Control Engineering Practice 61 (2017) 93–111

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Indoor low-cost localization system for controlling aerial robots

Milton C.P. Santos[a,*], Lucas V. Santana[b], Alexandre S. Brandão[c], Mário Sarcinelli-Filho[d], Ricardo Carelli[e]

[a] Federal Institute of Espírito Santo, Rodovia ES-080, Km 93, São João de Petrópolis, Santa Teresa - ES 29660-000, Brazil
[b] Federal Institute of Espírito Santo, Av. Filogónio Peixoto, 2220 - Aviso, Linhares - ES 29901-291, Brazil
[c] Federal University of Viçosa, Viçosa - Minas Gerais 36570-900, Brazil
[d] Department of Electrical Engineering, Federal University of Espírito Santo (UFES), Av. Fernando Ferrari, 514, 29075-910 Vitória - ES, Brazil
[e] Institute of Automatics (INAUT), National University of San Juan (UNSJ), Av. San Martin Oeste 1109, 5400 San Juan, Argentina

ARTICLE INFO

Keywords:
UAV
Path-following
Positioning
Fusion filter
RGB-D sensor
Indoor localization

ABSTRACT

This paper presents a low-cost localization system to guide an Unmanned Aerial Vehicle (UAV) in indoor flights, considering an environment with invariant texture and typical indoor illumination. The first contribution of the paper is the proposal of a system to estimate the position and orientation of the UAV, through a multi-sensor fusion scheme, dealing with data provided by a RGB-D sensor, an inertial measurement unit (IMU), an ultrasonic sensor and optical flow-based velocity estimates. A second contribution of the paper is the proposal of a high-level control system to guide the UAV in path-following tasks, involving two controllers: a kinematic one, responsible for generating reference velocities for the vehicle, and a PD one, responsible for tracking such reference velocities, thus characterizing a cascade controller. Experiments with such a localization and control systems, during which abrupt disturbances are applied, were carried out to check the effectiveness of the developed capture and control systems, whose results validate the proposed framework.

1. Introduction

In the last few years there have been an increasing development of rotorcraft Unmanned Aerial Vehicles (UAV), motivated by their capability to fly in indoor and outdoor environments (Ferrick, Fish, Venator, & Lee, 2012) and their advantages over other flying machines, such as the possibility of to take off and land in the vertical, to hover, to move ahead and aside, and to change its direction of flight and to stop their motion abruptly (Pedro Castillo Garcia & Lozano, 2005). These features allow rotorcraft machines, mainly the smaller ones, to fly in indoor environments, such as offices and labs (Tournier, Valenti, How, & Feron, 2006). Another meaningful aspect associated to rotorcraft UAVs, still in terms of navigation, is that they are more versatile than ground vehicles, for making possible to get a global view of the workspace, what is fundamental in tasks like surveillance or inspection, for instance.

Traditional navigation systems based on wireless-transmitted information, such as Global Positioning System (GPS), are widely used to ensure a self-positioning task. However, indoor environments remain inaccessible to external positioning systems, limiting the navigation ability of the satellite-based GPS systems (Flores Colunga, Zhuo, Lozano, & Castillo, 2014). Most indoor applications use commercially available devices with infrared light to localize the UAV through computer vision, as the VICON system (http://www.vicon.com) and the CODA system (http://www.codamotion.com). The ETH Zurich Flying Machine Arena is an example of an indoor research space built specifically for the study of autonomous systems and aerial robotics (Lupashin, Schollig, Hehn, & D'Andrea, 2011; Lupashin et al., 2014). There a VICON system generates position data for small quadrotors accomplishing different tasks (Hehn & D'Andrea, 2011; Hehn, Ritz, & D'Andrea, 2012; Lupashin & D'Andrea, 2011; Schoellig, Siegel, Augugliaro, & D'Andrea, 2014; Willmann et al., 2012). Another famous example is the GRASP Laboratory (Michael, Mellinger, Lindsey, & Kumar, 2010), where quadrotors fly precisely through narrow gaps, automatically perch on inverted surfaces, and execute aggressive formation flights (Mellinger, Michael, & Kumar, 2014; Turpin, Michael, & Kumar, 2012).

However, the VICON system can be much expensive. An alternative is to use visual markers, such as in Santana, Brandão, and Sarcinelli-Filho (2016), where the position and orientation of the UAV are estimated through the extraction of features from images of known markers in the environment, using a camera onboard the UAV. Other

* Corresponding author.
E-mail addresses: milton.santos@ifes.edu.br (M.C.P. Santos), lucas@ifes.edu.br (L.V. Santana), alexandre.brandao@ufv.br (A.S. Brandão), mario.sarcinelli@ufes.br (M. Sarcinelli-Filho), rcarelli@inaut.unsj.edu.ar (R. Carelli).

http://dx.doi.org/10.1016/j.conengprac.2017.01.011
Received 30 April 2016; Received in revised form 23 January 2017; Accepted 24 January 2017
0967-0661/ Published by Elsevier Ltd.

studies, in turn, propose similar solutions, although using different camera arrangements, such as monocular vision (Weiss et al., 2013), stereo vision (Acgtelik, Bachrach, He, Prentice, & Roy, 2009) and RGB-D (Flores Colunga et al., 2014; Huang et al., 2011). Despite being low-cost solutions, these equipments have disadvantages, such as the sensitivity to light changes, the high processing cost, and the requirement of external markers.

An interesting approach was recently adopted in Huang et al. (2011), where a method that performs the fusion of data provided by a RGB camera with a depth map is proposed for mapping tasks. Following this approach, in order to reduce costs and not to require markers in the environment, this paper proposes a new low-cost localization platform to carry out experiments with UAVs in indoor environments.

The first results with this platform are in Santos, Santana, Martins, Brandão, and Sarcinelli-Filho (2015), where it is presented a method based on polynomial functions to localize the UAV in the environment. In such work a simple algorithm to guide the UAV to accomplish path-following tasks is also proposed, based on a waypoint navigation approach. An imitation is that the UAV orientation is not controlled. As a sequence of such work, after detecting the UAV and estimating its 3D position (x, y, z), a strategy for obstacle avoidance was developed, as well as a method to allow tracking the UAV, using position provided by a fusion filter (Santos, Santana, Brandao, & Sarcinelli-Filho, 2015).

This paper presents the final version of the aforementioned low cost platform, discussing all the details of the proposed localization system. It is shown how to get the UAV states (position and orientation), through a fusion scheme that considers data provided by different sensors, as well as how the model used to design a controller to guide the UAV (feedback linearization through inverse dynamics is the control strategy adopted) is identified. An algorithm to supervise the orientation of the vehicle is also presented, and a solution for the orientation drift is proposed. Besides this, a comparison is made between the proposed UAV detection algorithm and the method of Hu (1962). With this new platform to localize the UAV, it is possible to perform different tasks, considering different applications, and validate controllers as well. Thus, the first meaningful contribution of this article is the development of a low-cost, simple and efficient platform to estimate the UAV localization in an indoor environment, independently of illumination and visual markers.

In the sequel, the paper also proposes a controller to guide an UAV in path-following tasks, which is validated using the proposed localization framework. Regarding a path-following controller, the main objective is to make the "cross-track" error (ρ), the distance from the UAV to the path, and the heading error $|\psi_d - \psi|$ ($|\cdot|$ means absolute value) as close to zero as possible (to get $\rho \to 0$ and $|\psi_d - \psi| \to 0$ along the navigation). Nonlinear control techniques are popular for path-following applications (Sujit, Saripalli, & Sousa, 2014). In this case, good estimates of the position (x, y, z) and orientation (ψ) of the UAV are essential to guide it along the desired path (Chen, Chang, & Agate, 2013). A current problem in the development of this specific navigation strategy is that the robot should keep a fixed velocity along the path being followed, with the possibility of a null velocity, to allow the UAV to hover over the path being followed, if desired. Thus, path-following can be a smooth convergent task (Aguiar, Hespanha, & Kokotovic, 2005), ideal for applications involving capturing images of the working environment, for instance.

However, when the path-following controllers are based only on the robot kinematics, changes in the velocity of the vehicle is either not allowed (Rhee, Park, & Ryoo, 2010) or limited (Nelson, Barber, McLain, & Beard, 2007; Park, Deyst, & How, 2007). Therefore, the control objective, namely to keep the path error close to zero when the vehicle travels at a fixed velocity along the path, is not well accomplished (the velocity change demands a nonzero acceleration time, thus resulting in a difference between the velocity commanded by the kinematic controller and the velocity effectively developed by the UAV). In such a context, this paper proposes to add a simple dynamic controller in cascade with the kinematic one, to guide the UAV to perform path-following tasks as well as positioning tasks (in this case by setting the velocity of the vehicle to zero), which is its second meaningful contribution. Notice that this second class of tasks is perfectly compatible with rotorcraft UAVs, which are able to hover in a predefined 3D position.

Therefore, this paper proposes an indoor and low-cost platform to validate controllers based on different navigation strategies. Such platform includes a RGB-D system, which detects and estimates the posture and position of different types of objects. Commonly, this detection is performed using Hu (1962) moments. However, a new approach based on polynomial functions is here proposed. Both approaches are compared, in the specific case of detecting a Parrot Ar.Drone 2.0 quadrotor. After estimating the UAV position and orientation, based on the data provided by the RGB-D sensor, a multi-sensor fusion approach, using the information provided by the RGB-D system and the information provided by the sensors onboard the vehicle, is proposed, to decrease possible drift errors embedded in the measurements provided by inertial navigation systems (INS). Such a data fusion scheme is implemented in two steps. The first one is performed by the autopilot onboard the Parrot Ar.Drone 2.0, and the data outputted is the fusion of the information provided by an IMU, an ultrasonic sensor that gives information about the altitude of the vehicle and velocity estimates in the directions x and y. The second data fusion step involves the fusion of the result of the first data fusion step and the information provided by the RGB-D sensor. In addition, a cascade controller is designed to execute two navigation tasks, namely path-following and positioning.

To address such topics, the paper is hereinafter split as follows: Section 2 presents the AR.Drone quadrotor, the UAV used in this work, whereas Section 3 presents the indoor platform, showing how to detect the UAV and the necessary conversions to calculate its 3D position and orientation. The method designed to perform the sensory data fusion to improve the estimation of the position and velocities of the UAV is shown in Section 4. In the sequel, Section 5 presents the nonlinear controller proposed to control the position of the vehicle, using the sensory data delivered by the estimation subsystem proposed. Finally, experimental results obtained using the proposed data capture and position-estimate subsystem and the proposed controller in connection with a Parrot Ar.Drone 2.0 Power Edition® quadrotor are presented in Section 6, which is followed by some conclusions.

2. The AR.Drone quadrotor

The experimental rotorcraft machine chosen to validate our proposals is the Parrot AR.Drone 2.0 Power Edition quadrotor, which is shown in Fig. 1.

It is an autonomous aerial vehicle commercialized as a hi-tech toy,

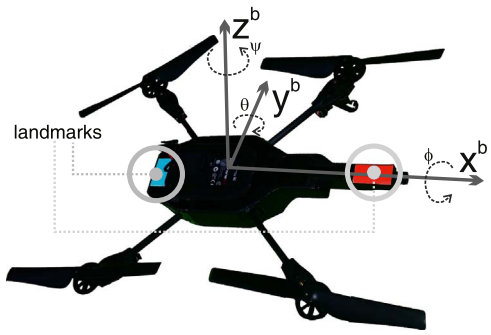

Fig. 1. The AR.Drone 2.0 Power Edition quadrotor and the body coordinate system $\langle b \rangle$. The visual markers are used to estimate its orientation (ψ). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

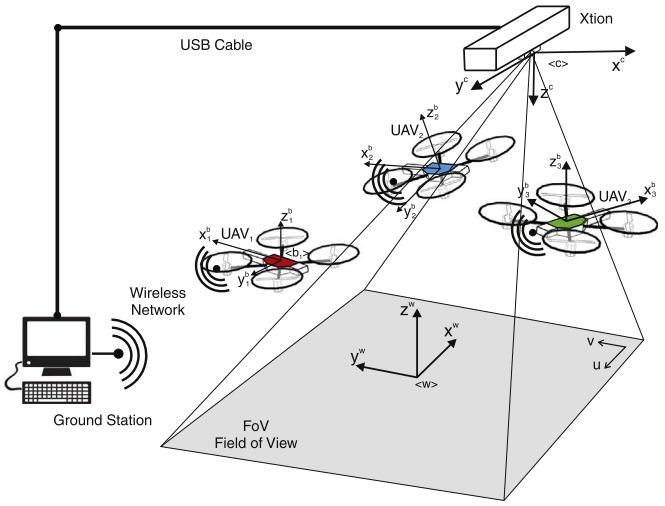

Fig. 2. Three-dimensional sketch of the low-cost indoor platform here proposed.

originally designed to be controlled using smartphones or tablets, via Wi-Fi network and specific communication protocols. It is equipped with two embedded boards (Santana, Brandão, Filho, & Carelli, 2014). The first one is the sensor board, and contains a set of sensors, such as an ultrasonic sensor, a barometric sensor and an inertial sensor (IMU) with accelerometers, gyroscopes, and magnetometers. Such IMU will be used in this specific work to measure the UAV velocities (Section 4). The second one, labeled the main board, is a processing unit based on an ARM Cortex-A8 processor, with 1 GHz of clock frequency, running an embedded Linux operating system. This board manages the data coming from the sensor board, the video streamings coming from a frontal and a bottom cameras, and the wireless communication network of the UAV. It also includes an autopilot subsystem not accessible to the user, which stabilizes the vehicle and keep it hovering while waiting for velocity commands to move, and makes available velocity data and GPS coordinates (when a USB GPS module is attached to the onboard computer).

