

Article

Towards the Development of a Smart Flying Sensor: Illustration in the Field of Precision Agriculture

Andres Hernandez *, Harold Murcia, Cosmin Copot and Robin De Keyser

Department of Electrical Energy, Systems and Automation (EeSA), Ghent University, 9000 Ghent, Belgium; E-Mails: harold.murcia@unibague.edu.co (H.M.); cosmin.copot@ugent.be (C.C.); Robain.DeKeyser@ugent.be (R.D.K.)

* Author to whom correspondence should be addressed; E-Mail: Andres.Hernandez@ugent.be; Tel./Fax: +32-9264-55-84.

Academic Editor: Felipe Gonzalez Toro

Received: 15 May 2015 / Accepted: 3 July 2015 / Published: 10 July 2015

Abstract: Sensing is an important element to quantify productivity, product quality and to make decisions. Applications, such as mapping, surveillance, exploration and precision agriculture, require a reliable platform for remote sensing. This paper presents the first steps towards the development of a smart flying sensor based on an unmanned aerial vehicle (UAV). The concept of smart remote sensing is illustrated and its performance tested for the task of mapping the volume of grain inside a trailer during forage harvesting. Novelty lies in: (1) the development of a position-estimation method with time delay compensation based on inertial measurement unit (IMU) sensors and image processing; (2) a method to build a 3D map using information obtained from a regular camera; and (3) the design and implementation of a path-following control algorithm using model predictive control (MPC). Experimental results on a lab-scale system validate the effectiveness of the proposed methodology.

Keywords: UAV; remote sensor; precision agriculture

1. Introduction

In the last few years, there has been great interest from different industries to obtain better product quality at higher production rates, to improve energy efficiency, while decreasing production costs. An

essential element in achieving these goals is sensing; without reliable and accurate measurements, it is impossible to quantify productivity and, therefore, unfeasible to make timely corrections.

Depending on the application, different instrumentation structures can be found, but with a clear trend towards the use of multiple sensors, in order to collect all possible information about the system. Therefore, typically, sensor networks are used [1]. This is especially true for applications, such as production machines, automation, mapping, precision agriculture and weather forecasting, where a large area needs to be covered [2].

Sensing of large areas leads to difficulties, such as power supply, calibration procedures, data delays, accessibility issues for installation and maintenance, as well as high costs. Researchers have been working on developing wireless sensor networks to mitigate some of these problems [3]. A possibly more effective solution consists of using a remote sensor to ‘freely’ move in the space, thus increasing flexibility while diminishing costs, because that solution requires less sensors and is easier to maintain. In this matter, the use of aerial vehicles appears as an interesting option, since they have the ability to maneuver through complex environments. Even more interesting is the use of autonomous unmanned aerial vehicles (UAV) that can execute complex tasks without sustained human supervision, given their capability to perform tasks that are difficult or costly for manned aerial vehicles to accomplish [4]. In order to undertake the challenging task of automated flight and maneuvering, a versatile flight control design is required. One of the aerial vehicles that can accomplish this is a quadrotor due to its relatively small size, ability to hover and mechanical simplicity [5].

A large number of papers have emerged in the literature on quadrotors. Modeling, identification and control of a quadrotor are described by [6] using on-board sensing. Furthermore, simultaneous localization and mapping (SLAM) was implemented to create 3D maps of the environment, as well as to establish the quadrotor position in space [7]. Automatic navigation and object recognition with filtered data from on-board sensors and cameras is reported in [8]. Complex tasks, such as catching a falling object using a single quadrotor, have been accomplished in [5] or for a group of quadrotors in cooperative formation in [9], where high-speed external cameras were used to estimate the position of both the object and UAV. Difficult tasks, such as flying in cities or forests, require further progress on image processing, to achieve reliable detection of obstacles using low-weight hardware and low computational costs [10]. Current implementations in UAVs still require an on-ground operator with visual contact to the aerial vehicle for tasks, like taking off, landing, collision avoidance and adaptive path-planning. There exists a need to design methodologies to cope with these conditions in order to increase the degree of intelligence and therefore autonomy of UAVs.

Regarding applications for remote sensing, precision agriculture, exploration, surveillance and mapping are some of the main activities, because a smart remote sensor can be useful to take information from different angles and/or follow predefined paths in areas that are difficult access for human beings [11,12]. For example, in the field of precision agriculture, the goal is to gather and analyze information about the variability of soil and crop conditions, in order to maximize the efficiency of crop inputs within small areas of the farm field [13]. This requires obtaining reliable information about the crop conditions in real time. A possible solution might be to install a net of ground sensors; unfortunately, this is a costly and difficult to maintain solution. Mapping is also of high interest in activities such as cartography, archeology and architecture [14]. Here, the challenge lies in the accessibility to remote

areas for a human being. In the case of exploration and surveillance, a smart remote sensor could autonomously fly above a pipeline of gas or petrol, in order to find leaks or detect corrosion, once the UAV is equipped with the necessary instrumentation.

This paper presents the first steps towards the development of a smart flying sensor, using as a flying platform a low-cost quadrotor. The task of mapping the volume of grain inside a trailer during forage harvesting is used as a test case to demonstrate the concept of smart remote sensing and to assess the performance of the proposed solution. In the context of this research, smart is linked to autonomy or the capacity of the flying sensor to make decisions without human intervention. Thus, automatic detection of the trailer, precise position control to avoid the phantom effect on the pictures and image processing algorithms to automatically generate a 3D profile are some of the features that make this a smart sensor. The main contributions are the development of: (i) a position-estimation method with delay compensation based on sensor fusion; (ii) a method to build a 3D map using information obtained from a regular camera; and (iii) a path-following predictive control algorithm to guarantee the accurate position of the UAV in space.

The paper is structured as follows. A description of the hardware, instrumentation and position estimation using sensor fusion is presented in Section 2. In Section 3, the quadrotor's dynamics and modeling is presented, followed by the path-following control algorithm in Section 4. Next, the effectiveness of the proposed smart remote sensor is tested for the task of mapping in a precision agriculture application, as described in Section 5. The final section summarizes the main outcome of this contribution and presents the next challenges.

2. Quadrotor Characteristics and Sensory Equipment

Quadrotors have four rotating blades, which enable flight similar to that of a helicopter. Movement is attained by varying the speeds of each blade, thereby creating different thrust forces. Today, they are equipped with on-board sensory equipment and the ability to communicate wirelessly with a command station, thus making it possible to implement advanced control algorithms to achieve precise control, even during aggressive aerial maneuvers.

In this work, the commercially available and low-cost AR.Drone 2.0 is used as the flying platform. A description of its main characteristics, sensory equipment and position estimation using sensor fusion is presented in this section.

2.1. Characteristics of the AR.Drone 2.0 Quadrotor

The quadrotor comes with internal in-flight controllers and emergency features, making it stable and safe to fly [15]. The only downside would be that access to the quadrotor's internal controller is restricted. The internal software is a black-box system, and the parameters that refer to control and other calibration procedures are not fully documented. There are four brushless DC motors powered with 14.5 W each, from the three-element 1500-mA/H LiPo rechargeable battery that provides an approximate flight autonomy of 15–20 min. Two video cameras are mounted on the central hull, pointing to the front and to the bottom of the quadrotor.

This on-board black-box system in the AR.Drone 2.0 can be considered the low layer. The high layer is represented by the command station, which defines the references to the internal controllers located in the low layer. A schematic representation is depicted in Figure 1.

