motivations behind the proposed hybrid control law

Variuos approaches for the attitude control can be found in literature, yet few of them take into account the topological obstruction that hides behind this problem. One typical approach consists in parametrizing the $SO(3)$ manifold. For instance, the Euler angles attitude representation is a parametrization of $SO(3)$ but it suffers singularities issues. Moreover, since this mapping function is surjective, it could lead to undesired behaviour such as *unwinding* [18].

Another approach consists in the design of the attitude control law directly in $SO(3)$ with a continuous control law. It can be prooved (see [11]) that, the continuous control law, leads to more than one equilibrium point and, if the initial condition is close to this undesired equilibrium points, due to continuity, a large amount of time could be required to converge to the stable equilibrium. The solution to these problems comes from discontinuous control laws that, on the other hand, suffer from measurement noise. When the system state is close to switching condition, an arbitrary small measurement noise may leads undesired switches, i.e. chattering.The solution implemented in this paper is capable of overcoming this drawback by means of an hysteresys region on the switching condition.

V. Experiments

To validate the proposed distributed control law and to show the effectiveness of the overall control architecture, in this section experimental results are presented. We expect two crazyflie nano-quadrotors to track a desired trajectory, generated by the Guidance Layer. The quadrotors are hand deployed by the human operator. One nano-quadrotor is deployed mid-air with a harsh initial condition (it's deployed overturned, facing downward and with initial high linear speed) and thus has to perform an attitude recovery maneuver, assuring that the proposed control law is globally asymptotically stable. The second quadrotor is then deployed. As soon as the quadrotors are deployed and have recovered the hovering state, the desired trajectory is generated and they are requested to track such a reference.

It is important to stress that the harsh initial conditions from which the quadrotor may start, can impact gravely on a small aerial vehicle without a GAS control law: in our case, the proposed control law guarantees that, after a small transient, the quadrotors recover the hovering condition globally.

A. Experimental Setup

Two Crazyflie mini-quadrotors are equipped with a carbon fiber structure, to bear markers for the Optitrack System (Figure 3). To avoid limitation in the thrust, the structure is designed to position the markers outside the airflow. The whole structure and markers weight is approx. 3.5 grams. This payload impacts on the total flight time, which drops from 7 to 5 and a half minutes.

B. Guidance Layer

In this experiment, the Guidance Layer is a simple centralized trajectory generator. As soon as the two quadrotors reach hovering condition, it generates two circular path at constant height, in counter-clockwise direction, with a phase of $\pi\ rad$ and radius of $0.85\ m$.

C. Parameters

The control parameters employed in the experiments are the following.

The mass is computed by adding to the weight of a Crazyflie, the weight of markers and structure. The marker's weight is approx. $1\ g$ each, while the weight of the structure is negligible. It results a final mass of $22\ g$.

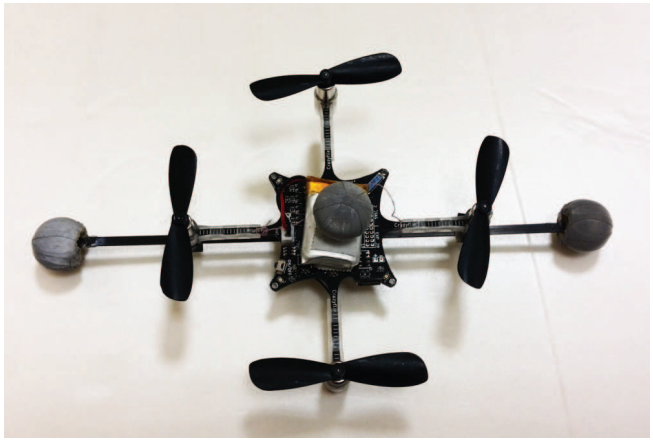

Fig. 3: Crazyflie quadrotor with structure and markers

TABLE I: Control parameters for the experiment

$\lambda_1 = 0.1$	$\lambda_2 = 0.15$	$\lambda_3 = 0.24$
$k_1 = 0.01$	$k_2 = 0.1$	$k_3 = 0.11$
$k_{p,x} = 60000$	$k_{p,y} = 30000$	$k_{p,z} = 40000$
$k_{d,x} = 70$	$k_{d,y} = 60$	$k_{d,z} = 130$
$\delta = 0.1$	$K_T = 8.5e^{-7}$	$K_Q = 1.3e^{-8}$