2.1. Mathematical model

The AR.Drone firmware takes into account aerodynamic effects to achieve flight stabilization, but developers cannot access the model parameters (The Navigation & Control Technology Inside the AR.Drone Micro UAV). However, as reported in Santana et al. (2014), Krajník, Vonásek, Fišer, and Faigl (2011), Engel, Sturm, and Cremers (2012), Hernandez, Copot, De Keyser, Vlas, and Nascu (2013), and Falcon, Barreiro, and Cacho (2013), it is possible to analyze the AR.Drone response based on its internal controllers, instead of dealing with its complex dynamics. The most important of such platform is the relation between the control actions and the quadrotor position. This relation is approximated by a linear model of four states (Santana et al., 2014). Thus, it is possible to represent the dynamics of the AR.Drone as

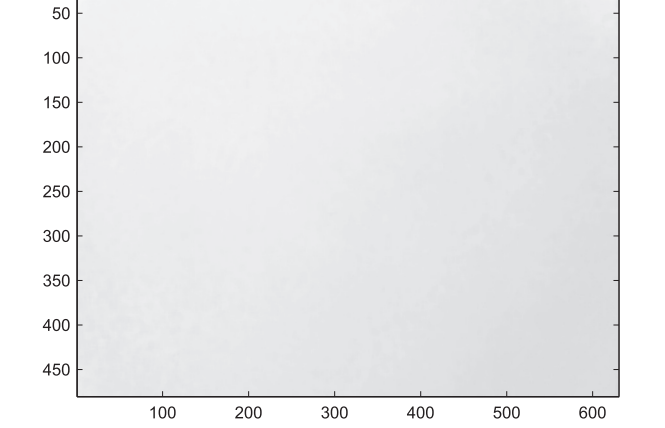

Fig. 3. A depth image of the background.

$$\ddot{\mathbf{x}}^b = \mathbf{K}_u\mathbf{u} - \mathbf{K}_v\dot{\mathbf{x}}^b \tag{1}$$

where the superscript b means the system of coordinates attached to the UAV, $\mathbf{u}$ is the vector of normalized command signals, defined as $\mathbf{u} = [u_{\dot{x}}\ u_{\dot{y}}\ u_{\dot{z}}\ u_{\dot{\psi}}]$, whose entries are in the interval $[-1.0, +1.0]$. As for $\mathbf{K}_u = diag[k_1\ k_3\ k_5\ k_7]$ and $\mathbf{K}_v = diag[k_2\ k_4\ k_6\ k_8]$, they are diagonal positive defined gain matrices. Considering all the coordinate systems in Fig. 9 and (1), one can write

$$\begin{aligned}
\ddot{x}^b &= k_1u_{\dot{x}} - k_2\dot{x}^b \\
\ddot{y}^b &= k_3u_{\dot{y}} - k_4\dot{y}^b \\
\ddot{z}^b &= k_5u_{\dot{z}} - k_6\dot{z}^b \\
\ddot{\psi}^b &= k_7u_{\dot{\psi}} - k_8\dot{\psi}^b,
\end{aligned} \tag{2}$$

where $\ddot{x}^b$, $\ddot{y}^b$ and $\ddot{z}^b$ are the accelerations in the axes x^b, y^b and z^b, respectively, in the referential $\langle b\rangle$ and $\ddot{\psi}^b$ represents the angular acceleration related to the axis z. The signals $u_{\dot{x}}$, $u_{\dot{y}}$ and $u_{\dot{z}}$ are the input variables related to the axes x^b, y^b and z^b, respectively, while $u_{\dot{\psi}}$ is the input variable related to the angular movement around the axis z.

To get the complete model, it is only necessary to identify the parameters $k_1, k_2, \ldots, k_8$. The identification results for such parameters, considering the Parrot Ar.Drone 2.0 Power Edition quadrotor, are dealt with in Section 6.

To design the control law, it is necessary to rewrite (1), now considering the global reference system (referred to by the superscript w). Such a transformation is performed through the rotation matrix correspondent to the orientation ψ of the UAV. The reason is that the pitch and roll angles are limited to values close to zero degrees by the autopilot onboard the UAV, so that one can consider that there is no rotation associated to the axes x and y. As shown in Santana et al. (2014), the result is

$$\ddot{\mathbf{x}}^w = \mathbf{F}_1\mathbf{u} - \mathbf{F}_2\dot{\mathbf{x}}^b, \tag{3}$$

where

$$\mathbf{F}_1 = \begin{bmatrix} k_1\cos(\psi^b) & -k_3\sin(\psi^b) & 0 & 0 \\ k_1\sin(\psi^b) & k_3\cos(\psi^b) & 0 & 0 \\ 0 & 0 & k_5 & 0 \\ 0 & 0 & 0 & k_7 \end{bmatrix}, \quad \text{and} \tag{4}$$

$$\mathbf{F}_2 = \begin{bmatrix} k_2\cos(\psi^b) & -k_4\sin(\psi^b) & 0 & 0 \\ k_2\sin(\psi^b) & k_4\cos(\psi^b) & 0 & 0 \\ 0 & 0 & k_6 & 0 \\ 0 & 0 & 0 & k_8 \end{bmatrix}. \tag{5}$$

Replacing (5) in (3), the result is

$$\mathbf{u} = \begin{bmatrix} g_{(0,0)} & g_{(0,1)} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & g_{(1,2)} & g_{(1,3)} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddot{z}^w & \dot{z}^w & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \ddot{\psi}^w & \dot{\psi}^w \end{bmatrix}\boldsymbol{\theta}, \tag{6}$$

where

$$g_{(0,0)} = \ddot{x}^w\cos\psi + \ddot{y}^w\sin\psi \tag{7}$$

$$g_{(0,1)} = \dot{x}^w\cos\psi + \dot{y}^w\sin\psi \tag{8}$$

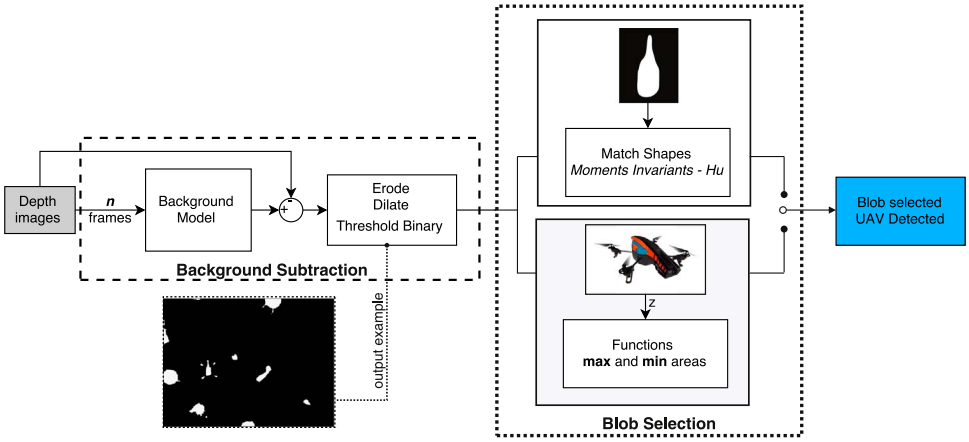

Fig. 4. Diagram representing how the UAV is detected in 2D images.

$$g_{(1,2)} = -\ddot{x}^w \sin\psi + \ddot{y}^w \cos\psi \tag{9}$$

$$g_{(1,3)} = -\dot{x}^w \sin\psi + \dot{y}^w \cos\psi \tag{10}$$

and

$$\boldsymbol{\theta} = \left[\frac{1}{k_1}\ \frac{k_2}{k_1}\ \frac{1}{k_3}\ \frac{k_4}{k_3}\ \frac{1}{k_5}\ \frac{k_6}{k_5}\ \frac{1}{k_7}\ \frac{k_8}{k_7}\right]^T. \tag{11}$$

Therefore, (6) can be rewritten as

$$\mathbf{u} = \mathbf{G}\boldsymbol{\theta}. \tag{12}$$

To perform the identification, experimental signals are applied and (12) is written for each sampling instant, thus getting the matrices

$$\mathbf{G}_E = \begin{bmatrix} \mathbf{G}_{E1}(0) \\ \vdots \\ \mathbf{G}_{E1}(n) \\ \mathbf{G}_{E2}(0) \\ \vdots \\ \mathbf{G}_{E2}(n) \end{bmatrix}, \quad \mathbf{u}_E = \begin{bmatrix} \mathbf{u}_{E1}(0) \\ \vdots \\ \mathbf{u}_{E1}(n) \\ \mathbf{u}_{E2}(0) \\ \vdots \\ \mathbf{u}_{E2}(n) \end{bmatrix}, \tag{13}$$

where n is the last instant of the experiment, $\mathbf{G}_{Ei}$ is a row vector composed of the i-th row of the matrix $\mathbf{G}_E(k)$ (matrix $\mathbf{G}_E$ in the instant k), $\mathbf{u}_{Ei}(k)$ is the value of the i-th row of the vector $\mathbf{u}_E(k)$ (vector $\mathbf{u}_E$ in the instant k) (sub-index E refers to the experimental data used for the identification). Thus, one has

$$\mathbf{G}_E\hat{\boldsymbol{\theta}} = \mathbf{u}_E. \tag{14}$$

Finally, the estimated vector $\hat{\boldsymbol{\theta}}$ of identified parameters is given by (Astrom & Wittenmark, 1994)

$$\hat{\boldsymbol{\theta}} = (\mathbf{G}_E^T\mathbf{G}_E)^{-1}\mathbf{G}_E^T\mathbf{u}_E, \tag{15}$$

using the least mean squares method.

3. Indoor platform

Some laboratories adopted the concept of intelligent spaces for mobile robots, as reported in Morioka and Hashimoto (2004), Steinhaus, Strand, and Dillmann (2007), and Losada, Mazo, Palazuelos, Pizarro, and Marrón (2010). Some of the technologies used for indoor localization achieve centimeter-level precision, such as ultra-wide band (UWB) (Ahn, Yu, & Lee, 2007), infrared light (Lupashin et al., 2011; Want et al., 1992), stereo vision systems (Elmezain, Al-Hamadi, & Michaelis, 2009) or computer vision systems (Lupashin et al., 2011; Rampinelli et al., 2014). However, all of such methods need a considerable investment for structuring the environment (Hazas, Scott, & Krumm, 2004).

Kinect© and Xtion Pro Live© have come out as low-cost sensors, in comparison with the aforementioned systems, with similar results (Regazzoni, de Vecchi, & Rizzi, 2014). In this work, it is proposed a simple and low-cost solution, using depth-image information provided by these sensors, to detect and localize objects in an indoor environment. Fig. 2 depicts the structure of the proposed platform.

The points of the depth image generated by the depth sensor indicate the distance from the sensor to objects in the navigating environment. Thus, the first step to detect the UAV in an indoor environment can be carried out through background subtraction (Piccardi, 2004). In short, such a method involves the acquisition of a sequence of depth images of the environment in the absence of the UAV, from which an average model of the background is generated. An example of a depth image of the background is shown in Fig. 3, where the pixels contain the values of the depth correspondent to the objects in the background. As one can see in such a figure, the depth values are high (gray levels close to white), meaning objects far from the depth sensor installed in the roof. Later, after introducing the UAV in the environment, the subtraction of the background is performed every time a new image is acquired, so that what remains in the image much likely contains the UAV. Then, the resulting image is binarized, resulting in a final image containing some blob regions (see image in Fig. 4, for instance), which correspond to the objects that were not included in the background model (UAV included). In the sequel, the blobs are compared to a model, and those blobs similar to the model are selected to the next stage, which uses polynomial functions relating the UAV altitude to the correspondent blob area to decide if a blob in the image is or is not the UAV (this is because the area of the blob correspondent to the UAV is dependent on how far or how close to the sensor installed in the roof the UAV is).

In the next subsections different methods for detecting the UAV in the binarized image will be presented. In Section 3.1.1 the method of Hu (1962), with its moment invariants, is adopted to detect and classify the UAV. Such a strategy is commonly adopted in robotic applications to detect and classify objects. As for our approach to detect the UAV in

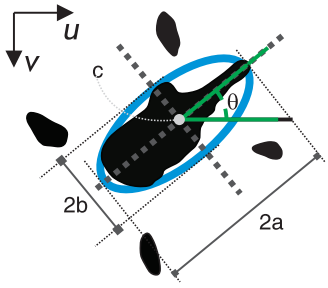

Fig. 5. Image blob and the respective inertial ellipse. The major axis indicates the UAV orientation.

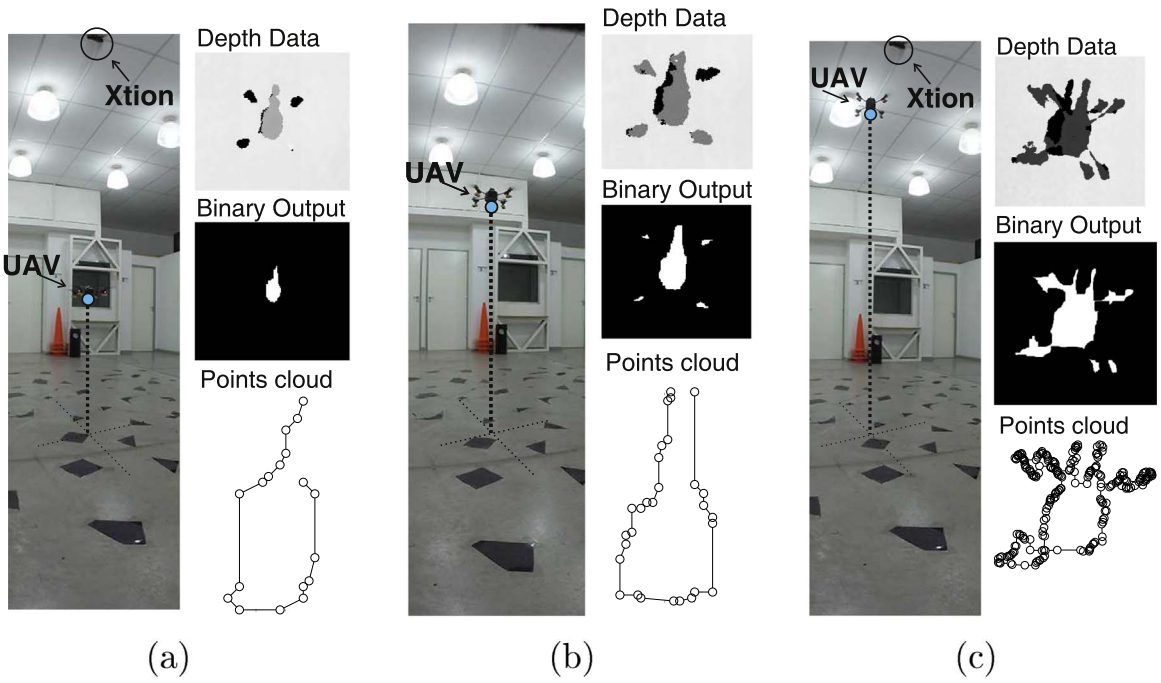

Fig. 6. Images correspondent to different instants during a flight. In (a), the UAV is at 0.925 m of altitude, the contour of the blob has 20 points and the total points of the blob is 180. In (b), the drone is at the altitude of 1.60 m, with 31 points in the contour and a total of 723 points. In (c), the UAV is at an altitude of 2.6 m, the contour has 220 points and the blob has 14 075 points in total.

the binarized image, which is based only in polynomial functions relating the altitude of the UAV and the correspondent area of the blob in the image, it is discussed in Section 3.1.2.

3.1. UAV detection

3.1.1. Using moment invariants

Moment invariants (Hu, 1962) have been frequently used as features for image processing (Mercimek, Gulez, & Mumcu, 2005), shape recognition (Rothe, Susse, & Voss, 1996) and classification. They can provide characteristics of an object that uniquely represent its shape. Invariant shape recognition is performed by classification in the multidimensional moment invariant feature space.