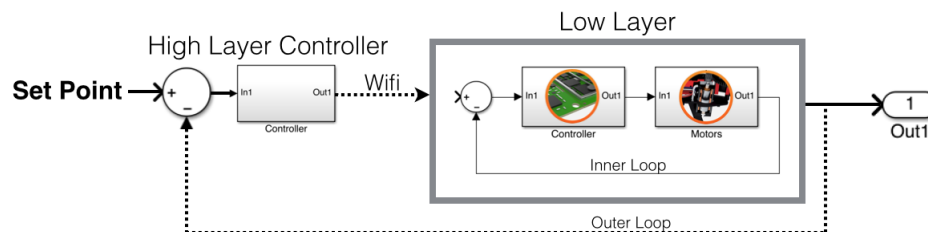


Figure 1. Quadrotor layers: the low layer represents the electronic assistance and the embedded operative system on the AR.Drone; the high layer represents the pilot (natively a smart device *i.e.*, iPhone).

In this application, the high layer consists of a C++ application in Visual Studio, which allows accessing all AR.Drone communication channels, therefore enabling functions to send commands or set configurations, receive and store data from sensors and video-stream. Thus, data can be interpreted off- or on-line for modeling, identification or control purposes. Movement in the quadrotor is achieved by furnishing reference values as input to the internal black-box controllers.

2.2. Technical Specifications and Sensory Equipment

2.2.1. Inertial Measurement Unit Board

The micro-electro-mechanical systems (MEMS)-based sensors are located below the central hull. They consist of:

- a three-axis accelerometer of ± 50 mg precision
- a three-axis gyroscope with $2000^\circ/\text{s}$ precision
- a three-axis magnetometer of 6° precision

which together form the inertial measurement unit (IMU). The IMU provides the software with pitch, roll and yaw angle measurements. The measurements are also used for internal closed-loop stabilizing control (black-box).

2.2.2. Ultrasonic Sensor

The ultrasonic sensor is used for low altitudes (below 3 m); it operates using two different frequencies, 22.22 Hz and 25 Hz, in order to reduce noise or perturbations between quadrotors. It is important to clarify that even if the sound propagates in a cone-like manner, the ultrasonic sensor does not provide a map of what is inside the cone's area. Instead, it outputs a single measured value, which corresponds to the highest object on the ground present somewhere within the cone's area, as illustrated in Figure 2. This effect makes it more difficult to accurately determine the altitude in rough terrain or

the presence of ground obstacles. The relation between the altitude and the diameter of the cone is represented in Equation (1).

$$D_{cone}[m] = 2 * \tan(0.2182 * altitude[m]) \quad (1)$$

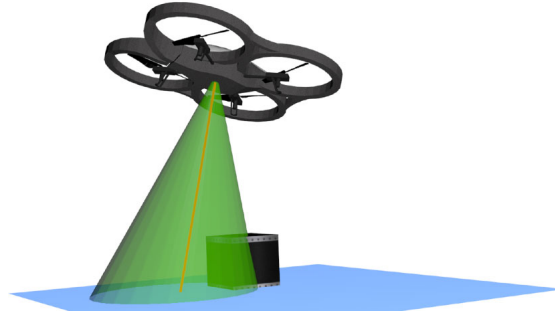


Figure 2. Overview of the ultrasound sensor cone: the green cone indicates the range of the ultrasound sensor [7].

2.2.3. On-Board Cameras and Calibration Procedure

The AR.Drone 2.0 has two cameras. The bottom camera, which uses a CMOS sensor, has an image size of 320×240 pixels. It is the fastest camera with a speed of 60 FPS and an angle of vision approximately 64° . The frontal camera can be used for difference sizes: 1280×720 pixels or 640×360 pixels; the speed is 30 FPS, and the angle of vision is $80\text{--}90^\circ$, approximately; this big angle of vision is responsible for the radial distortion in the images, similar to a fish-eye effect.

The cameras represent the main source of information for the system. Therefore, it is important to characterize the camera experimentally, in order to define the relation between pixels and meters. The experiment consists of taking pictures of a reference object of known characteristics (*i.e.*, height, width and color) at different distances. A fitting procedure is performed in order to characterize the cameras using the experimental as depicted in Figure 3, thus obtaining a relation which allows to compute the distance between the quadrotor and a reference object using Equations (2) and (3).

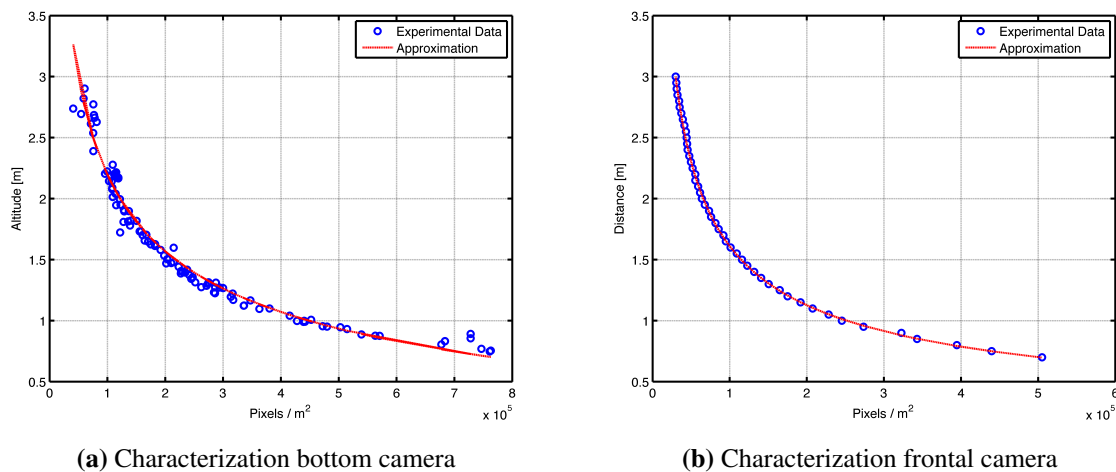


Figure 3. Camera characterization. Experimental data and approximation obtained for the (a) bottom and (b) frontal camera.

$$Altitude [m] = 148.6 \left(\frac{Pixels}{Area} \right)^{-0.339} - 0.8036 \quad (2)$$

$$Distance [m] = 599.3 \left(\frac{Pixels}{Area} \right)^{-0.5138} - 0.006038 \quad (3)$$

Since the quadrotor is equipped with low-cost general-purpose cameras, high distortion is observed when taking images. Fortunately, they can be characterized and corrected using a calibration and mapping procedure. In order to correct the distortion, it is necessary to take into account the radial and tangential factors. The presence of the radial distortion manifests in the form of the “barrel” or “fish-eye” effect, while tangential distortion occurs because the lenses are not perfectly parallel to the image plane.

In this work, we made use of the algorithms available from OpenCV libraries [16]. Currently, OpenCV supports three types of objects for calibration: asymmetrical circle pattern, a symmetrical circle pattern and a classical black-white chessboard. The method used for the elimination of the optical distortion on the images from the frontal camera of the AR.Drone 2.0 was the chessboard method. The procedure consists of taking snapshots of this pattern from different points of view (POV) of the chessboard; the algorithm implemented detects the corners and the intersections on the chessboard and creates an equation. To solve the equation, it is required to have a predetermined number of pattern snapshots to form a well-posed equation system.

In practice, due to the amount of noise present in our input images, good results were obtained using 10 snapshots of the input pattern from different positions. After solving the equation system, the parameters of the correction matrix are obtained and output as XML/YAML files.