The inertia tensor is computed using Huygens-Steiner theorem and considering separate masses. In particular, the frame and the marker mounted on the battery are considered as a parallelepiped of dimension 4x4x3 cm and mass $12\ g$. The inertia contribution of this parallelepiped is given by:

$$\begin{aligned} I_{xx,f} &= 13g/cm^2 \\ I_{yy,f} &= 13g/cm^2 \\ I_{zz,f} &= 32g/cm^2 \end{aligned}$$

The motors and propellers are considered as point masses of $2\ g$ at a distance of $4\ cm$ from the center of mass. The markers are considered as point masses centered at $7,5\ cm$ from the center of mass, in the y-axis. Contribution of motors to inertia is given by:

$$\begin{aligned} I_{xx,m} &= 64g/cm^2 \\ I_{yy,m} &= 64g/cm^2 \\ I_{zz,m} &= 128g/cm^2 \end{aligned}$$

Contribution of markers to inertia is given by:

$$\begin{aligned} I_{xx,m} &= 112.5g/cm^2 \\ I_{yy,m} &= 0g/cm^2 \\ I_{zz,m} &= 112.5g/cm^2 \end{aligned}$$

The total inertia is given by:

$$\begin{aligned} I_{xx} &= I_{xx,f} + I_{xx,m} + I_{xx,m} = 189.5g/cm^2 \\ I_{yy} &= I_{yy,f} + I_{yy,m} + I_{yy,m} = 77g/cm^2 \\ I_{zz} &= I_{zz,f} + I_{zz,m} + I_{zz,m} = 272.5g/cm^2 \end{aligned}$$

D. Results

In Figure 4, the attitude (reference and actual) of the first quadtotor is shown: due to the harsh initial condition, it has to perform an acrobatic maneuver (attitude recovery) to reach the desired hovering position before the trajectory starts. Between 7 sec and 9 sec, the human operator manually turns the quadrotor downward. At time 9 sec, the quadrotor is deployed and in the time interval 9 sec to 12 sec, the quadrotor recovers the attitude and reaches the requested position.

We can notice an offset in tracking the 3rd and 4th component of the quaternion during the whole experiments, due to the fact that the real attitude is estimated on-board, while the attitude plotted in Figure 4 is given by the Optitrack System. A sequence of the deployment and attitude recovery maneuver can be seen in Figure 5. The condition (12) for switching the hybrid parameter h of the attitude control is fulfilled when the quadrotor is deployed. The change can be seen in Figure 6.

In Figure 7 the thrust command of first quadrotor is depicted.

In Figure 8 and 9, the position of the two quadrotors performing trajectory tracking is shown: the real position is the black line and the trajectory reference generated by the Guidance Layer is dashed red.

A video of the experiment can be seen at [20].

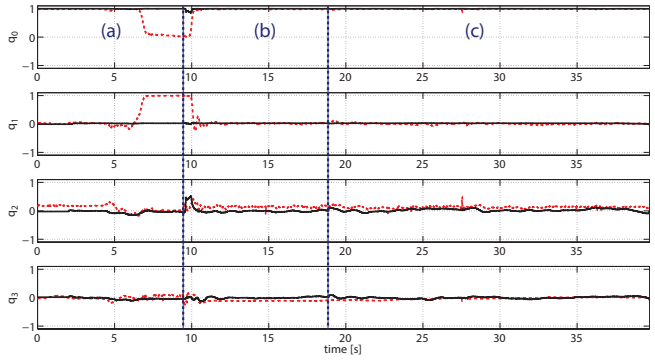

Fig. 4: Desired (black solid) and actual (red dashed) attitude of Quadrotor n.1. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

VI. Conclusion

In this paper we presented the design of an open-source control architecture to test control and coordination algorithm for micro quadrotors. We described the main features of the proposed architecture and its main components (both hardware and software): this architecture relies on a hybrid layered control law with computation distributed between the quadrototrs and a ground station PC. The proposed solution was then tested and experimental results were presented.