Such invariant moments are computed over the shape boundary and its interior region. Because of these characteristics, they are used to compare an image (A) to a pattern (B), using

$$I(A, B) = \max_{i=1\ldots7} \frac{\text{sgn}(\phi_i^A)\log(\phi_i^A) - \text{sgn}(\phi_i^B)\log(\phi_i^B)}{|\text{sgn}(\phi_i^A)\log(\phi_i^A)|}, \tag{16}$$

where sgn is a simple function that results ± 1 when the values are positive or negative. As for ϕ_i, $i = 1, \ldots, 7$, they are seven invariant moments set out by Hu (1962), which are

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02}\\
\phi_2 &= (\eta_{20} + \eta_{02})^2 + 4\eta_{11}^2\\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (\eta_{03} - 3\eta_{21})^2\\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{03} + \eta_{21})^2\\
\phi_5 &= (3\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]\\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]\\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] +\\
&\quad 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] +\\
&\quad (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03}) \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{30})^2],
\end{aligned} \tag{17}$$

where η_{pq} are the relative moments, the mathematical moment of order $p + q$, which are calculated using the equation for central moments, whose value is defined as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \tag{18}$$

where the normalization factor is $\gamma = (p + \frac{q}{2}) + 1$.

The idea is now to use the value of $I(A, B)$ to classify the blobs in an image, correspondent to different objects in the environment, as a way to identify where the UAV is.

3.1.2. Using polynomial functions

In the particular case of this paper, one wants to identify the UAV in an environment containing some objects. Hence, it is necessary to choose the blob correspondent to it in the image so far processed. A simple statistical method that correlates the altitude of the UAV with its effective area in the binarized image is adopted. Fig. 4 shows the diagram for UAV detection, based on depth images.

A manually controlled flight is performed, and several binarized images corresponding to the quadrotor are collected, together with its altitudes. From such information the minimum and maximum values of the UAV area are determined as a function of the altitude. The procedure is repeated considering that the UAV may fly with or without its indoor hull. For each set of blob area/altitude measurements two polynomial functions were generated, relating the maximum and minimum areas that the UAV occupies in the image as a function of its altitude. In this case, one can intuitively notice that the area correspondent to the blob that represents the UAV in an image increase when it is closer to the camera in the roof (farer from the ground) and vice versa.

The functions $g_1(z)$ and $g_2(z)$ represent the maximum and minimum areas of the UAV as a function of its altitude $z(\in \mathbb{R}^+)$. Hence, during takeoff the proposed capture system provides the altitude z_{UAV} correspondent to an object in the navigation space through a depth map, as well as the area that the correspondent blob occupies in the binarized image. From such data the system checks the rules

If $blob_{area} > g_1(z_{UAV})$ then UAV not detected

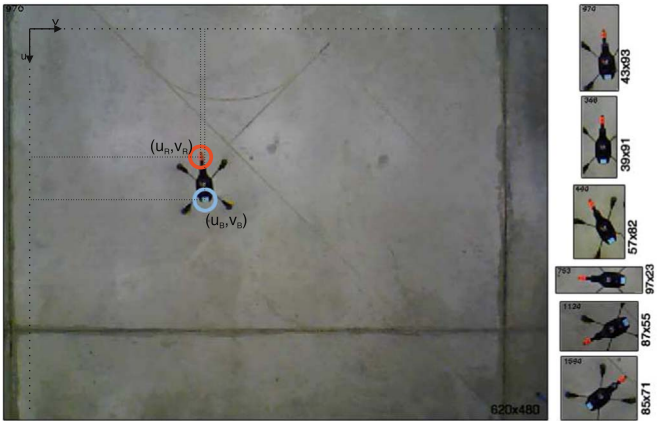

Fig. 7. RGB sensor image and ROI to reduce the search window to find the markers. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

else If $blob_{area} < g_1(z_{UAV})$ and
$blob_{area} < g_2(z_{UAV})$ then UAV detected

else UAV not detected

After determining if the UAV is in the captured image, the correspondent polynomial functions are used to select the values of maximum and minimum areas associated to the UAV, according to its altitude (z_{UAV}). Thus, the detection of the UAV is performed simply checking if the blob area of the object present in the binarized image is in between the maximum and minimum values for the UAV area, whenever a new image is captured. This way, even when more than one blob (more than one object) is detected in an image, the capture system is able to detect the UAV, supposing that it is in the field of view of the camera.

3.2. Estimating the UAV position and orientation

After detecting the UAV, its position and orientation can be calculated. To perform such calculations, one needs first to transform the 2D image coordinates into 3D coordinates. Such 3D coordinates, x_c^w, y_c^w and z_c^w, will be given by the Xtion Pro Live camera (c) in the world referential (w). In order to do this, it is important to understand some geometric transformations. They are included in

$$z_{ij}\begin{bmatrix} u_i \\ v_j \\ 1 \end{bmatrix} = \begin{bmatrix} fs_u & fs_\theta & o_u \\ 0 & fs_v & o_v \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_c^w \\ y_c^w \\ z_c^w \\ 1 \end{bmatrix}, \tag{19}$$

where z_{ij} is the depth at the pixel in the position (i, j), f is the focal length, s_u and s_v are scaling factors, (o_u, o_v) is the displacement of the center of the image axes, and $\mathbf{R}$ and $\mathbf{T}$ are translation and rotation matrices, respectively. Finally, x_c^w, y_c^w and z_c^w are the coordinates of the reference point in the world (see Ma, Soatto, Kosecka, & Sastry, 2001; Santos, Santana, Martins et al., 2015 for more details).

Basically, notice that (19) depends on intrinsic and extrinsic parameters of the camera, and the depth values of the point (u, v) that are located at the row i and column j of the depth map.

In this work the position and the orientation of the UAV are determined by the RGB-D camera. The position of the UAV is calculated as the center of the blob representing the UAV, detected from the depth images using background substraction. On the other hand, the method for estimating the orientation of the UAV is based on the detection of an inertial ellipse, which also uses the blob that represents the UAV in the depth image.

The ellipse used (inertial ellipse), as shown in Chaumette (2004), is centered on the centroid c of the object and its axes, $2a$ and $2b$, are the lines passing through c, such that the central moments of second order are maximums and minimums. The blob and the ellipse representing it can be seen in Fig. 5.

The dimensions of the semi-axes of the ellipse (a, b) and the orientation (θ) of the greater semi-axis with respect to the x-axis of the image plane can be obtained using the central moments (represented by u) of zero, first and second orders, as follows

$$a = 2\sqrt{\frac{A}{\mu_{00}}}, \tag{20}$$

$$b = 2\sqrt{\frac{B}{\mu_{00}}}, \quad \text{and} \tag{21}$$

$$\theta = \frac{1}{2}\arctan\left(\frac{\mu_{11}}{\mu_{20} - \mu_{02}}\right), \tag{22}$$

where

$$A = \frac{(\mu_{20} + \mu_{02}) + [(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2]}{2}, \quad \text{and} \tag{23}$$

$$B = \frac{(\mu_{20} + \mu_{02}) - [(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2]}{2}. \tag{24}$$

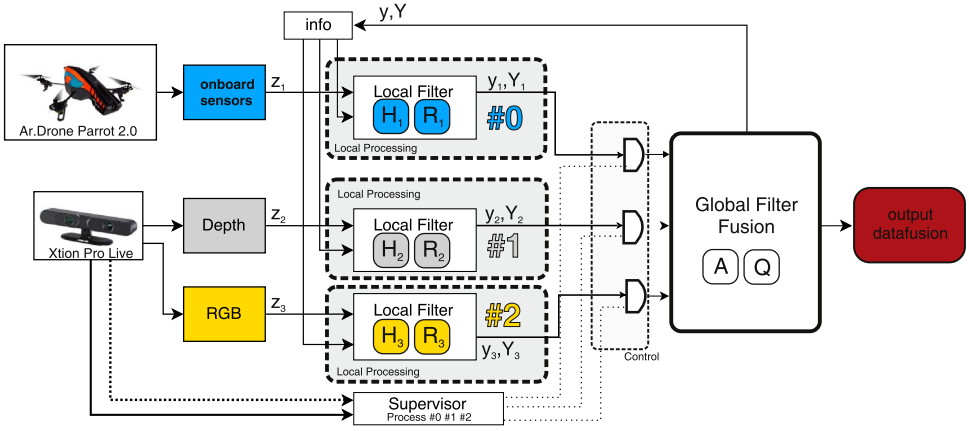

Fig. 8. The Decentralized Information Filter proposed.

The integration of the central moment in a digital image results in

$$\mu_n = \sum (x - \mu)^n f(x). \tag{25}$$

In this work the central moments used, μ_{20} and μ_{02}, are the central moments of second order of x and y, i.e., the variance of each one of the variables. The variances of x and y are, respectively,

$$\mu_{20} = \sigma_x^2 = \sum (x - \mu)^2 f(x) \tag{26}$$

$$\mu_{02} = \sigma_y^2 = \sum (y - \mu)^2 f(y). \tag{27}$$

The UAV blob in the binarized image, as shown in Fig. 5, suggests that the extraction of the orientation could be made through feature analysis of the frontal extremity of the blob with the object center. However, the calculation of the UAV orientation through the inertial ellipse is faster, since it does not use all the points of the blob contour. It is important to highlight that the blob area is related to the UAV altitude in the environment. Fig. 6 presents an example of the number of points necessary to determine the UAV orientation through the calculation of the orientation of the inertial ellipse.

In addition to the ellipse method, it was implemented a method to track colored visual markers attached to the drone to calculate the orientation ψ of the UAV, in order to validate the results obtained with the ellipse method (ground truth).

To estimate ψ using the RGB information, two visual markers were added to the drone, as shown in Fig. 1. Notice that the use of color information to detect objects in motion is a common procedure in robotics. The algorithms to track the markers were developed based on the segmentation of the HSV channels, used to represent the image. To accelerate the searching for the markers, a region of interest (ROI) is determined, so that instead of searching for the whole $u \times v$ image one calculates a search rectangle of dimension $m \times n$, where $m \ll u$ and $n \ll v$. To determine such a ROI, it is used as the midpoint of the blob representing the UAV (c_u, c_v) and, according to the contour of the blob, the size of the ROI is determined. Fig. 7 shows the image obtained by the RGB camera and a few frames with the ROI in which the search is performed.

Therefore, one obtains the values (u_R, v_R) and (u_B, v_B), which are the centers of the blobs representing the red and blue markers, respectively. To determine the UAV orientation angle it is adopted the relation

$$\theta_{RGB} = \tan^{-1}\left(\frac{u_R - u_B}{v_R - v_B}\right). \tag{28}$$

The angles θ and θ_{RGB} have as reference the 2D image space, and thus $[\theta, \theta_{RGB}] \in [-90°, 90°]$. For that, a function is designed so that the angle calculated from the image can follow the UAV movement, i.e. $\psi_{eli} = f_c^w(\theta)$ and $\psi_{RGB} = f_c^w(\theta_{RGB})$. Such a function, $f_c^w(\cdot)$, is shown in Algorithm 1.

Algorithm 1. Used to convert the angle in the 2D image space (c) to the real world (w).

Precondition: ψ_{k-1} and θ are the angle converted in the previous step and the input angle to be converted, respectively.

function CONVERTTO$_c^w(\psi_{k-1}, \theta)$
  var $\psi_k \leftarrow -\theta$
  var $s \leftarrow \text{sgn}(\psi_k)$
  var $\delta \leftarrow \theta - \psi_{k-1}$
  if $|\delta| \geq \delta_{max}$ **then**
    $\psi_k \leftarrow \psi_{k-1} + \delta - (s)*\pi$
  else
    $\psi_k \leftarrow \psi_{k-1} + \delta$
  end if
  return ψ_k
end function

4. Decentralized estimation through a multi-sensor fusion engine

Through using the IMU and the ultrasonic sensor onboard the vehicle and an external RGB-D sensor, it is possible to estimate the 3D position and the orientation of the UAV relative to the real world. Since each sensor is prone to errors, sensory data fusion is adopted in order to improve such data information (Mutambara, 1998). Thus, position coordinates and orientation (x, y, z, ψ) will be estimated through the implementation of an Information Filter (IF) and a Decentralized Information Filter (DIF) (Saadeddin, Abdel-Hafez, & Jarrah, 2013). The IF was used because it is a recursive algorithm that can produce an optimal estimation of linear systems and estimates the states faster than the Kalman Filter (KF) (Assimakis, Adam, & Douladiris, 2012).

The Information Filter is essentially a KF expressed in terms of information measures (Assimakis et al., 2012; Freire, Filho, Filho, & Carelli, 2004) on the states of interest, rather than state estimates and their associated covariances (Mutambara, 1998). The equations correspondent to the Information Filter are

$$\mathbf{Y}_k = \mathbf{P}_{(k|k-1)}^{-1} \tag{29}$$

and

$$\hat{\mathbf{y}}_k = \mathbf{P}_{(k|k-1)}^{-1}\hat{x}_{(k|k-1)} = \mathbf{Y}_k\hat{x}_{(k|k-1)}, \tag{30}$$

which give, respectively, the State Information matrix and the State Information vector. As for $\mathbf{P}$, it is the error covariance matrix, whereas x is the state estimate. Then, the equations correspondent to the IF prediction step are

$$\hat{\mathbf{y}}_{(k|k-1)} = \mathbf{L}_k\hat{\mathbf{y}}_{(k-1|k-1)}, \quad \text{and}\mathbf{Y}_{(k|k-1)} = [\mathbf{A}_k\mathbf{Y}_{(k-1|k-1)}^{-1}\mathbf{A}_k^{\mathbf{T}} + \mathbf{Q}]^{-1}, \tag{31}$$

where $\mathbf{A}$ is a state transition matrix and $\mathbf{Q}$ is the matrix of process error covariance. As for the IF estimation equations, they are

$$\hat{\mathbf{y}}_{(k|k)} = \hat{\mathbf{y}}_{(k|k-1)} + \mathbf{i}_k, \quad \text{and}\mathbf{Y}_{(k|k)} = \mathbf{Y}_{(k|k-1)} + \mathbf{I}_k, \tag{32}$$

where

$$\mathbf{L}_k = \mathbf{Y}_{(k|k)}\mathbf{A}_k\mathbf{Y}_{(k|k-1)}^{-1}, \tag{33}$$

$$\mathbf{i}_k = \mathbf{H}_k^t\mathbf{R}_k^{-1}\mathbf{z}_k, \quad \text{and} \tag{34}$$

$$\mathbf{I}_k = \mathbf{H}_k^t\mathbf{R}_k^{-1}\mathbf{H}_k \tag{35}$$

are, respectively, the coefficient of information propagation, the contribution of state information and the Information matrix associated to each state.

4.1. Decentralized Information Filter

A decentralized filter consists of a network of filters, each one with its own processing unit. In such a system, the fusion occurs locally on each node, based on local information and information transmitted from neighbor filters. For a decentralized system of data fusion, the processing filter is a sensor node, which distributes observations and local information to other fusion nodes. Later, it assimilates this information and computes a local estimation, i.e., in the decentralized system, the local filter/node uses the information to generate a fused local output.