This calibration experiment needs to be carried out only once. Then, inside the main application, once the files are loaded, a mapping function from OpenCV libraries is executed to eliminate the camera distortion. Finally, the distortion of the original image (Figure 4a) is eliminated as depicted in Figure 4b. Although a small part of the information is removed during the image processing procedure, the image is distortion free afterward.

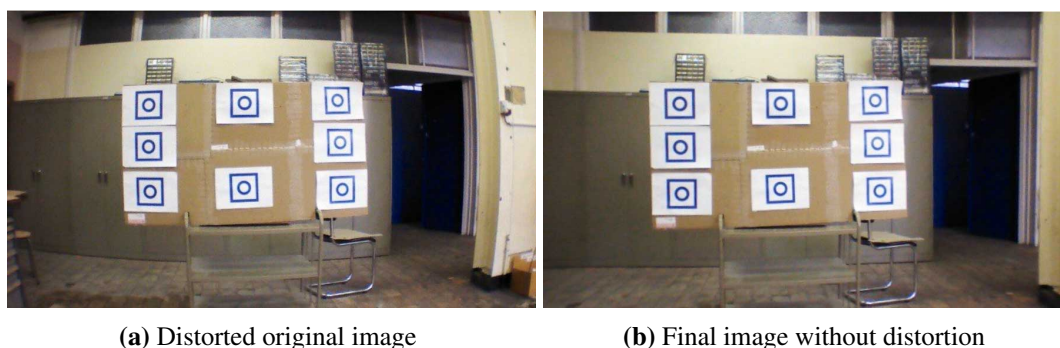


Figure 4. AR.Drone 2.0 pictures from the frontal camera: (a) original image with radial distortion; (b) image obtained after the remap process with the calculated distortion parameters.

2.2.4. Processing Unit and Communication Channels

Two main circuit boards compose the processing unit of the drone:

- The mother-board holds the 1-GHz 32-bit ARM Cortex A8 processor with 800-MHz video DSP TMS320DMC64X, running a Linux-based real-time operating system.
- The second board uses a 16-bit PIC micro-controller navigation board, which interfaces with the sensors at a frequency of 40 Hz.

Regarding communication, there are four main services to connect with the AR.Drone:

Control and configuration of the drone is realized by sending AT commands on UDP port 5556. The transmission latency of the control commands is critical to the user experience. Those commands are to be sent on a regular basis (usually 30-times per second).

Information about the drone (like its status, its position, speed, engine rotation speed, *etc.*), called navdata, are sent by the drone to its client on UDP port 5554. These navdata also include tag detection information that can be used to create augmented reality games. They are sent approximately 30-times per second.

A video stream is sent by the AR.Drone to the client device on port 5555 with the TCP protocol. Given this protocol has a confirmation step, it presents a video streaming time delay of 360 ms, approximately. Image and video resolution can be selected between 360 p and 720 p. However, changing the video to 720 p creates a very noticeable lag between real time and the video. There was about a two-second delay. Images from this video stream can be decoded using the codec included in this SDK. The embedded system uses a proprietary video stream format, based on a simplified version of the H.263 UVLC (Universal Variable Length Code) format.

A fourth communication channel, called the control port, can be established on TCP port 5559 to transfer critical data, in opposition to the other data that can be lost with no dangerous effect. It is used to retrieve configuration data and to acknowledge important information, such as the configuration.

2.3. Position Estimation Using Sensor Fusion

Depending on the application, there are two possibilities to estimate the position of the quadrotor: (1) from camera images to situate the quadrotor on the X, Y plane; and (2) the AR.Drone provides an estimation of the translational speeds by using its on-board sensors and an optical flow algorithm, making it also possible to estimate the position by integrating the mentioned speeds.

Figure 5 describes the sensor's possibilities to estimate the position on the (X, Y) plane, each having positive and negative characteristics. On the one hand, odometry allows position estimation with almost no delay, but it suffers from drifting, producing an error that increases with time. On the other hand, to compute the position estimation based on optical measurements requires the use of patterns located in known positions, introducing also the problem of the additional time delays and noisy signals due to varying light conditions. An additional obvious difficulty is that once the pattern is out of the image, it is not possible to estimate the position.

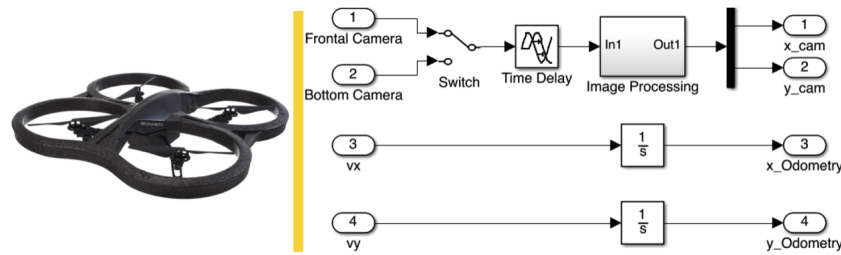


Figure 5. Sensors used for position estimation of the AR.Drone in the x, y plane.

The solution to achieve a reliable position estimation consists of combining the information from the two sensors. In order to reduce the drift effect and noise, odometry is used to read the variations and the optical sensor to find an offset-free measurement. The simplest and functional combination consists of using the optical sensor only when the standard deviation of the last five samples obtained from odometry is bigger than a tolerance value.

A time delay of about 100 ms is present due to latency in the wireless communication. However, the video signal has an additional time delay present in the video channel, which is directly related to the amount of data to be sent. For example, higher camera resolution introduces larger delays (*i.e.*, 330 ms approximately for an image of 640×360 pixels).

The position obtained from the camera represents the offset-free position, but “ n ” samples before, where “ n ” represents the time delay in samples (*i.e.*, $n = 5$ with $T_s = 66$ (ms)). Next, it is possible to integrate the speed values of the last five samples, in order to obtain the position estimation from odometry up to time “ $n-1$ ”. Equation (4) describes the method presented to eliminate the time delay effect on the video signal using a combination with odometry, assuming the dead time is a multiple of the sample time.

$$x_{(k)} = x_{cam}(N_d) + T_s \sum_{k=-(N_d-1)}^{k=0} v_{x(k)} \quad (4)$$

where x is the final position in meters, $N_d = T_d/T_s$, and T_s is the sample time: 66 ms; k represents the samples; v_x is the speed on the x axis; and x_{cam} represents the position obtained from the camera with a constant time delay $T_d = 330$ ms.

Figure 6 presents the performance of the estimation obtained, after using the proposed data fusion to correct the position measurements in the “ Y ” axis.

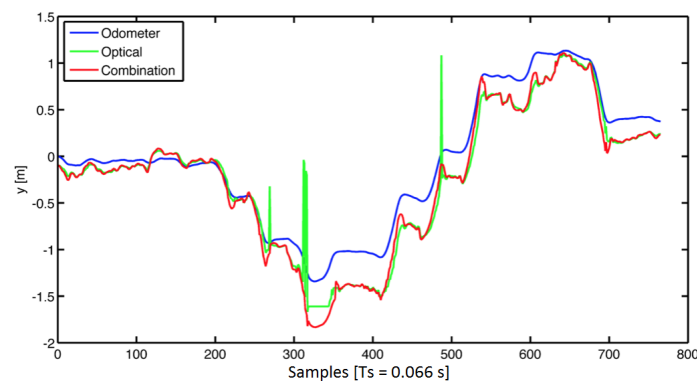


Figure 6. Position values in an open loop obtained from the image processing (green), the odometry (blue) and the fused response (red).