This architecture allows to rapidly test new control laws and in particular to implement coordination protocols for the multi-quadrotors scenario: in future works, we will focus on the *so called* Guidance Layer, in order to accomplish collective behavior and to perform complex task.

Software-wise, we will investigate the possibility of directly exchanging information between Crazyflie quadrotors with the purpose to implement completely decentralized

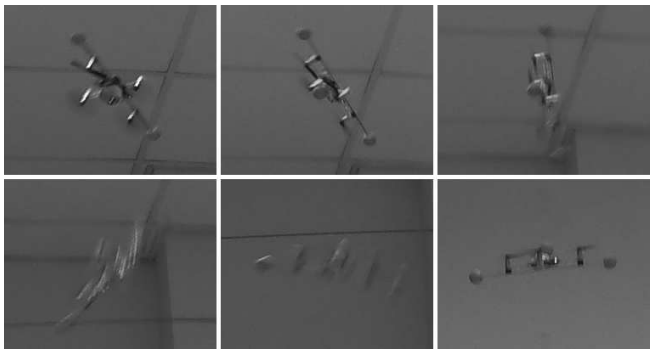

Fig. 5: Sequence of deployment maneuver and attitude recovery

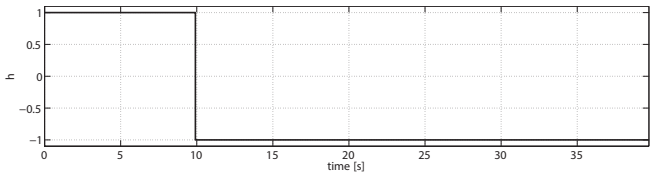

Fig. 6: Hybrid parameter h of Quadrotor n.1

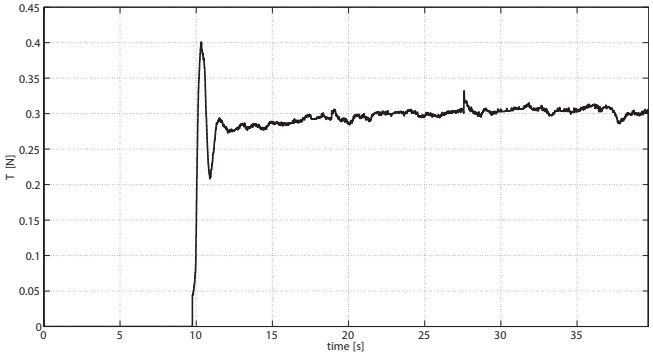

Fig. 7: Thrust command of Quadrotor n.1

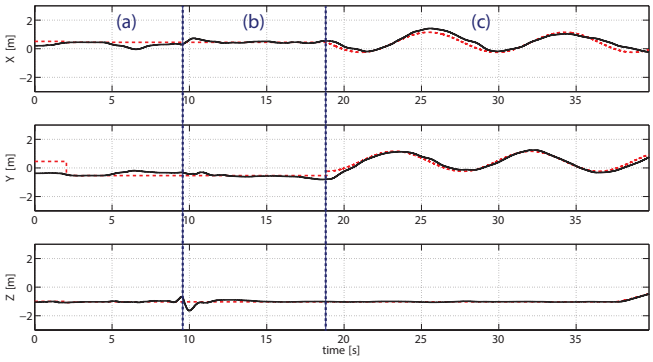

Fig. 8: Position reference and state of Quadrotor n.1. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

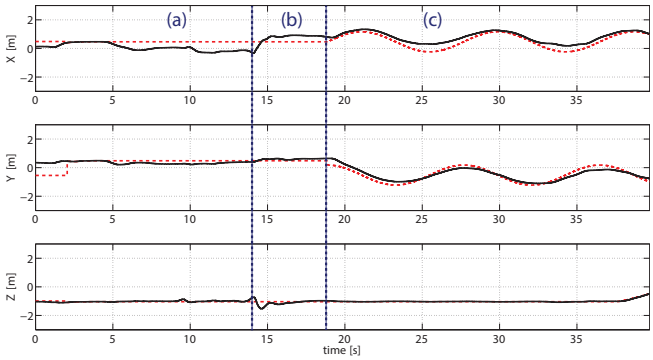

Fig. 9: Position reference and state of Quadrotor n.2. (a): deployment phase. (b): recovery and stabilization phase. (c): trajectory tracking phase.

coordination algorithms.