In this work, the information comes from two set of sensors, two of them onboard the vehicle (the IMU and an ultrasonic sensor, whose information is previously fused in the autopilot onboard the quadrotor used in the work, generating the output of the onboard sensors set), and the RGB-D sensor installed in the roof of the working environment. However, as the RGB-D sensor provides a depth map and an image as well, the depth map will be associated to a local filter and the image will be associated to another one, thus configuring a DIF with three local filters (the third one deals with the data provided by the

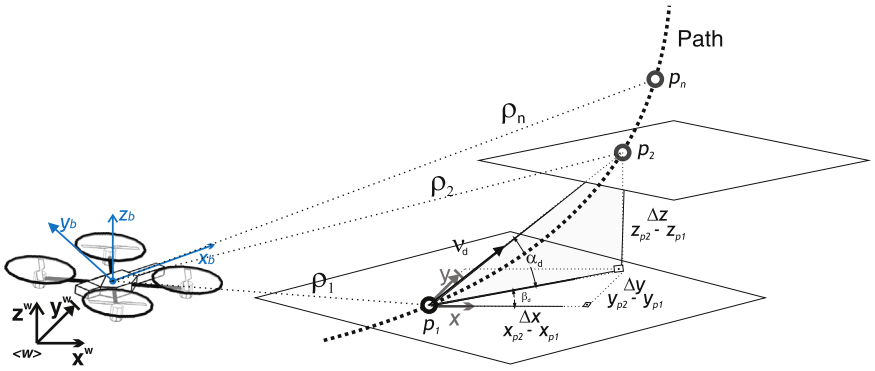

Fig. 9. Desired position along the path, with the UAV velocity tangent to it.

IMU and the ultrasonic sensor onboard the UAV), as it is shown in Fig. 8. Thus, the way the Decentralized Information Filter deals with the incoming data is as follows: in the instant k, for the i-th local filter one has

$$\hat{\mathbf{y}}_{i(k)} = \hat{\mathbf{y}}_{(k|k-1)} + \mathbf{i}_{i(k)} \quad \text{and} \tag{36}$$

$$\mathbf{Y}_{i(k)} = \mathbf{Y}_{(k|k-1)} + \mathbf{I}_{i(k)}, \tag{37}$$

whereas for the global filter one has

$$\hat{\mathbf{y}}_{(k)} = \sum_{i=1}^{n} \hat{\mathbf{y}}_{i(k)} - (n-1)\hat{\mathbf{y}}_{(k|k-1)} \quad \text{and} \tag{38}$$

$$\mathbf{Y}_{(k)} = \sum_{i=1}^{n} \mathbf{Y}_{i(k)} - (n-1)\mathbf{Y}_{(k|k-1)}, \tag{39}$$

where n is the number of local filters ($n = 3$ in this paper).

4.2. State estimation

The global filter is configured as

$$\mathbf{x}_{(k|k-1)} = \mathbf{A}\mathbf{x}_{(k-1|k-1)} + \mathbf{w}_k\mathbf{z}_{(k|k-1)} = \mathbf{H}\mathbf{x}_{(k-1|k-1)} + \boldsymbol{\nu}_k, \tag{40}$$

where $\mathbf{w}$ and $\boldsymbol{\nu}$ are the noise vectors, and $\mathbf{x}_{(k|k-1)}$ and $\mathbf{x}_{(k-1|k-1)}$ are state vectors. Such a filter relates the states at the previous time step $(k-1)$ to the states at the current time step (k). In this article, the states of the UAV, estimated by the DIF, are

$$\mathbf{x} = [x_b^w \;\; y_b^w \;\; z_b^w \;\; \psi_b^w \;\; \dot{x}_b^w \;\; \dot{y}_b^w \;\; \dot{z}_b^w \;\; \dot{\psi}_b^w]^T. \tag{41}$$

As aforementioned, the information used to estimate the UAV position comes from the IMU and the ultrasonic sensor onboard the UAV and from the RGB-D sensor, corresponding to three local IF, whose configuration matrices are

Filter-1, which receives data from the IMU. Its measurement vector and observation model are

$$\mathbf{z}_1 = [\dot{x}_b^w \;\; \dot{y}_b^w \;\; z^w \;\; \psi_b^w]^T,$$

$$\mathbf{H}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \text{ and}$$

$$\mathbf{R}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & \sigma^2_{\dot{x}_{z1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{\dot{y}_{z1}} & 0 & 0 \\ 0 & 0 & \sigma^2_{z_{z1}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\psi_{z1}} & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{42}$$

Filter-2, which receives data from the depth sensor. Its measurement vector and observation model are

$$\mathbf{z}_2 = [x_c^w \;\; y_c^w \;\; z_c^w \;\; \psi_{Eli}]^T$$

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \text{ and}$$

$$\mathbf{R}_2 = \begin{bmatrix} \sigma^2_{x_{z2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_{z2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{z_{z2}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\psi Eli_{z2}} & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{43}$$

Filter-3, which receives data from the RGB sensor. Its measurement vector and observation model are

$$\mathbf{z}_3 = [\psi_{RGB}]^T,$$

$$\mathbf{H}_3 = [0 \;\; 0 \;\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0], \text{ and}$$

$$\mathbf{R}_3 = \begin{bmatrix} 0 & 0 & 0 & \sigma^2_{\psi RGB_{z3}} & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{44}$$

Therefore, the fusion filter deals with data provided by three sensors. The choice of a decentralized filter, in this case, is because of the different acquisition times, when regarding the onboard IMU/ultrasonic sensor and the vision system. This way, although the processes are executed in the same computer, three processes were created, which are executed in parallel, to compensate for the delay in the data acquisition by the RGB-D sensor. Indeed, the data provided by the RGB camera and the depth map go through a digital processing for the extraction of features for the localization of the UAV and the estimation of its position and orientation. Thus, the delivery of the position/orientation data through the RGB-D sensor demands more time than the delivery time for the onboard sensors, in general. However, the data provided by the IMU/ultrasonic sensor onboard the UAV may have a delay caused by the communication network established between the ground station from where the UAV is controlled and the UAV. Effectively, such a ground station receives the information coming from the sensors onboard the vehicle, processes the data provided by the RGB-D sensor, runs the fusion filter to get the state estimates, runs the control algorithm and sends the control systems to the autopilot onboard the UAV.

These filters enable to fuse data coming from the sensors (three different data), resulting in an output data whose variance is smaller than the smallest variance associated to the data measured by each sensor (Santos, Santana, Brandao et al., 2015). In order to succeed, special attention should be paid to matrices $\mathbf{R}_i$. They store the weights relative to data coming from the onboard sensors and the RGB-D device. If the values of the entries of $\mathbf{R}_1$ are higher than those of the entries of $\mathbf{R}_2$, then data provided by the onboard IMU/ultrasonic sensor have a higher weight in the estimation than data provided by the RGB-D device. On the other hand, if the values of the entries of $\mathbf{R}_2$ are higher than those of the entries of $\mathbf{R}_1$, then data provided by the RGB-D

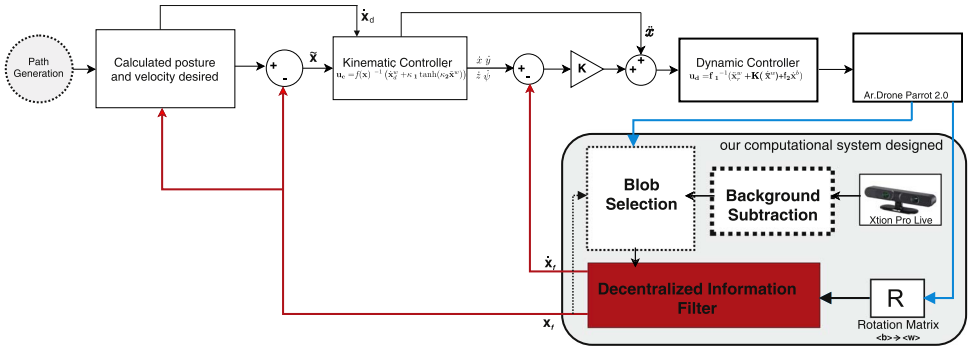

Fig. 10. The proposed control system architecture.

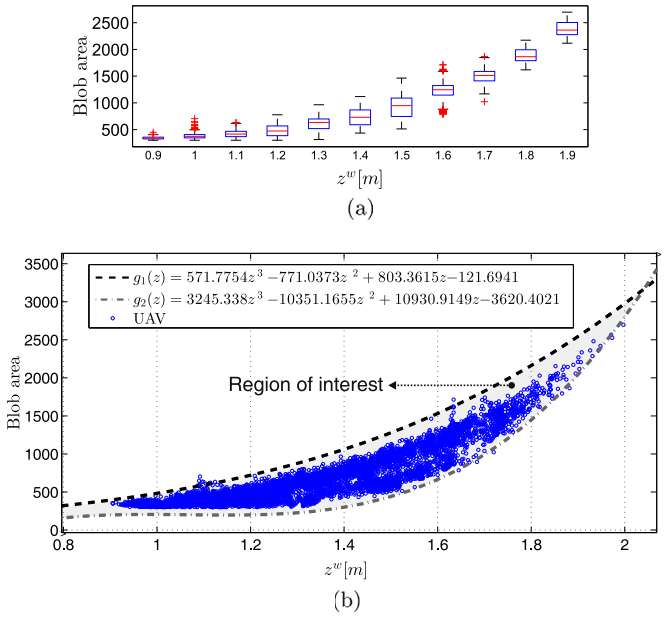

Fig. 11. (a) Data used to calculate the functions g_1 and g_2. (b) The polynomial functions and the region of interest to detect the UAV.

device contributes more to the estimate of the position and orientation of the UAV. Fig. 8 shows the diagram of the Decentralized Information Filter proposed.

The result outputted by the DIF is

$$\hat{\mathbf{x}}^w_{fusion} = [x^w_f \;\; y^w_f \;\; z^w_f \;\; \psi^w_f \;\; \dot{x}^w_f \;\; \dot{y}^w_f \;\; \dot{z}^w_f \;\; \dot{\psi}^w_f]. \tag{45}$$

Thus, the output of these filters will correspond to data whose variance is smaller than the smallest variance associated to the data measured by each sensor, which means a more reliable information.

5. The autonomous controller

In autonomous navigation tasks it is necessary to feedback the position of the aircraft to the controller. The more accurate the information delivered to the controller, the less the position errors will be. In this work the position information provided by the multi-sensor fusion engine described in the previous section is adopted, since the information the DIF provides is more accurate than the information provided either by the sensors onboard the vehicle or the RGB-D sensor individually. As the image processing can be a time consuming task, it is interesting to have a navigation algorithm without time restrictions. This is why path-following is the navigation task addressed in this paper: there is no restriction about the velocity of the UAV in following the prescribed path, in opposition to the case of trajectory tracking, a task in which time is fundamental. A sketch of the desired position along a path to be followed with the aircraft developing a velocity tangent to the path is shown in Fig. 9.

As for the proposed controller, a cascade control algorithm consisting of two subsystems is designed: a kinematic controller and a stabilizer for the dynamics of the aircraft. The kinematic controller, the first subsystem, whose function is to generate reference velocities for the UAV, is based on feedback linearization, adopting a saturation on the control signals when the position errors are big. By its turn, the dynamic controller acts compensating the dynamics of the vehicle, allowing a better tracking of the reference velocities generated by the kinematic controller.

In the following equations some indexes are used in connection with the variables involved. The superscript b is added to the values referred to the UAV body, while the superscript w is added to refer to the global reference system. As for the reference signal, the subscript r is used, whereas the subscript d is adopted for the desired position, velocity or acceleration.

5.1. Cascade control algorithm

To follow a given path requires the convergence to zero of the error between the current position of the UAV and the point of the path closest to it. To accomplish such a task, the aircraft should follow the path with any desired velocity.

The path is defined by a sequence of points $[p_1, p_2, \ldots, p_n]$, $n \in N^+$, as shown in Fig. 9. There ρ represents the distance between the aircraft and the point of the path that is closest to it. Such a distance is a function of the position (x^w_b, y^w_b, z^w_b) of the aircraft and the position of $[p_1, p_2, \ldots, p_n]$, $n \in N^+$, given by (x^w_n, y^w_n, z^w_n). As for the angles α and β, they represent the orientation of the desired velocities along the path.

Given a geometric path $\mathbf{p} \in \Re^3$ and a desired velocity $\mathbf{v}^w_d \in \Re$ it is

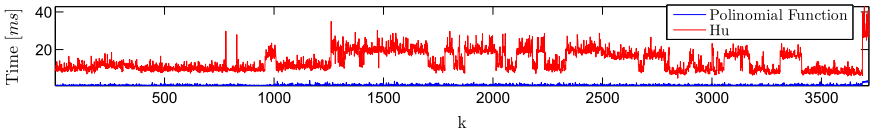

Fig. 12. Difference between time processing using the Moments of Hu and our polynomial functions to detect the UAV.

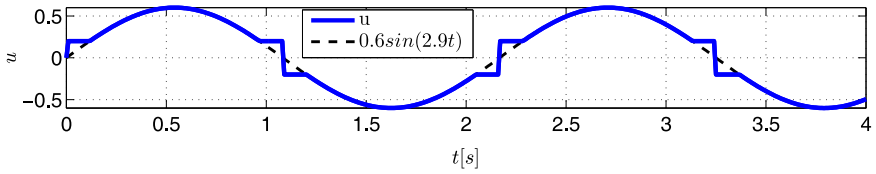

Fig. 13. Signal used as input for (2).