3. Quadrotor Dynamics and Identification

3.1. Coordinates System

The quadrotor's aerial movements are similar to those of a conventional helicopter. The quadrotor has four degrees of freedom (DOF): rotation over pitch, roll and yaw and translational movements over x , y and z , as depicted in Figure 7a. Notice that through rotational movement along the transversal y axis (pitch), translational movement on the x axis is achieved. A similar conclusion can be drawn for rotation over roll and translational movement on y .

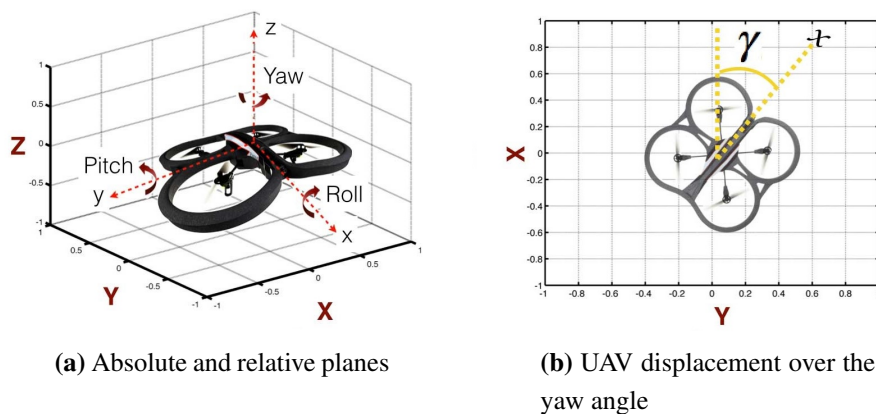


Figure 7. UAV axes: (a) difference between absolute axes (X, Y, Z) and relative axes (x, y, z); (b) UAV displacement on the (x, y) plane with respect to the absolute plane.

It is worth noting that the coordinate system described above (x, y, z), represents a relative coordinate system used by the internal controllers (low layer). Using such a coordinate system instead of absolute coordinates (e.g., X, Y, Z) in the high layer will yield errors. For example, notice that by rotating the quadrotor, the relative coordinates (x, y) will change with respect to the absolute coordinates, as depicted in Figure 7b.

Concerning the relationship between the relative and the absolute coordinate systems, four cases were analyzed (Figure 8). The first case represents a null angular displacement on the UAV orientation with respect to the absolute space, thus meaning that the speeds in the relative axes are the same as those of the absolute axes: “ $V_X = v_x$ ” and “ $V_Y = v_y$ ”. Cases 2 and 4 represent an angular displacement of 90° and -90° , respectively. The third case represents an angular displacement of -180° , which implies an opposite effect in the “ x ” and “ y ” axes.

Clearly, the relationship between the two coordinate systems is defined by the gamma (γ) angle, defined in Figure 7b. Consequently, equations describing the speeds of the UAV in the absolute system are defined as a function of the speed in the relative coordinates and (γ), as follows:

$$V_X = v_x \cos(\gamma) - v_y \sin(\gamma) \quad (5)$$

$$V_Y = v_x \sin(\gamma) + v_y \cos(\gamma) \quad (6)$$

After integrating the “ V_X ” and “ V_Y ” absolute speeds, it is possible to estimate the position of the UAV in the 3D space; this procedure is known as odometry. It is also important to note that Equation (6) depends on the yaw angle, which suffers from drifting over time, thus producing a “biased” estimation.

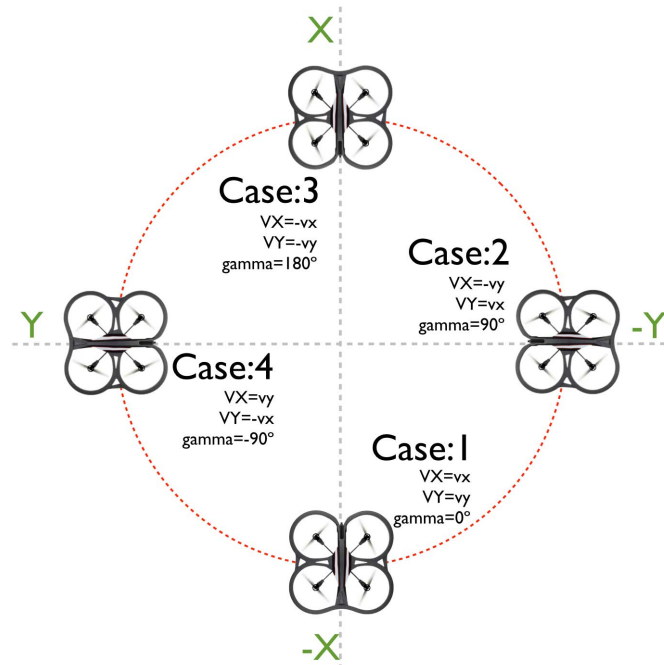


Figure 8. Case studies to describe the relation between the absolute and the relative coordinate systems.

3.2. System Identification

Due to the internal control, the quadrotor behaves as a set of single-input single-output (SISO) systems, therefore making it possible to perform parametric identification on each degree of freedom. This is realized using the prediction error method [17] and a pseudo-random binary signal (PRBS) as the excitation signal. A sampling time of 5 ms for yaw and 66 ms for other degrees of freedom are chosen based on the analysis of dynamics performed in a previous work [18]. The transfer functions obtained are given by:

$$\begin{aligned}
 \frac{v_x(s)}{v_x^*(s)} &= \frac{7.27}{(1.05s + 1)} e^{-0.1s} \\
 \frac{v_y(s)}{v_y^*(s)} &= \frac{7.27}{(1.05s + 1)} e^{-0.1s} \\
 \frac{v_z(s)}{v_z^*(s)} &= \frac{0.72}{(0.23s + 1)} e^{-0.1s} \\
 \frac{\dot{\gamma}(s)}{\dot{\gamma}^*(s)} &= \frac{2.94}{(0.031s + 1)} e^{-0.1s}
 \end{aligned} \tag{7}$$

where $\dot{\gamma}$ is the angular speed in yaw. The time delay in Equation (7) represents the average time delay present due to the communication in the control channel, *i.e.*, the action on the motors is received approximately 100 milliseconds after it is set on the computer.

Notice that Equation (7) corresponds to the identification of the closed-loop system using information coming from the IMU board. The inputs are the setpoints for speed (*i.e.*, v_x^* , v_y^* , v_z^* , $\dot{\gamma}^*$), and the outputs are the response of the internal control to follow those setpoints (*i.e.*, v_x , v_y , v_z , $\dot{\gamma}$). In other words, what is being identified is the closed-loop dynamics of the quadrotor for each degree of freedom.

4. Path-Following Predictive Control

A robust position controller of the quadrotor is required to follow either a set of way-points or a trajectory and to reject disturbances efficiently. Based on previous work [19,20], it has been found that model predictive control (MPC) fulfills the required specifications for tasks of tracking and positioning in 3D space [21]. In this section, the control structure and the implemented position controller using MPC is introduced.

4.1. EPSAC-MPC Algorithm

MPC refers to a family of control approaches, which makes explicit use of a process model to optimally obtain the control signal by minimizing an objective function [22]. In this contribution, the extended prediction self-adaptive control (EPSAC) approach to MPC is briefly described; for a more detailed description, the reader is referred to [23].