REFERENCES

[1] R. Mahony, V. Kumar, and P. Corke. Multirotor aerial vehicles. modeling, estimation and control of quadrotors. *Robotics Automation Magazine*, 19(3):20–32, 2012.

[2] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a large quadrotor robot. *Control Eng. Pract.*, 18(7):691–699, 2010.

[3] S. Bouabdallah and R. Siegwart. *Advances in Unmanned Aerial Vehicles*, chapter Chapter 6: Design and Control of a Miniature Quadrotor, pages 171–210. Springer Press, Feb. 2007.

[4] R. Cunha, D. Cabecinhas, and C. Silvestre. Nonlinear trajectory tracking control of a quadrotor vehicle. In *Proc. European Control Conference*, 2009.

[5] A. Bachrach, R. He, and N. Roy. Autonomous flight in unknown indoor environments. *International Journal of Micro Air Vehicles*, 1(4):217–228, 2009.

[6] M. Fumagalli, R. Naldi, A Macchelli, F. Forte, AQ.L. Keemink, S. Stramigioli, R. Carloni, and L. Marconi. Developing an aerial manipulator prototype: Physical interaction with the environment. *Robotics Automation Magazine, IEEE*, 21(3):41–50, Sept 2014.

[7] S. Lupashin, A. Schollig, M. Hehn, and R. D'Andrea. The flying machine arena as of 2010. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2970–2971, 2011.

[8] Alex Kushleyev, Daniel Mellinger, Caitlin Powers, and Vijay Kumar. Towards a swarm of agile micro quadrotors. *Autonomous Robots*, 35(4):287–300, 2013.

[9] The Crazyflie nano quadrotor. `http://www.bitcraze.se/crazyflie/`.

[10] R. Naldi, M. Furci, R.G.. Sanfelice, and L. Marconi. Global trajectory tracking for underactuated vtol aerial vehicles using a cascade control paradigm. In *Proceedings of IEEE Conference on Decision and Control*, Florence, IT, 2013.

[11] S.B. Bhat and D.S. Bernstein. A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon. *System & Control Letters*, 39:63–70, 1999.

[12] M.D. Shuster. A survey of attitude representation. *The Journal of the Astronautical Sciences*, 41(4):439–517, December 1993.

[13] E. Feron and E.N. Johnson. Aerial robotics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 1009–1027. Springer, 2008.

[14] R. Mahony, T. Hamel, and J.-M. Pfimlin. Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions of Automatic Control*, 53(5):1203–1218, 2008.

[15] Optitrack motion tracking system. `www.naturalpoint.com/optitrack`.

[16] Crazyflie control architecture code. `https://sites.google.com/site/robertonaldi/research-interests/aerial-robotics`.

[17] M. Hua, T. Hamel, P. Morin, and C. Samson. Introduction to feedback control of underactuated VTOL vehicles. *IEEE Control Systems Magazine*, 33(2):61–75, February 2013.

[18] C. Mayhew, R. Sanfelice, and A. Teel. On path-lifting mechanisms and unwinding in quaternion-based attitude control. *IEEE Transactions on Automatic Control*, 58(5):1179–1191.

[19] C.G. Mayhew, R.G. Sanfelice, and A.R. Teel. Quaternion-based hybrid controller for robust global attitude tracking. *IEEE Transactions on Automatic Control*, 56(11):2555–2566, November 2011.

[20] Video of icuas experiment on sherpa youtube channel. `https://www.youtube.com/watch?v=oreVFqRntXY`.