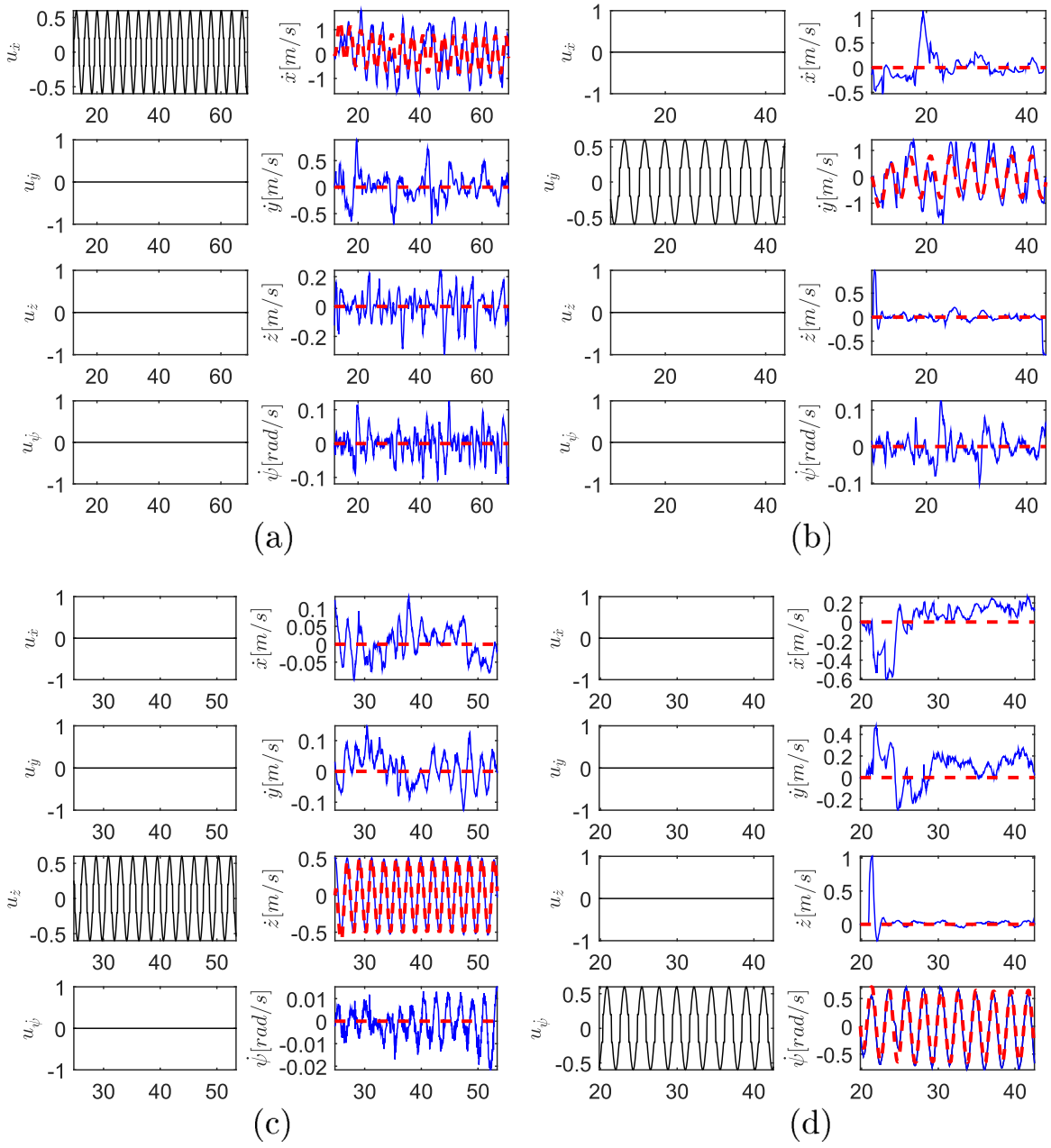

Fig. 14. The identification results are depicted in (a)–(d), where it is possible to see the control signals (the continuous black lines), the AR.Drone sensor readings (the continuous blue lines) and the simulated response of the model (2) (the dot-dashed red line). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

required that the aircraft velocity $\mathbf{v}_b^w$ converges to $\mathbf{v}_d$, as well as the position and orientation errors (ρ, $\tilde{\alpha}$, $\tilde{\beta}$) are kept inside a small region around zero.

The kinematic model of the UAV used to design the proposed controller is composed by a set of four velocities represented in the reference system $\langle b \rangle$ attached to the vehicle projected to the global system $\langle w \rangle$ according to its orientation ψ^b. Each linear velocity of the UAV refers to the reference axes. For instance, u_z causes a displacement along the z-axis, while u_x and u_y cause frontal and lateral displacements, respectively. The angular velocity ω causes a turn around the axis z, in the counterclockwise direction. Thus, the movement of the vehicle can be described by $\dot{\mathbf{x}}_r^w = \mathbf{F}(x)\mathbf{u}$, or,

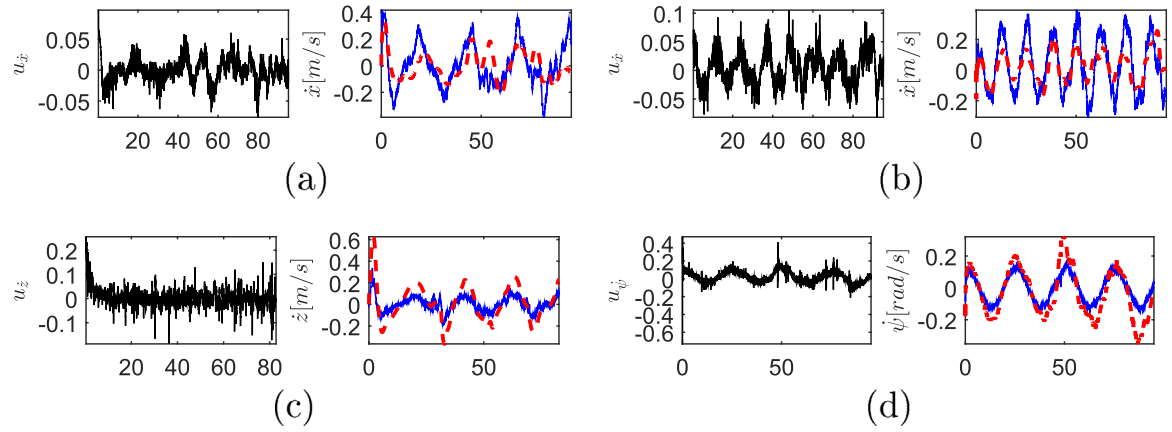

Fig. 15. Comparison between the real UAV output (the continuous blue lines) and the output of the UAV dynamic model (the dot-dashed red line). The continuous black lines are the control signals. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

Table 1
The values identified for the model parameters.

k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8
4.7202	0.2660	6.2330	0.5328	2.6504	2.5761	2.3788	1.5216

$$\dot{\mathbf{x}}^w = \begin{bmatrix} \dot{x}^w \\ \dot{y}^w \\ \dot{z}^w \\ \dot{\psi}^w \end{bmatrix} = \begin{bmatrix} \cos\psi^b & -\sin\psi^b & 0 & 0 \\ \sin\psi^b & \cos\psi^b & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \\ \omega \end{bmatrix}. \tag{46}$$

Considering $\mathbf{x}_d^w = [x_d^w \; y_d^w \; z_d^w \; \psi_d^w]^T$ as the desired posture (see Fig. 9), defined by the path being followed, its first temporal derivative is

$$\dot{\mathbf{x}}_d^w = \begin{bmatrix} \dot{x}_d^w \\ \dot{y}_d^w \\ \dot{z}_d^w \\ \dot{\psi}_d^w \end{bmatrix} = \begin{bmatrix} v_d\cos(\alpha)\cos(\beta) \\ v_d\cos(\alpha)\sin(\beta) \\ v_d\sin(\alpha) \\ \omega_d \end{bmatrix}, \tag{47}$$

where $\alpha = \tan^{-1}\left(\frac{\Delta_{z_d}}{\sqrt{\Delta_{x_d}^2 + \Delta_{y_d}^2}}\right)$ and $\beta = \tan^{-1}\left(\frac{\Delta_{y_d}}{\Delta_{x_d}}\right)$.

It is proposed to use the control law

$$\mathbf{u}_k = \mathbf{F}(\mathbf{x})^{-1}(\dot{\mathbf{x}}_d^w + \boldsymbol{\kappa}_1 \tanh(\boldsymbol{\kappa}_2 \tilde{\mathbf{x}}^w)), \tag{48}$$

where $\boldsymbol{\kappa}_1$ and $\boldsymbol{\kappa}_2$ are positive gain matrices, and $\tilde{\mathbf{x}}^w = \mathbf{x}_d^w - \mathbf{x}_b^w$ is the current posture error for the UAV. An important detail in (48) is the use of the function tanh, whose objective is to saturate the control signal when the errors are big (using it one prevents the physical saturation of the actuators of the vehicle, thus guaranteeing that no unpredictable nonlinearity will drive the system to instability).

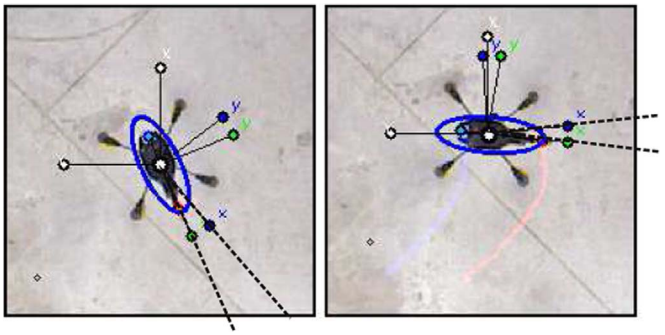

Fig. 16. UAV orientation based on information from the IMU and the RGB-D sensors, represented by the blue and green axes, respectively. The snapshots show the error obtained when calculating the orientation using the IMU, which increases along time. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

5.2. Stability analysis

By defining the difference between the Cartesian velocity of the aircraft ($\dot{\mathbf{x}}^w$) and its desired velocity along the path ($\dot{\mathbf{x}}_d^w$) as

$$\boldsymbol{\gamma} = \dot{\mathbf{u}}_k - \dot{\mathbf{x}}^w, \tag{49}$$

the closed loop system equation can be obtained by introducing (49) and (48) in (46). The result is

$$\dot{\tilde{\mathbf{x}}}^w + \boldsymbol{\kappa}_1 \tanh(\boldsymbol{\kappa}_2 \tilde{\mathbf{x}}^w) = \boldsymbol{\gamma}, \tag{50}$$

where $\tilde{\mathbf{x}}^w = \mathbf{x}_d^w - \mathbf{x}^w$ is the control error, whose first temporal derivative is $\dot{\tilde{\mathbf{x}}}^w = \dot{\mathbf{x}}_d^w - \dot{\mathbf{x}}^w$. The control objective is to get $\tilde{\mathbf{x}}^w = 0$, which is the equilibrium of the closed loop system equation.

The stability of the control system under the action of the proposed control law is analyzed using the theory of Lyapunov. Let $V = \frac{1}{2}\tilde{\mathbf{x}}^{wT}\tilde{\mathbf{x}}^w > 0$ be the Lyapunov candidate function, whose first

Table 2
The values identified for the camera parameters.

Parameter	Value
fs_u	527.77365
fs_v	527.22889
fs_θ	0
o_u	312.15707
o_v	235.82853
R	$\begin{bmatrix} -0.014798 & -0.960444 & -0.278081 \\ -0.998049 & -0.002685 & 0.062384 \\ -0.060663 & 0.278461 & -0.958530 \end{bmatrix}$
T	$\begin{bmatrix} 179.759986 \\ 31.442704 \\ 3414.074341 \end{bmatrix}$

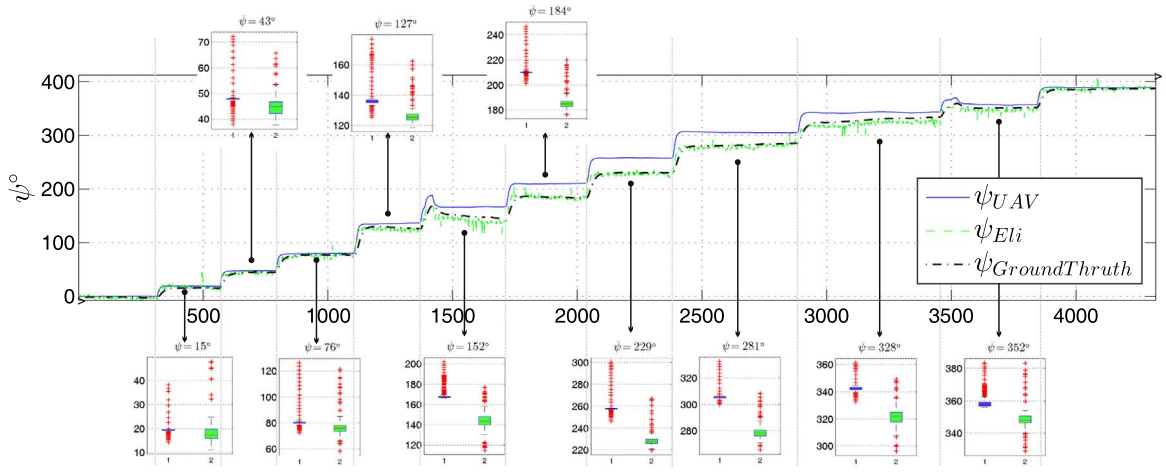

Fig. 17. UAV orientation in a manual flight experiment.

Table 3
Data to tune the variables of the fusion filter.

Ref. ψ°	UAV $\overline{\psi}_{UAV} \pm \sigma_{\psi_{UAV}}$	Depth $\overline{\psi}_{eli} \pm \sigma_{\psi_{eli}}$
15	19.62 ± 2.34	18.41 ± 5.07
43	48.29 ± 3.90	44.87 ± 4.11
76	80.98 ± 6.12	76.59 ± 7.18
127	137.45 ± 7.30	126.53 ± 5.91
152	169.48 ± 6.84	145.19 ± 10.34
184	210.86 ± 4.78	185.76 ± 5.77
229	258.66 ± 6.33	229.07 ± 6.36
281	305.74 ± 2.66	278.80 ± 4.95
328	342.47 ± 2.16	321.50 ± 5.66
352	359.22 ± 4.75	348.46 ± 5.39

temporal derivative is

$$\dot{V} = \tilde{\mathbf{x}}^{wT}\dot{\tilde{\mathbf{x}}}^{w}. \tag{51}$$

After introducing (50) in (51), one obtains

$$\dot{V} = \tilde{\mathbf{x}}^{wT}\boldsymbol{\gamma} - \tilde{\mathbf{x}}^{wT}\boldsymbol{\kappa}_1\tanh(\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w}). \tag{52}$$

A sufficient condition to have $\dot{V}$ negative, to guarantee the system stability, is $|\tilde{\mathbf{x}}^{wT}\boldsymbol{\kappa}_1\tanh(\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w})| > |\tilde{\mathbf{x}}^{wT}\boldsymbol{\gamma}|$. For great values of $\tilde{\mathbf{x}}^{w}$, one can consider that $\boldsymbol{\kappa}_1\tanh(\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w}) \approx \boldsymbol{\kappa}_1\text{sgn}\,\tilde{\mathbf{x}}^{w}$. Thus, $\dot{V}$ will be negative if $\lambda_{min}\{\boldsymbol{\kappa}_1\,\text{sgn}\,\tilde{\mathbf{x}}^{w}\} > \|\,\boldsymbol{\gamma}\,\|$, a condition that causes $\tilde{\mathbf{x}}^{w}$ to decrease. On the other hand, for small values of $\tilde{\mathbf{x}}^{w}$, $\boldsymbol{\kappa}_1\tanh(\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w}) \approx \boldsymbol{\kappa}_1\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w}$. Under such a condition (50) can be written as $\dot{\tilde{\mathbf{x}}}^{w} + \boldsymbol{\kappa}_1\boldsymbol{\kappa}_2\tilde{\mathbf{x}}^{w} = \boldsymbol{\gamma}$, and, finally, $\tilde{\mathbf{x}}$ is limited (to perform this analysis it was considered that $\boldsymbol{\gamma}$ is limited, which is a realistic assumption).