A typical set-up for the MPC optimization problems is as follows:

$$\Delta U = \arg \min_{\Delta U \in \mathbb{R}^{N_u}} \sum_{k=N_1}^{N_2} [r(t+k|t) - y(t+k|t)]^2 \quad (8)$$

for $k = N_1 \dots N_2$, where N_1 and N_2 are the minimum and maximum prediction horizons, ΔU is the optimal control action sequence, N_u is the control horizon, $r(t+k|t)$ is a future setpoint or reference sequence and $y(t+k|t)$ is the prediction of the system output.

In EPSAC, the predicted values of the output are given by:

$$y(t+k|t) = x(t+k|t) + n(t+k|t) \quad (9)$$

Then, it follows that $x(t+k|t)$ is obtained by recursion of the process model; using the control input $u(t+k|t)$ and $n(t+k|t)$ represents the prediction of the noise, which includes the effect of the disturbances and modeling errors.

A key element in EPSAC is the use of base and optimizing response concepts. The future response can then be expressed as:

$$y(t+k|t) = y_{base}(t+k|t) + y_{optimize}(t+k|t) \quad (10)$$

The two contributing factors have the following origin:

- $y_{base}(t+k|t)$ is the effect of the past inputs, the *a priori* defined future base control sequence $u_{base}(t+k|t)$ and the predicted disturbance $n(t+k|t)$.
- $y_{optimize}(t+k|t)$ is the effect of the additions $\delta u(t+k|t)$ that are optimized and added to $u_{base}(t+k|t)$, according to $\delta u(t+k|t) = u(t+k|t) - u_{base}(t+k|t)$. The effect of these additions is

the discrete time convolution of $\Delta U = \{\delta u(t|t), \dots, \delta u(t + N_u - 1|t)\}$ with the impulse response coefficients of the system (G matrix), where N_u is the chosen control horizon.

Once Equation (10) is obtained, then Equation (8) can be solved, thus obtaining the optimal control action that minimizes the cost function. At the next sampling instant, the whole procedure is repeated, taking into account the new measured outputs according to the receding horizon principle, thus introducing feedback into the control law.

4.2. Performance of Position Control

Using the identified model of the quadrotor Equation (7), the EPSAC controller is tuned to achieve the shortest settling time without overshoot. The tuning parameters for the proposed controller are presented in Table 1.

Table 1. Design parameters for the extended prediction self-adaptive control (EPSAC) controllers. SISO, single-input single-output.

SISO System	N_1	N_2	N_u
x, y	1	15	1
z	1	30	1
γ	1	10	1

Furthermore, the position controller is tested for path-following as depicted in Figure 9. It is observed that the controller is able to follow the trajectory with a small tracking error and almost no overshoot.

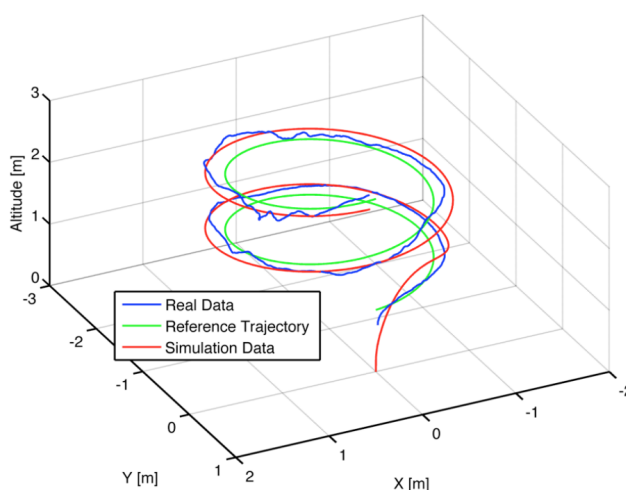


Figure 9. 3D simulated and real response for path following of the AR.Drone 2.0 using EPSAC-MPC on (X, Y and Z) degrees of freedom.

It is also important to highlight that the controllers are expected to work for the absolute coordinate system (X, Y); therefore, a transformation of the control action given by the MPC controllers to the

relative coordinate system (x, y) is still necessary. This is performed based on Equations (5) and (6), thus determining the control actions as:

$$v_x^* = V_X \cos(\gamma) + V_Y \sin(\gamma) \quad (11)$$

$$v_y^* = -V_X \sin(\gamma) + V_Y \cos(\gamma) \quad (12)$$

where V_X and V_Y represent the absolute control action on the X and Y axis, respectively, at each sample time T_s .

5. Application of Mapping for Precision Agriculture

In this study, we focus on improving the loading process in a forage harvester, by measuring the volume of grain inside a container in real time. This application lies in the field of mapping and precision agriculture.

5.1. Loading Process during Forage Harvesting

Corn silage is a high-quality popular forage for ruminant animals. During this process, the corn on the ground is extracted, chopped and then ensiled. Using combine harvesting machines for forage processes is a very common and demanding operation in agriculture. It requires two operators to operate the combine harvester and the tractor next to it. The forage is discharged from the harvester via a spout, where an orbital motor drives the spouts rotational movement. On the end of the spout, a flipper ensures that the forage is discharged accurately into the trailer. Figure 10 presents a combine harvester machine in a conventional loading process.



Figure 10. Combine harvester machine in a conventional loading process.

The success of the process depends on the two operators and their ability to maintain the correct position of the machines. Good synchrony allows one to reduce the loss of material and to optimize the trailer's space through efficient filling, thus ensuring an ideal flat material profile in the container. However, to achieve this synchronization, the harvester driver must focus on two tasks at once: (1) align the machine with respect to the tractor and the crop; and (2) adjust the actions to manipulate the spout with two degrees of freedom to send the material towards the container in the best way possible.

Automation of this process would enable operators to accurately drive the harvester while the system automatically fills the trailer, disregarding the distance or position of the two vehicles and even when

visibility is limited. Consequently, the driver would benefit from improved operating comfort and safety, focusing only on the front of the machine, thus increasing the working rate.

In order to automate this process, the system must perform some special tasks: read arm angles, read the distance between the harvester and the trailer and make a 3D map of the material inside the trailer. In this work, we focus on producing a reliable 3D map of the volume of grain inside the trailer. This information can later be used directly for the operator or by a controller to guide the spout to the interest point of the lower height of the material to get a flat profile. Companies in the field have proposed several alternatives, which are briefly described in the next subsection.

5.2. Commercial Assistant Systems

Several alternatives (academic and commercial) have emerged to automate and improve the performance of the loading process. Here is included a short description for the three most representative solutions.

5.2.1. Asuel

Asuel is prototype intended for position control of the spout of a forage harvester during the loading process [24]. The main objective is to build a mathematical model to estimate the volume and shape of the material inside the trailer. Estimation of the volume is corrected using information from Global Positioning Systems (GPS) and a model of the different actuators. The accuracy of the estimated profile is very low, given that it is difficult to obtain a good mathematical model, with the additional low resolution of GPS.

5.2.2. Auto Fill

Autofill is a stereo camera-based spout control system [25] developed by Claas company. Using a fixed stereo camera has a significant advantage, allowing 3D perception compared to the traditional 2D images. When a trailer approaches the side of the forage harvester, the vision system detects its position. Once the trailer is detected, an overlay is drawn on the picture and shown to the driver. Then, a green line showing the estimated material level is drawn, thus engaging the AutoFill system. The system predicts where the crop jet will hit within the trailer using measurements of the spout and deflector rotations. Due to crop conditions and drift, the precision of the predicted hit point is not sufficient. Thus, the predicted jet trajectory is corrected online by measuring the distance to the jet. For this system, the main problem arises from the disturbance created from the dust created once the material falls inside the trailer.