However, the control actions generated by (48) are affected by the UAV dynamics, as it is shown in (3). Thus, a sufficient condition to make the UAV to follow the desired path is to ensure that the velocity differences between the reference signals generated by the kinematic path-following controller so far designed and the UAV current velocities are close to zero. Thus, when the UAV asymptotically tracks the velocities generated by the kinematic controller of (48) the path-following errors will converge to zero asymptotically, what means that $\boldsymbol{\gamma} \to 0$ in steady-state.

As it is known, the path-following controller just generates the reference velocities. Thus, one should propose a way to connect the path-following controller with the UAV dynamics (identified in Fig. 10), to obtain an asymptotic tracking of the reference velocities. For such a dynamic compensator the control law

$$\mathbf{u}_d = \mathbf{F}_1^{-1}(\ddot{\mathbf{u}}_k + \mathbf{K}\boldsymbol{\gamma} + \mathbf{F}_2\dot{\mathbf{x}}^{b}) \tag{53}$$

is adopted, where $\mathbf{K}$ is a diagonal matrix with positive constant entries ($\mathbf{K} > 0$).

By introducing (53) in (3) one gets

$$\ddot{\mathbf{x}}^{w} = \mathbf{F}_1\mathbf{F}_1^{-1}(\ddot{\mathbf{u}}_k + \mathbf{K}\boldsymbol{\gamma} + \mathbf{F}_2\dot{\mathbf{x}}^{b}) - \mathbf{F}_2\dot{\mathbf{x}}^{b}, \text{ then} \tag{54}$$

$$0 = \underbrace{\ddot{\mathbf{u}}_k - \ddot{\mathbf{x}}^{w}}_{\dot{\boldsymbol{\gamma}}} + \mathbf{K}\boldsymbol{\gamma}, \tag{55}$$

where $\ddot{\mathbf{x}}^{w}$ and $\ddot{\mathbf{u}}_k$ are the UAV accelerations and the reference accel-

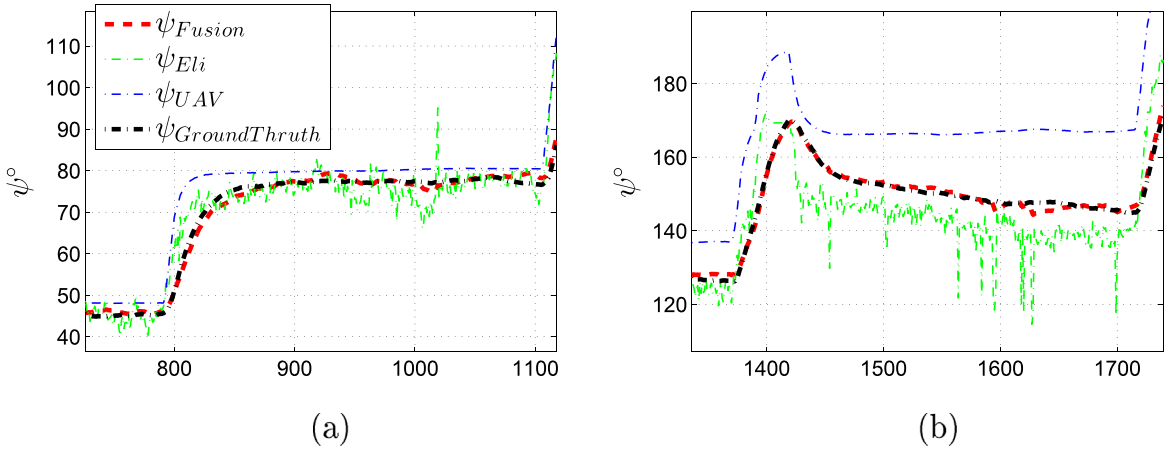

Fig. 18. Results obtained with data fusion filter.

Fig. 19. The desired path for the experiments 1(a) and 2(b).

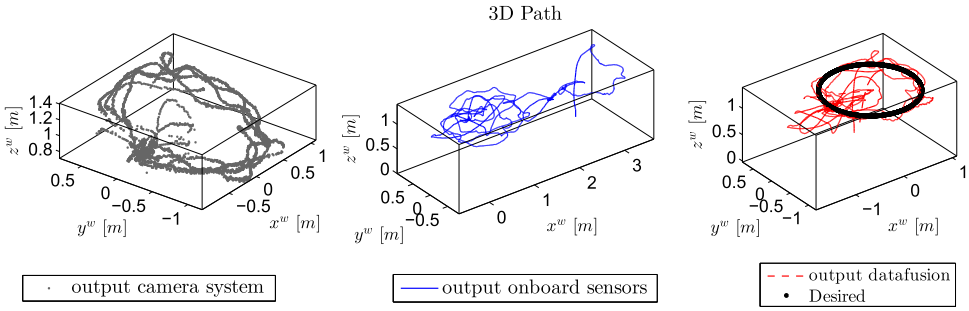

Fig. 20. Experiment 1: Path followed by the UAV during the experiment.

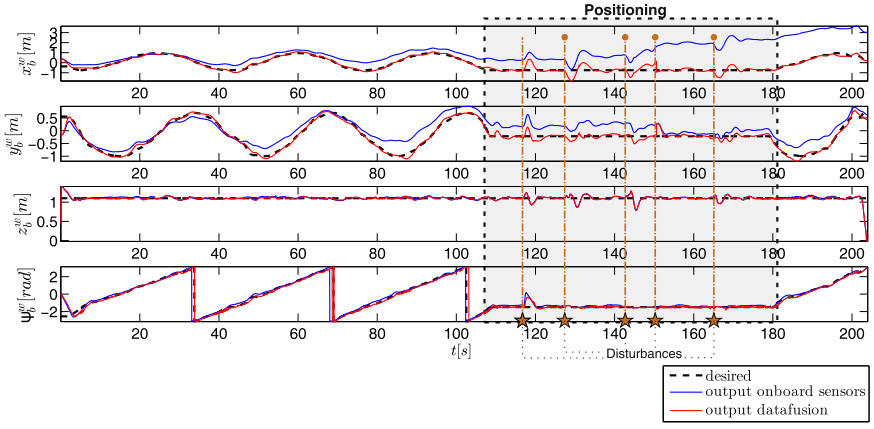

Fig. 21. Experiment 1: UAV positioning and orientation during the experiment. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

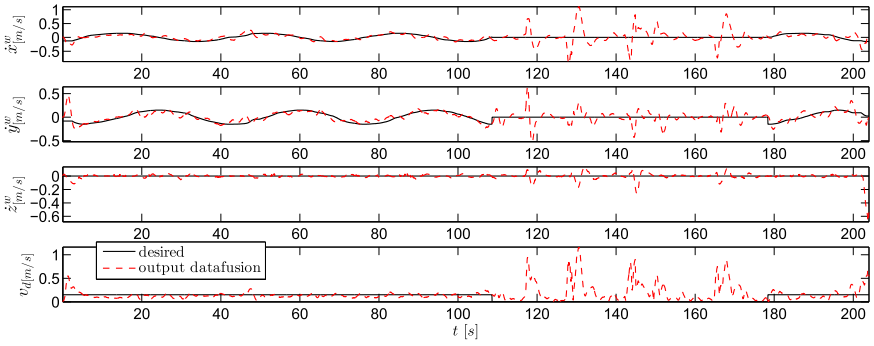

Fig. 22. Experiment 1: UAV velocity during the experiment.

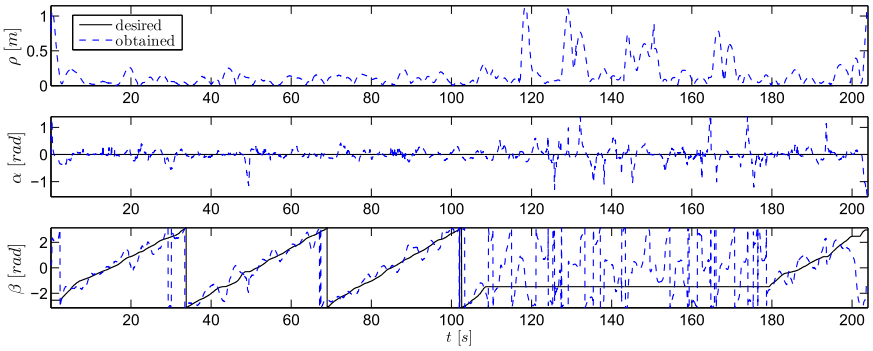

Fig. 23. Experiment 1: UAV position and orientation errors during the experiment.

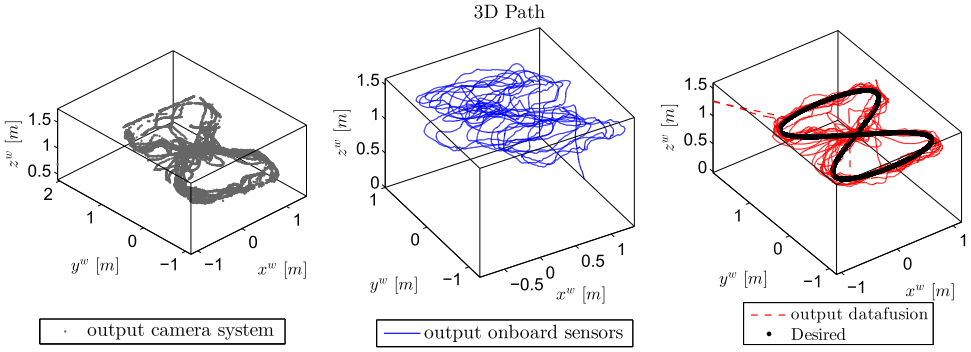

Fig. 24. Experiment 2: Path travelled by the UAV during the experiment.

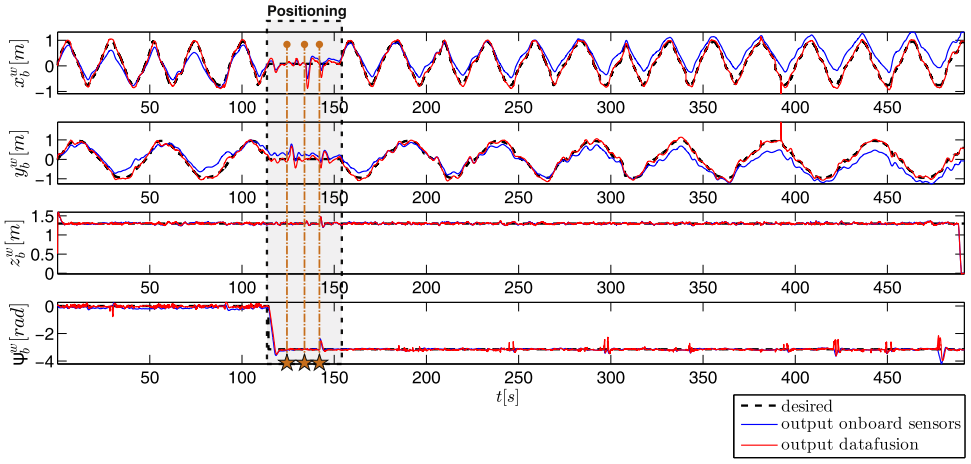

Fig. 25. Experiment 2: UAV position and orientation during the experiment.

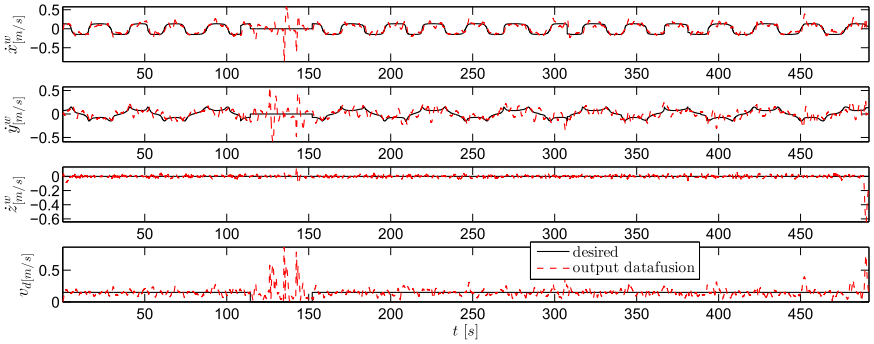

Fig. 26. Experiment 2: UAV velocities during the experiment.

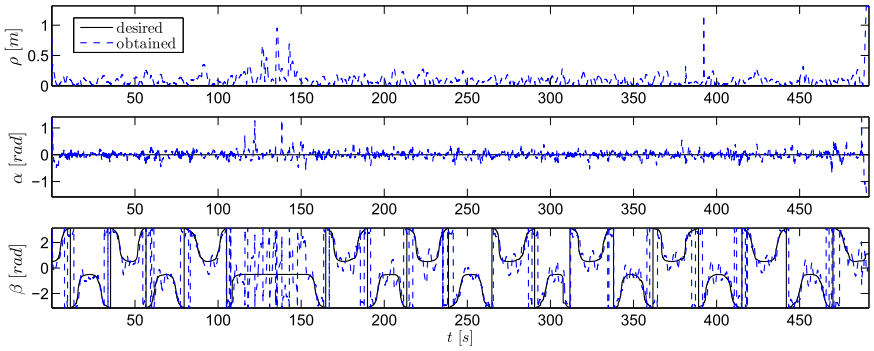

Fig. 27. Experiment 2: UAV position and orientation errors.