5.2.3. IntelliFill

Case New Holland (CNH) has recently launched a forage spout guidance called IntelliFill, which is based on a Time-Of-Flight (TOF) camera with 3D images. The optical sensor reads the light reflected back, obtaining a picture of the container filling. This camera allows the system to have functionality in the dark, as well as in bright day light [26]. The camera is mounted on the harvester's spout, and it

can measure the distance to the four corners of the container and the filling volume inside the container. Through this information, the spout's turning angles can be automatically controlled.

5.3. Possible Limitation When Using a Fixed Camera

As described above, the goal is to reduce the loss of material and to optimize the trailer's space through an efficient filling, thus ensuring an ideal flat material profile in the container. This is achieved if operators maintain the correct position of the machines and if the operator of the harvester correctly manipulates the spout to send the material towards the container in the best possible way.

Achieving a flat material profile inside the trailer is possible under good visibility conditions and flat terrains, with the additional help of a system to supervise the loading process as depicted in Figure 11a. Nevertheless, some limitations appear when placing the sensor (*i.e.*, camera) in the arm of the harvester machine. For example, noise in the images due to interference coming from dust, the small particles of chopped material and mechanical vibrations (Figure 11b) or a decrease of visibility due to an increase of the distance between the vehicles (Figure 11c). These difficulties can be diminished by using a flying sensor, because the camera can be placed in a better position, thus increasing visibility inside the trailer despite dust or a large distance between the vehicles (Figure 11d). Additional advantages can be obtained if other information is extracted from images during flight (e.g., density of the crop in front of the harvester to regulate the harvester speed properly) or if other sensors are installed.

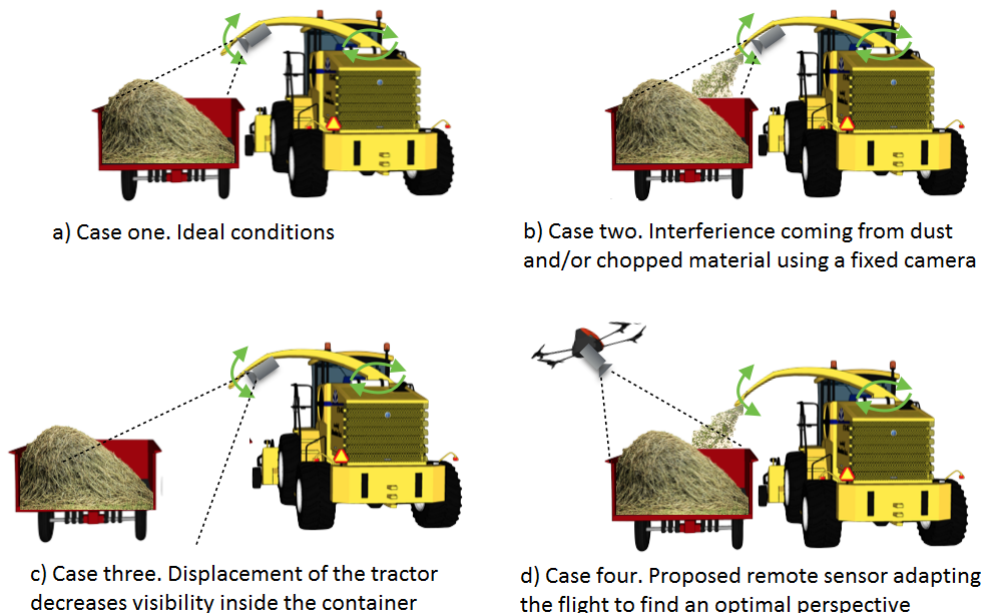


Figure 11. Possible limitations when using a fixed sensor and advantages of using a smart flying sensor for a loading application during forage harvesting.

5.4. Proposed Alternative by Using a UAV

A solution to the overloading problem could be the use of an UAV acting as a remote sensor, as depicted in Figure 11d. The quadrotor should follow the vehicles, read the profile disposition inside

the container and, through image processing, detect the relative distance between the harvester and the trailer, *i.e.*, to minimize forage losses during the discharging process. A simple lab-scale system is utilized as proof-of-concept of the proposed solution. Figure 12 shows the setup platform used to emulate the tractor-trailer with the material. The emulated container has $2.0 \times 1.5 \times 1.0$ m for length, width and height, respectively.

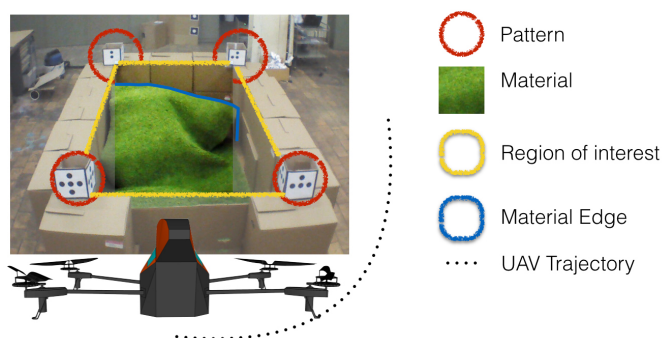


Figure 12. Experimental setup description.

The UAV is in charge of collecting information from the process; in other words, it must follow the container and read the profile of the material. The patterns in the corners of the container are placed to provide references to the quadrotor and to delimit the region of interest. Although color patterns can be used, a higher robustness was observed with black and white patterns, given they have less variations for different light conditions.

6. Experimental Validation of the Proposed Smart Flying Sensor

6.1. Structure of the Proposed Solution

The experiment consists of flying with the quadrotor around the emulated container referenced by the patterns on the corners. Once the view is focused on a point of view (POV), the container is segmented by using triangulation with the pattern references. Subsequently, color segmentation can be applied to identify the material profile. Note that at least two pictures are needed to reconstruct a 3D surface representing the relief of the material.



Figure 13. Methodology to build a 3D map of the material inside the trailer during a loading process using a forage harvester.

The complete procedure to build a 3D map of the material inside the container can be represented by the scheme in Figure 13. A description of the steps for the application are hereafter described:

- Take-off and hold on: First of all, the quadrotor must be stabilized in space. Therefore, after the take-off, the quadrotor goes to an altitude setpoint of two meters (*i.e.*, $z^* = 2$ m) and tries to hold the position.
- Read the sensors: Once the quadrotor is hovering, the next step is to use the front camera and to take a picture in order to activate the image recognition process. It is important to mention that the camera is at about -45° with respect to the horizontal position. This is done in order to have a better view inside the container.
- Reference the quadrotor: The distance between the UAV and the container is computed using information from the patterns. An image processing algorithm is in charge of computing the area inside the square of the pattern. Since this value is predefined and therefore known, the relative distance between the container and the UAV can be computed using Equation (3) and the time delay correction Equation (4). The controllers are designed to keep a fixed distance between the UAV and the container, centered on the container wall in a similar manner as depicted in Figure 12.
- Read the area of interest: The controllers maintain the quadrotor at a fixed 3D point with respect to the container. When the position error is low and the angles (pitch and roll) are appropriate, the picture is reduced only to the inner container size Figure 14a. Here, color segmentation is used to extract the projection of the material on the walls by using edge detection techniques, as shown in Figure 14b.

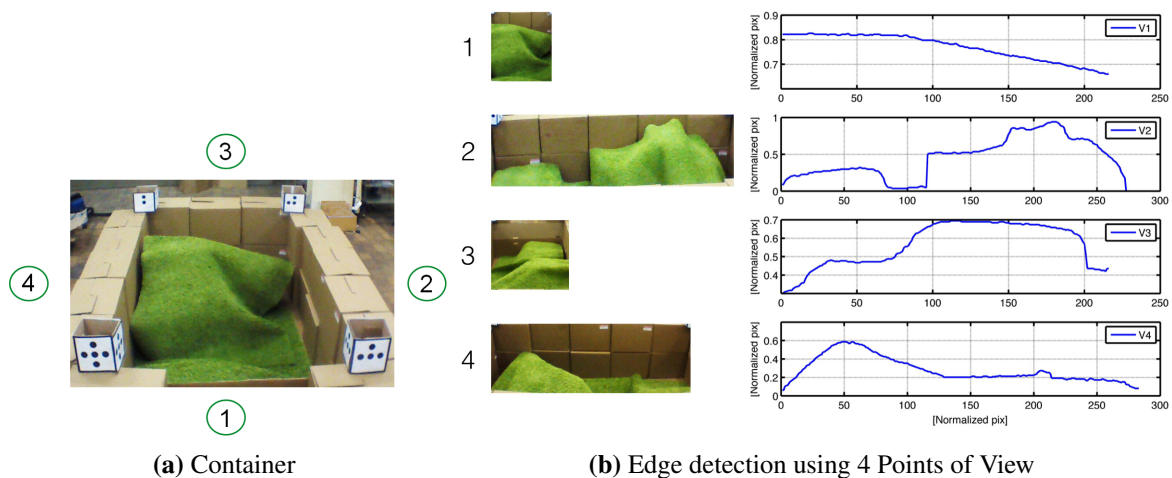


Figure 14. Example of (a) material profile in the container and (b) the information extracted using the edge detection algorithm considering four points of view (POV).

- Change the point of view (POV): In computer vision, the methods used to know depth information are based on the combination of images taken from different points of view (e.g., stereo vision). Given the flexibility of the UAVs, since a path-control algorithm using MPC has been implemented, it is possible to fly to other positions in order to get pictures from different points of view. The next step is thus to fly around the container pointing the camera inward, following a trajectory in four degrees of freedom in space, similar to the experiment performed in Figure 9, but using a constant altitude. The trajectory is calculated as a second order polynomial, knowing the

actual coordinates and the destination point by using the translation between absolute and relative coordinate systems presented in Equations (11) and (12).

- Extract the information: Once the pictures have been taken, a correction on the pictures was implemented to remove the “fish eye” effect present on the native photos. Then, the material is segmented, and the edge function is applied to calculate the contours of the material for each picture. These contours are in the form of vectors (e.g., V_1 , V_2 , V_3 and V_4) containing the information about the shape of the material on the container’s wall. Figure 14 shows the material and its corresponding contour for four different points of view.

6.2. 3D Profile Computation

Image processing is composed of two main parts: the first part corresponds to the pre-processing (*i.e.*, acquisition, segmentation and classification), which is executed in OpenCV at the same sample time as in the UAV; the second part used to compute 3D map is executed in parallel in MATLAB with the process, but using a longer sampling time. Following MATLAB’s notation, the surface obtained using four vectors (see Figure 14b) corresponding to the edge of the material inside the container is:

$$S = 0.25 * [V_1(end : -1 : 1)' * V_2 + V_1(end : -1 : 1)' * V_4 + V_3' * V_2 + V_3' V_1(end : -1 : 1)] \quad (13)$$

A trade-off between accuracy and update time of the 3D map must be considered. Although using four images (four POV) has the advantage that a more accurate profile is obtained, this also implies a longer update time, since the UAV will require more time to take all pictures. A possible solution consists of using only two pictures, for which a good approximation of the 3D map can be computed, as depicted in Figure 15.

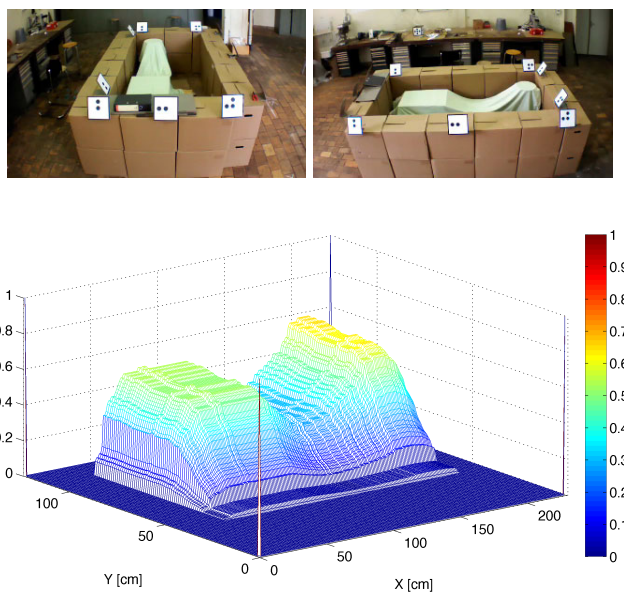


Figure 15. 3D profile obtained experimentally using the smart flying sensor and two pictures.

6.3. Performance of the Path-Following Controller

An important element for this application is the path-following controller, which is required to guarantee an accurate position of the quadrotor in the 3D space and to reject possible wind disturbances. Using the controller designed in Section 4, the quadrotor is able to automatically take-off and follow a pre-defined path around the container while taking the necessary number of pictures to compute the 3D profile.

The performance of the controller for the case of taking two pictures to approximate the material profile is depicted in Figure 16, including the final path followed by the UAV in Figure 16a and the control actions (i.e., setpoints to the low-layer internal controllers) required by the MPC in Figure 16b.

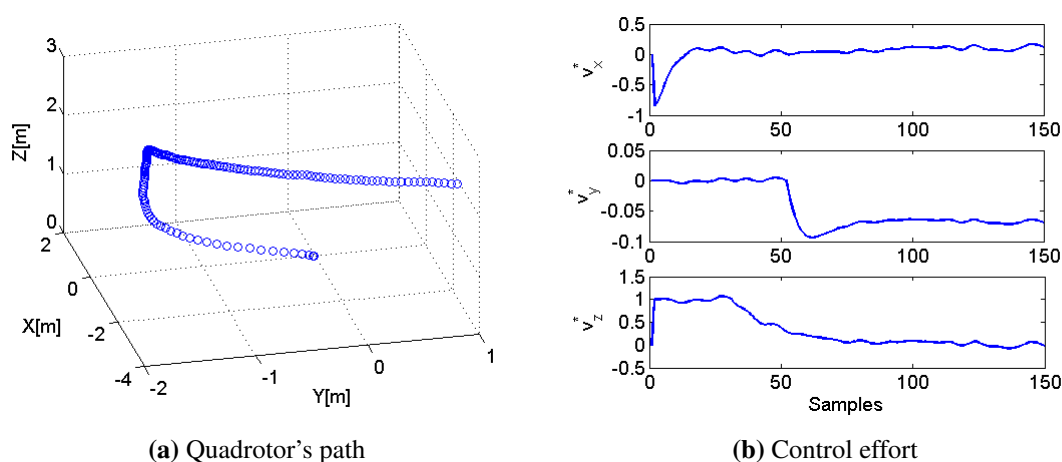


Figure 16. Performance of the path-following controller. (a) Quadrotor's path around the container in order to take two pictures and (b) control effort required by the MPC strategy.