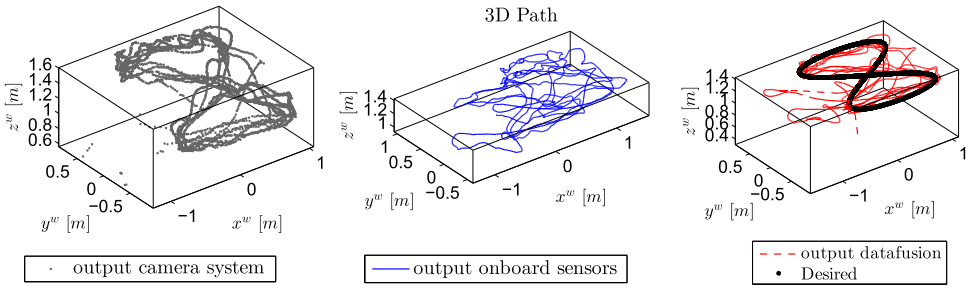

Fig. 28. Experiment 3: Path followed by the UAV during the experiment. The fusion engine uses the data from the onboard (blue) and depth (gray) sensors. The output result is the travelled path (red). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

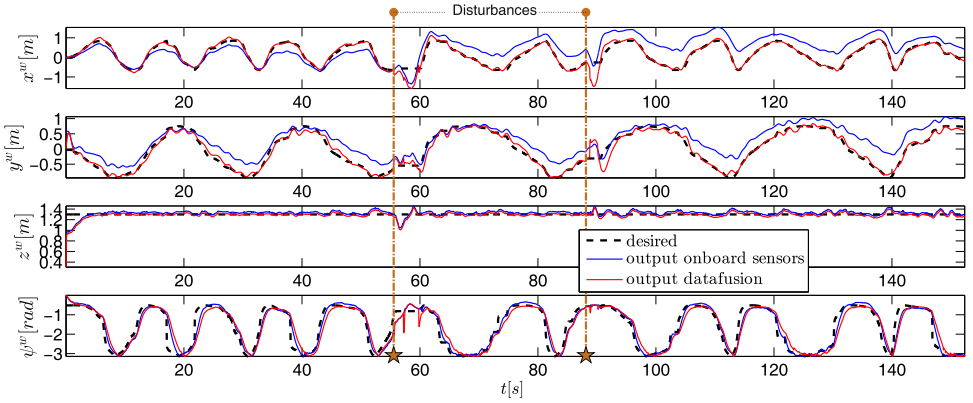

Fig. 29. Experiment 3: UAV positioning and orientation during the experiment. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

erations based on the Cartesian movement, respectively, both with reference in $\langle w \rangle$. A solution for the proposed velocity tracking is

$$\dot{\boldsymbol{\gamma}} + \mathbf{K}\boldsymbol{\gamma} = \mathbf{0}. \tag{56}$$

As a consequence, taking into account that $\boldsymbol{\gamma}(t) \to 0$ (as discussed in connection with (50)) one can conclude that $\dot{\tilde{\mathbf{x}}} \to 0$, so that $\tilde{\mathbf{x}} \to 0$. This way, one guarantees that the reference velocities generated by the path-following algorithm will be reached in a finite time.

Note 1: This study is based on kinematic and exact dynamic models. If such consideration is not accomplished, final control errors will persist, and the conclusion will be practical stability (ultimately bounded errors).

Note 2: It is noteworthy to observe that the reference velocities are dependent on the desired velocity $\mathbf{v}_d$, as one can see in (47). Thus, when $\mathbf{v}_d = \mathbf{0}$ the reference velocities of the kinematic controller are sent to the difference signal $\boldsymbol{\gamma}$. Meanwhile, the signal goal $\boldsymbol{\gamma}$ continues to be the same to maintain the tracking of the reference velocity. This way, it is possible to hover the UAV in a desired waypoint, situation in which one has a positioning control. The conclusion is that the proposed control is suitable for path-following control and positioning control as well.

6. Experiments and results

In addition to the basic hardware used in this work, the Parrot AR.Drone 2.0 Power Edition© quadrotor, the ASUS XTION Pro Live© depth sensor and a computer used as ground station, the Software Development Kits (SDK) CV Drone[1] and OpenCVKinect[2] are used for implementing the necessary algorithms. As for the ground station, considering that the computer configuration affects the system, even more when image processing is involved, like in this work, the computer used was the Notebook ASUS K450J, with the Intel Core i7-4700HQ© processor and 2.40 GHz of clock frequency. Using such a

[1] https://github.com/puku0x/cvdrone
[2] https://github.com/devkicks/OpenCVKinect

setup, the control signal update is performed each 33 ms, with the data paths correspondent to the depth images and the onboard sensors being asynchronous. Regarding such a time, 2 ms correspond to the data fusion process and control algorithm, whereas 5 ms are spent to process the depth image (resolution of 640×480, with capture at 30 fps). The remaining time is supposed to be enough for receiving the data provided by the sensors onboard the vehicle and transmitting the generated control signals through the wi-fi channel established between the ground computer and the drone. Notice that there is no delay in the data path correspondent to the depth sensor, since it is directly connected to the ground computer through a USB cable.

Using such a setup and configurations, several practical experiments were run to validate the platform and the proposed controller. As a first step, results related to UAV detection, identification of the Ar.Drone model and multi-sensor data fusion are presented. Such results are necessary to have the proposed framework ready to run navigational experiments. After having the framework ready, the proposed controller is also validated through experimentation.

6.1. UAV detection results

To estimate the functions $g_1(z)$ and $g_2(z)$, relating the minimum and maximum values of the blob area correspondent to the Ar.Drone 2.0 quadrotor in any image and its altitude, some manual flights were performed, without using its internal hull, to collect the necessary data.

The detection and classification of the objects through the Hu moments are important to the platform, since it allows the detection of any desired shape, including the UAV. However, this study proposes a less demanding method, in terms of computation (our approach demands 1.1831 ± 0.3103 ms and the algorithm of Hu demands 13.8443 ± 5.2632 ms of processing time, difference that is illustrated in Fig. 12). Our approach to detect the UAV is based on the functions g_1 and g_2 obtained as described in the sequel.

Based on the images collected and the correspondent altitudes, for the flights manually operated, Fig. 11 gives the constants estimated for the 3-rd order polynomial functions, which will help in the detection of the UAV in the indoor navigating environment. As it can be seen in the figure, the region of interest is well defined by the functions g_1 and g_2.

6.2. Model identification results

The determination of the parameters $k_1, \ldots, k_8$ of the matrices $\mathbf{K}_u$ and $\mathbf{K}_v$ is performed by acquiring the real output correspondent to $\mathbf{G}$ (see (6)) to a deformed sinusoidal input, shown in Fig. 13, for each control input. The input and output signals are logged into the ground control stations, for an off-line analysis using a system identification software (Ljung, 1995).

Notice in Figs. 14 that when a specific control signal is varying the other three are fixed in zero.

In order to check the reliability of such parameter values, an additional experiment is run, collecting experimental data from flights in which the AR.Drone receives commands simultaneously in its four control signals. The result of such experiments is presented in Fig. 15.

The result of the identification is in Table 1, and the real output, together with the output generated by using the identified dynamic model, is shown in Fig. 15. This comparison allows validating the model adopted and the identified parameters.

6.3. Data fusion results

To perform the data fusion, one should tune the constants of each node of the Decentralized Information Filter (Santos, Santana, Martins et al., 2015). Thus, an experiment to collect information to calculate the variance of each sensor in the estimation process of the orientation of the UAV was implemented. Fig. 16 shows snapshots of a navigation with the detection of the drone and the calculated orientations. The colored markers (Fig. 1) attached to the UAV give the orientation considered as a "ground truth". It is observed that, from snapshot to snapshot, a difference arises between the measurements provided by the IMU of the drone and those obtained with the RGB-D image processing.

Eq. (19) shows how to convert data from the camera to the real world, and Table 2 gives the values identified for the camera parameters (calculated using http://www.vision.caltech.edu/bouguetj/calib_doc).

The markers were then used to tune the information filter. Fig. 17 shows the results obtained in this stage.

With the values of Table 3, it is possible to determine the values for the error covariance obtained by the observations ($\sigma^2_{\tilde{\psi}_{UAV}} = 109.39°$, $\sigma^2_{\tilde{\psi}_{ell}} = 36.64°$). The errors between the data correspondent to the ground truth and the filtering result for such angle, using the inertial ellipse for angle calculation, are shown in Fig. 18. Still analyzing such a table, it is observed that the average error without the filtering process was $-1.468 \pm 6.053°$. However, after applying the filter, it obtained an average error of $-1.479 \pm 4.453°$, increasing the reliability of the estimated orientation.

Finally, the result of the data fusion obtained with the onboard IMU/ultrasonic sensor and the data acquired with the depth sensor is presented in Fig. 18.

6.4. Results for the navigation using the cascade controller

In this subsection experiments using the UAV orientation and position estimated using the sensory data fusion subsystem and the proposed cascade controller are discussed. Three experiments were run, changing the type of path (sequence of points and correspondent orientations). Figs. 19(a) and (b) present the desired path correspondent to the experiments 1 and 2, respectively. In the third experiment, it used the same path of experiment 2, but in this case the disturbances were applied with the UAV in movement, not hovering.

In experiment 1, the path has an orientation change between points, as it was seen in Section 5, represented by β. The tracking of this orientation is important for some practical tasks, and is a classical procedure of the path-following controllers (Roza & Maggiore, 2012). For that, the controller has to adopt $\psi_d^w \to \beta$. As shown in Fig. 20, the UAV follows an ellipsoidal path.

During the experiment, it established an observation point, where the UAV stays hovering ($\dot{\mathbf{x}}_d^w = 0$) for a while. In Fig. 21, this event is denominated as "positioning". In the interval $t \approx [105, 182]$ s, the controller has the goal of keeping the vehicle positioned on such waypoint. To validate the positioning task, some disturbances were applied to the UAV, represented by the yellow lines with a star in Fig. 21.

These disturbances influence directly the UAV velocities. In Fig. 22, it can be observed, during the interval $t = [105, 182]$ s, that despite the desired velocities being zero, the magnitude of the disturbance assigns velocities to the UAV, which tend to vanish after a while (in such a situation the desired velocities are zero).

It was applied 5 disturbances during the observation task. Besides this, the goal of the path-following and positioning controllers (cascade control approach) is still successfully accomplished: observe, in Fig. 23, that the position and orientation errors tend to the desired values.

In the experiment 2 an eight-shaped path should be followed by the quadrotor. Such path is frequently used in robotics to validate controllers, since it executes control actions in every direction due to its characteristics and the presence of curvatures along the path. Thus, the task accomplishment validates that the cascade control system works in every direction. In this experiment, the goal of the orientation tracking differs from experiment 1. The UAV starts the path with the orientation $\psi_d^w = 0°$. When the UAV is hovering to execute the observation task, its desired orientation changes to $\psi_d^w = 180°$. Fig. 24 presents the path followed by the UAV during the experiment 2,

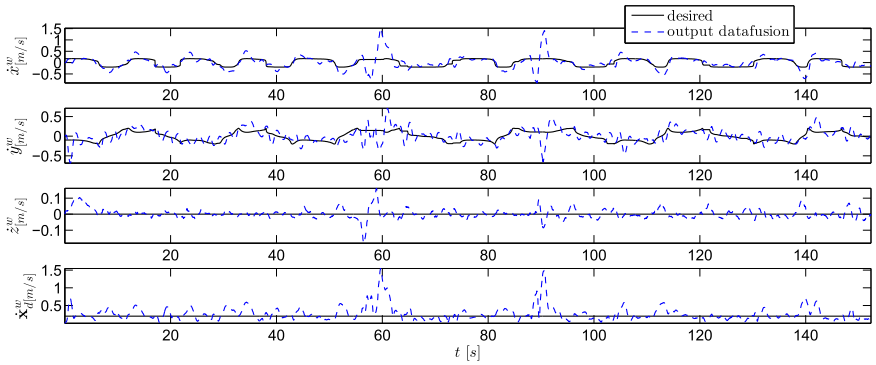

Fig. 30. Experiment 3: UAV velocity during the experiment.

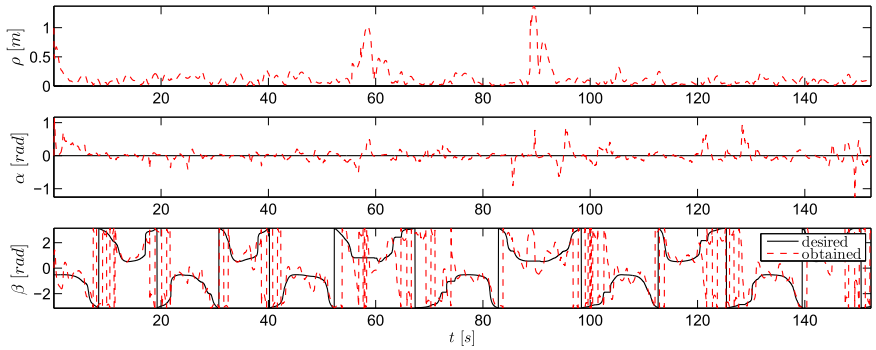

Fig. 31. Experiment 3: UAV position and orientation errors during the experiment.

whereas Fig. 25 shows the orientation change.

The observation point is highlighted in Fig. 25. In the interval $t \approx [115, 160]$ s, the controller has the objective of keeping the vehicle in this waypoint with an inverted orientation. Disturbances are applied in those instants.

The UAV velocities during the path-following are presented in Fig. 26. The effect of the disturbances are clearly visible in Figs. 26 and 27. In addition, the objective of the path-following and positioning controller (cascade control approach) is reached, once the position and orientation errors achieve the desired values, as shown in Fig. 27.

In both experiments so far described the tasks of path-following and positioning were executed. Disturbances were applied to the UAV when it stayed hovering, which validated the robustness of the controller to such disturbances. To help the readers to understand the experiments, the link https://youtu.be/p7WdYI4pfJQ addresses a video that shows the complete experiment.

The main difference of experiment 3, compared to experiment 2, is the fact that disturbances are applied to the UAV in motion. In Fig. 28 the output result of the data fusion subsystem, in such a case, can be clearly observed.

In the instants 56.23 s and 88.18 s, some disturbances were applied to the UAV, represented by yellow lines with a star in Fig. 29.

As it was seen before, these disturbances influence directly the UAV velocities. In Fig. 30, in the instants 56.23 s and 88.18 s, the magnitude of the disturbances changes the velocities of the UAV, which it corrects along time.

As for the control and orientation errors, in Fig. 31 it is observed that they tend to the desired values.

The link https://youtu.be/A2RNt9O8NZM addresses a video that shows the complete experiment.

Notice that data delivered by the IMU/ultrasonic sensor onboard the Ar.Drone to estimate x^w and y^w do suffer bias or drift effect, whereas z^w and ψ do not do it. This is due to the fact that internally the AR.Drone performs a data fusion of the IMU information with other sensors, such as an ultrasonic module, gyroscopes and magnetometers. The two latter sensors influence the estimation of ψ, and the ultrasonic module influences the estimation of z. Figs. 21, 25 and 29 show exactly this behavior.

Notice that the position x_b^w, y_b^w, z_b^w and the orientation ψ_b^w of the UAV during the two first experiments, shown in Figs. 21 and 25, correspond to the multi-sensor fusion estimates. Despite the aforementioned errors, however, both path-following experiments have been successfully performed. Finally, as commented in Section 5, and according to Figs. 23, 27 and 31, the errors in the control variables $(\rho, \alpha) \to 0$ and $\beta \to \beta_d$ when $t \to \infty$, showing that the control objectives are achieved.

7. Conclusions

This work presents the development of a capture system that aims at obtaining the 3D position and orientation of an UAV in an indoor environment, with the aid of a depth sensor (Xtion Pro Live, from ASUS). It also presents a cascade controller for guiding the vehicle in positioning and path-following tasks with an AR.Drone quadrotor.