7. Conclusions

In this paper, we have presented the first steps towards the development of a smart flying sensor based on an unmanned aerial vehicle (UAV). The effectiveness of the proposed smart flying sensor is illustrated for the task of mapping the volume of grain inside a trailer during forage harvesting, using a lab-scale system.

The main achievements for this research are: (i) the obtained insight in the dynamics and coordinate systems of the low-cost quadrotor AR.Drone 2.0; (ii) the development of a position-estimation method with time delay compensation based on image processing and the inertial measurement unit (IMU); (iii) a method to build a 3D map using information obtained from a regular camera; and (iv) the design and implementation of a path-following control algorithm using MPC.

Future work includes incorporating a GPS for outdoor flight, together with the development of obstacle avoidance techniques to enhance the autonomy of the quadrotor. An extension to multiple UAVs and/or a combination with ground vehicles is also under investigation.

Acknowledgments

This work has been financially supported by the EeSA department (Electrical Energy, Systems and Automation) at Ghent University (Belgium). This work has been obtained as a result of the bilateral agreement between Ghent University (Belgium) and Universidad de Ibaguè (Colombia), in the framework of the Masters in Control Engineering. Andres Hernandez acknowledges the financial support provided by the Institute for the Promotion and Innovation by Science and Technology in Flanders (IWT-Vlaanderen), Belgium (grant nr. SBO-110006).

Author Contributions

Andres Hernandez contributed to the implementation of the communication between the quadrotor and the main station, besides performing the system identification and the design of the path-following control algorithm using model predictive control.

Harold Murcia developed the algorithm for time delay compensation using sensor fusion and image processing algorithms, contributed to the design and implementation of the control algorithms and performed simulations and experiments.

Cosmin Copot and Robin De Keyser contributed to the formulation and analysis of the stated problem, collaborated with advice in control design and image processing and with the analysis of the results.

All authors took part in the writing of the manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Sahota, H.; Kumar, R.; Kamal, A.; Huang, J. An energy-efficient wireless sensor network for precision agriculture. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Washington, DC, USA, 22–25 June 2010; pp. 347–350.
2. Blackmore, S. Precision Farming: An Introduction. *J. Outlook Agric.* **1994**, *23*, 275–280.
3. Siuli Roy, A.; Bandyopadhyay, S. Agro-sense: Precision agriculture using sensor-based wireless mesh networks. In Proceedings of the Innovations in NGN: Future Network and Services, First ITU-T Kaleidoscope Academic Conference, Geneva, Switzerland, 12–13 May 2008; pp. 383–388.
4. Berni, J.; Zarco-Tejada, P.J.; Suárez, L.; Fereres, E. Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle. *IEEE Trans. Geosci. Remote Sens.* **2009**, *47*, 722–738.
5. Bouffard, P. On-board Model Predictive Control of a Quadrotor Helicopter: Design, Implementation, and Experiments. Master's Thesis, University of California, Berkeley, CA, USA, 2012.
6. Dullerud, G. Modeling, Identification and Control of a Quad-Rotor Drone Using Low-Resolution Sensing. Master's Thesis, University of Illinois at Urbana-Champaign, Champaign, IL, USA, 2012.

7. Dijkshoorn, N.; Visser, A. An elevation map from a micro aerial vehicle for urban search and rescue. In Proceedings of the 16th RoboCup International Symposium, Mexico City, Mexico, 18–24 June 2012.
8. Mogenson, M.N. The AR Drone LabVIEW Toolkit: A Software Framework for the Control of Low-Cost Quadrotor Aerial Robots. Master's Thesis, TUFTS University, Boston, MA, USA, 2012.
9. Ritz, R.; Muller, M.; Hehn, M.; D'Andrea, R. Cooperative quadcopter ball throwing and catching. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Algarve, Portugal, 7–12 October 2012; pp. 4972–4978.
10. Barrows, G. Future visual microsensors for mini/micro-UAV applications. In Proceedings of the IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA 2002), Washington, DC, USA, 22–24 July 2002; pp. 498–506.
11. Clement, A. Advances in Remote Sensing of Agriculture: Context Description, Existing Operational Monitoring Systems and Major Information Needs. *Remote Sens.* **2013**, *5*, 949–981.
12. Pajares, G. Overview and Current Status of Remote Sensing Applications Based on Unmanned Aerial Vehicles (UAVs). *J. Photogramm. Eng. Remote Sens.* **2015**, *81*, 281–329.
13. Zarco-Tejada, P.; Hubbard, N.; Loudjani, P. Precision Agriculture: An Opportunity for EU Farmers—Potential Support with the CAP 2014–2020; Technical Report, Joint Research Centre (JRC) of the European Commission; Monitoring Agriculture ResourceS (MARS) Unit H04: Brussels, Belgium, 2014.
14. Mesas-Carrascosa, F.; Rumbao, I.; Berrocal, J.; Porras, A. Positional quality assessment of orthophotos obtained from sensors onboard multi-rotor UAV platforms. *Sensors* **2014**, *14*, 22394–22407.
15. Bristeau, P.; Callou, F.; Vissiere, D.; Petit, N. The Navigation and Control Technology inside the AR.Drone Micro UAV. In Proceedings of the 18th IFAC World Congress of Automatic Control, Milano, Italy, 28 August–2 September 2011; pp. 1477–1484.
16. OpenCV-Documentation. Available online: <http://docs.opencv.org> (accessed on 23 November 2013).
17. Ljung, L. *System Identification: Theory for the User*; Prentice Hall PTR: London, UK, 1999.
18. Vlas, T.E.; Hernandez, A.; Copot, C.; Nascu, I.; de Keyser, R. Identification and Path Following Control of an AR.Drone Quadrotor. In Proceedings of the 17th International Conference on System Theory, Control and Computing, Sinaia, Romania, 11–13 October 2013.
19. Murcia, H.F. A Quadrotor as Remote Sensor for Precision Farming: A Fill-Harvesting Case Study. Master's Thesis, Ghent University, Ghent, Belgium, 2014.
20. Hernandez, A.; Copot, C.; Cerquera, J.; Murcia, H.; de Keyser, R. Formation Control of UGVs Using an UAV as Remote Vision Sensor. In Proceedings of the 19th IFAC World Congress of the International Federation of Automatic Control, Cape Town, South Africa, 24–29 August 2014.
21. Hernandez, A.; Murcia, H.F.; Copot, C.; De Keyser, R. Model Predictive Path-Following Control of an AR.Drone Quadrotor. In Proceedings of the XVI Latin American Control Conference (CLCA'14), Cancun, Quintana Roo, Mexico, 14–17 October 2014; p. 31.
22. Camacho, E.F.; Bordons, C. *Model Predictive Control*, 2nd ed.; Springer-Verlag: London, UK, 2004; p. 405.

23. De Keyser, R. Model Based Predictive Control for Linear Systems. Available online: <http://www.eolss.net/sample-chapters/c18/e6-43-16-01.pdf> (accessed on 7 July 2015).
24. Happich, G.; Harms, H.H.; Lang, T. Loading of Agricultural Trailers Using a Model-Based Method. *Agric. Eng. Int. CIGR J.* **2009**, *XI*, 1–13.
25. Möller, J. Computer Vision—A Versatile Technology in Automation of Agricultural Machinery. *J. Agric. Eng.* **2010**, *47*, 28–36.
26. Posselius, J.; Foster, C. Autonomous self-propelled units: What is ready today and to come in the near future. In Proceedings of the 23rd Annual Meeting Club of Bologna, Bologna, Italy, 9–10 November 2012.

© 2015 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/4.0/>).