In the process of evaluating the system with data obtained through the onboard IMU/ultrasonic sensor, it was observed that the position data obtained by such sensors differ from the correspondent data delivered by the depth sensor. Then, a data fusion system was implemented using the data obtained with the depth sensor and the data provided by the IMU/ultrasonic sensor, and the result obtained in connection with path-following UAV navigation demonstrated that the developed system can be used to improve the process of defining the position and orientation of the UAV in the real world in indoor environments. This way, the UAV accomplished the path-following and surveillance operations (in this last case hovering at a desired waypoint with a desired heading). An important aspect is that both tasks are accomplished using the same controller.

The main contributions of this work are the implementation of an

experimental environment for validating high-level UAV controllers, with fast detection and localization algorithms, without needing markers, and the development of a control strategy that allows accomplishing path-following and positioning tasks using a cascade control approach. This approach has the advantage of defining any desired velocity through the path, including $v_d = 0$, which differs from other path-following controllers available in the literature.

Finally, it is important to mention that the whole system was developed with low cost electronics: just a depth camera and the Parrot AR.Drone 2.0 platform.

As future work, we are interested in using this platform for validating other controllers that our research team has been developing, for which just simulations have been performed so far (Brandao, Gandolfo, Filho, & Carelli, 2014; Brandão & Sarcinelli-Filho, 2015; Rosales, Leica, Sarcinelli-Filho, Scaglia, & Carelli, 2016).

Acknowledgment

The authors thank CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico, an agency of the Brazilian Ministry of Science and Technology to support scientific and technological development –, FAPES – Fundação de Amparo à Pesquisa e Inovação do Espírito Santo, the agency of the State of Espírito Santo that supports scientific and technological development –, for the financial support to this work. They also thank CAPES – Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, an agency of the Brazilian Ministry of Education to support high education –, for the scholarship granted to Mr. Santos, the Federal Institute of Espírito Santo, the Federal University of Espírito Santo and the Institute of Automatics of the National University of San Juan, Argentine, for supporting the development of this research.

References

Acgtelik, M., Bachrach, A., He, R., Prentice, S., & Roy, N. (2009). Stereo vision and laser odometry for autonomous helicopters in gps-denied indoor environments. In *Proceedings of the SPIE unmanned systems technology XI* (Vol. 7332).

Aguiar, A., Hespanha, J., & Kokotovic, P. (2005). Path-following for nonminimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50(2), 234–239. http://dx.doi.org/10.1109/TAC.2004.841924.

Ahn, H.-S., Yu, W., & Lee, J. (2007). Wireless localization network for ubiquitous robotic space: Approaches and experimental test. In *The 16th IEEE international symposium on robot and human interactive communication, 2007. RO-MAN 2007* (pp. 481–486) http://dx.doi.org/10.1109/ROMAN.2007.4415131.

Assimakis, N., Adam, M., & Douladiris, A. (2012) Information filter and Kalman filter comparison: Selection of the faster filter. *International Journal of Information Engineering, 2* (1).

Astrom, K. J., Wittenmark, B. (1994). *Adaptive control* (2nd ed). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Brandao, A. S., Gandolfo, D., Filho, M. S., & Carelli, R. (2014). Pvtol maneuvers guided by a high-level nonlinear controller applied to a rotorcraft machine. *European Journal of Control*, 20, 172–179.

Brandão, A. S., & Sarcinelli-Filho, M. (2015). On the guidance of multiple uav using a centralized formation control scheme and Delaunay triangulation. *Journal of Intelligent & Robotic Systems*, 1–17. http://dx.doi.org/10.1007/s10846-015-0300-5 (to appear).

Camera Calibration Toolbox for Matlab (2016) ⟨http://www.vision.caltech.edu/bouguetj/calib_doc⟩, [Online; accessed 04-March-2016].

Chaumette, F. (2004). Image moments: A general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, 20(4), 713–723. http://dx.doi.org/10.1109/TRO.2004.829463.

Chen, H., Chang, K., & Agate, C. (2013). Uav path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance. *IEEE Transactions on Aerospace and Electronic Systems*, 49(2), 840–856. http://dx.doi.org/10.1109/TAES.2013.6494384.

Elmezain, M., Al-Hamadi, A., & Michaelis, B. (2009). Improving hand gesture recognition using 3d combined features. In *Second international conference on machine vision* (pp. 128–132).

Engel, J., Sturm, J., & Cremers, D. (2012). Camera-based navigation of a low-cost quadrocopter. In *Proceedings of the 2012 IEEE/RSJ international conference on intelligent robots and systems*.

Falcon, P., Barreiro, A., & Cacho, M. (2013). Modeling of Parrot Ardrone and passivity-based reset control. In *2013 9th Asian control conference (ASCC)* (pp. 1–6) http://dx.doi.org/10.1109/ASCC.2013.6606362.

Ferrick, A., Fish, J., Venator, E., & Lee, G. (2012). Uav obstacle avoidance using image processing techniques. In *2012 IEEE international conference on technologies for practical robot applications (TePRA)* (pp. 73–78). http://dx.doi.org/10.1109/TePRA.2012.6215657.

Flores Colunga, G.-R., Zhuo, S., Lozano, R., & Castillo, P. (2014). A vision and gps-based real-time trajectory planning for a mav in unknown and low-sunlight environments. *Journal of Intelligent & Robotic Systems*, 74(1–2), 59–67 URL ⟨http://hal.archives-ouvertes.fr/hal-00923131⟩.

Freire, E. O., Filho, T. F. B., Filho, M. S., & Carelli, R. (2004). A new mobile robot control architecture: Fusion of the output of distinct controllers. *Transactions on Systems, Man and Cybernetics*. http://dx.doi.org/10.1109/ISIC.2002.1157753.

Hazas, M., Scott, J., & Krumm, J. (2004). Location-aware computing comes of age. *IEEE Computer*, 37(2), 95–97 URL ⟨http://research.microsoft.com/apps/pubs/default.aspx?id=64615⟩.

Hehn, M., & D'Andrea, R. (2011) Quadrocopter trajectory generation and control. In *Proceedings of the IFAC world congress* (pp. 1485–1491).

Hehn, M., Ritz, R., & D'Andrea, R. (2012). Performance benchmarking of quadrotor systems using time-optimal control. *Autonomous Robots*, 33, 69–88. http://dx.doi.org/10.1007/s10514-012-9282-3.

Hernandez, A., Copot, C., De Keyser, R., Vlas, T., & Nascu, I. (2013). Identification and path following control of an ar.drone quadrotor. In *2013 17th international conference on system theory, control and computing (ICSTCC)* (pp. 583–588) http://dx.doi.org/10.1109/ICSTCC.2013.6689022.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2), 179–187. http://dx.doi.org/10.1109/TIT.1962.1057692.

Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Fox, D., & Roy, N. (2011). Visual odometry and mapping for autonomous flight using an rgb-d camera. In *Proceedings of the robotic research*.

Krajník, T., Vonásek, V., Fišer, D., & Faigl, J. (2011). AR-Drone as a robotic platform for research and education. In *International conference on research and education in robotics*. Heidelberg: Springer.

Ljung L. (1995). *System identification toolbox for use with MATLAB: User's guide*, sOP-95-89/SOP-96-68. URL ⟨http://opac.inria.fr/record=b1070642⟩.

Losada, C., Mazo, M., Palazuelos, S., Pizarro, D., & Marrón, M. (2010). Multi-camera sensor system for 3d segmentation and localization of multiple mobile robots. *Sensors*, 10(4), 3261. http://dx.doi.org/10.3390/s100403261 URL ⟨http://www.mdpi.com/1424-8220/10/4/3261⟩.

Lupashin, S., & D'Andrea, R. (2011). Adaptive open-loop aerobatic maneuvers for quadrocopters. In *Proceedings of the IFAC world congress*.

Lupashin, S., Hehn, M., Mueller, M. W., Schoellig, A. P., Sherback, M., & D'Andrea, R. (2014). A platform for aerial robotics research and demonstration: The flying machine arena. *Mechatronics*(1). http://dx.doi.org/10.1016/j.mechatronics.2013.11.006.

Lupashin, S., Schollig, A., Hehn, M., & D'Andrea, R. (2011). The flying machine arena as of 2010. In *2011 IEEE international conference on robotics and automation (ICRA)* (pp. 2970–2971). http://dx.doi.org/10.1109/ICRA.2011.5980308.

Ma, Y., Soatto, S., Kosecka, J., & Sastry, S. S. (2001). *An invitation to 3-D vision from images to models*. Springer Verlag.

Mellinger, D., Michael, N., & Kumar, V., (2014). Trajectory generation and control for precise aggressive maneuvers with quadrotors. In *The 12th international symposium on experimental robotics* (pp. 361–373). Berlin, Heidelberg: Springer.

Mercimek, M., Gulez, K., & Mumcu, T. V. (2005). Real object recognition using moment invariants. *Sadhana*(6), 765–775. http://dx.doi.org/10.1007/BF02716709.

Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The grasp multiple micro-uav testbed. *IEEE Robotics Automation Magazine*, 17(3), 56–65. http://dx.doi.org/10.1109/MRA.2010.937855.

Morioka, K., & Hashimoto, H. (2004). Appearance based object identification for distributed vision sensors in intelligent space. In *Proceedings of the 2004 IEEE/RSJ international conference on intelligent robots and systems, 2004 (IROS 2004)* (Vol. 1, pp. 199–204) http://dx.doi.org/10.1109/IROS.2004.1389352.

Mutambara, A. G. (1998). *Decentralized estimation and control for multisensor systems*. CRC Press.

Nelson, D., Barber, D., McLain, T., & Beard, R. (2007). Vector field path following for miniature air vehicles. *IEEE Transactions on Robotics*, 23(3), 519–529. http://dx.doi.org/10.1109/TRO.2007.898976.

Park, S., Deyst, J., & How, J. P. (2007). Performance and Lyapunov stability of a nonlinear path following guidance method. *Journal of Guidance, Control, and Dynamics*, 30(6), 1718–1728.

Pedro Castillo Garcia, A. E. D., & Lozano, R. (2005). *Modelling and control of mini-flying machines*. Springer.

Piccardi, M. (2004). Background subtraction techniques: A review. In *2004 IEEE international conference on systems, man and cybernetics* (Vol. 4, pp. 3099–3104) http://dx.doi.org/10.1109/ICSMC.2004.1400815.

Rampinelli, M., Covre, V. B., de Queiroz, F. M., Vassallo, R. F., Bastos-Filho, T. F., & Mazo, M. (2014). An intelligent space for mobile robot localization using a multi-camera system. *Sensors*, 14(8), 15039. http://dx.doi.org/10.3390/s140815039 URL ⟨http://www.mdpi.com/1424-8220/14/8/15039⟩.

Regazzoni, D., de Vecchi, G., & Rizzi, C. (2014). Rgb cams vs rgb-d sensors: Low cost motion capture technologies performances and limitations. *Journal of Manufacturing Systems*, 33(4), 719–728. http://dx.doi.org/10.1016/j.jmsy.2014.07.011.

Rhee, I., Park, S., & Ryoo, C.-K. (2010), A tight path following algorithm of an uas based on pid control. In *Proceedings of SICE annual conference 2010* (pp. 1270–1273).

Rosales, C., Leica, P., Sarcinelli-Filho, M., Scaglia, G., & Carelli, R. (2016). 3d formation control of autonomous vehicles based on null-space. *Journal of Intelligent & Robotic Systems*, 1–15. http://dx.doi.org/10.1007/s10846-015-0329-5 (to appear).

Rothe, I., Susse, H., & Voss, K. (1996). The method of normalization to determine

invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4), 366–376. http://dx.doi.org/10.1109/34.491618.

Roza, A., & Maggiore, M. (2012). Path following controller for a quadrotor helicopter. In *American control conference (ACC), 2012* (pp. 4655–4660). http://dx.doi.org/10.1109/ACC.2012.6315061.

Saadeddin, K., Abdel-Hafez, M. F., & Jarrah, M. A. (2013). Estimating vehicle state by gps/imu fusion with vehicle dynamics. In *International conference on unmanned aircraft systems (ICUAS)*.

Santana, L. V., Brandão, A. S., Filho, M. S., & Carelli, R. (2014). A trajectory tracking and 3d positioning controller for the ar.drone quadrotor. In *Unmanned aircraft systems (ICUAS)*.

Santana, L. V., Brandão, A. S., & Sarcinelli-Filho, M. (2016). Navigation and cooperative control using the ar.drone quadrotor. *Journal of Intelligent & Robotic Systems*, 1–24. http://dx.doi.org/10.1007/s10846-016-0355-y.

Santos, M., Santana, L., Brandao, A., & Sarcinelli-Filho, M. (2015). Uav obstacle avoidance using rgb-d system. In *2015 international conference on unmanned aircraft systems (ICUAS)* (pp. 312–319) http://dx.doi.org/10.1109/ICUAS.2015.7152305.

Santos, M., Santana, L., Martins, M., Brandão, A., & Sarcinelli-Filho, M. (2015). Estimating and controlling uav position using rgb-d/imu data fusion with decentralized information/Kalman filter. In *2015 IEEE international conference on industrial technology (ICIT)* (pp. 232–239) http://dx.doi.org/10.1109/ICIT.2015.7125104.

Schoellig, A. P., Siegel, H., Augugliaro, F., & D'Andrea, R. (2014). So you think you can dance? Rhythmic flight performances with quadrocopters. In *Controls and art*. Springer International Publishing (pp. 73–105).

Steinhaus, P., Strand, M., & Dillmann, R. (2007). Autonomous robot navigation in human-centered environments based on 3d data fusion. *EURASIP Journal on Advances in Signal Processing*. http://dx.doi.org/10.1155/2007/86831.

Sujit, P. B., Saripalli, S., & Sousa, J. B. (2014). Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicles. *IEEE Control Systems*, 34(1), 42–59. http://dx.doi.org/10.1109/MCS.2013.2287568.

The Navigation and Control Technology Inside the AR.Drone Micro UAV. In *18th {IFAC} world congress* (Vol. 44) http://dx.doi.org/10.3182/20110828-6-IT-1002.02327.

Tournier, G. P., Valenti, M., How, J. P., & Feron, E. (2006). Estimation and control of a quadrotor vehicle using monocular vision and Moiré patterns. In *AIAA guidance, navigation and control conference* (pp. 2006–6711). AIAA.

Turpin, M., Michael, N., & Kumar, V. (2012). Trajectory design and control for aggressive formation flight with quadrotors. *Autonomous Robots*, 33(1), 143–156. http://dx.doi.org/10.1007/s10514-012-9279-y.

Want, R., Hopper, A., Falcão, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems*, 10(1), 91–102. http://dx.doi.org/10.1145/128756.128759.

Weiss, S., Achtelik, M. W., Lynen, S., Achtelik, M. C., Kneip, L., Chli, M. et al. (2013). Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, 30(5), 803–831. http://dx.doi.org/10.1002/rob.21466.

Willmann, J., Augugliaro, F., Cadalbert, T., D'Andrea, R., Gramazio, F., & Kohler, M. (2012). Aerial robotic construction: Towards a new field of architectural research. *International Journal of Architectural Computing*, 10, 439–460.