A vision-based system for autonomous vertical landing of unmanned aerial vehicles

Jamie Wubben[1], Francisco Fabra[2], Carlos T. Calafate[2], Tomasz Krzeszowski[3],
Johann M. Marquez-Barja[1,4], Juan-Carlos Cano[2], Pietro Manzoni[2]
[1]*Faculty of Applied Engineering - Electronics-ICT - IDLab*
University of Antwerp, Antwerp, Belgium
[2]*Departament of Computer Engineering (DISCA)*
Universitat Politècnica de València, Valencia, Spain
[3]*Faculty of Electrical and Computer Engineering*
Rzeszow University of Technology, Rzeszow, Poland
[4]*imec, Belgium*
Email: jamie.wubben@student.uantwerpen.be, frafabco@cam.upv.es, calafate@disca.upv.es, tkrzeszo@prz.edu.pl,
johann.marquez-barja@{uantwerpen.be||imec.be}, jucano@disca.upv.es, pmanzoni@disca.upv.es

***Abstract*—Over the last few years, different researchers have been developing protocols and applications in order to land unmanned aerial vehicles (UAVs) autonomously. However, most of the proposed protocols rely on expensive equipment or do not satisfy the high precision needs of some UAV applications, such as package retrieval and delivery. Therefore, in this paper, we present a solution for high precision landing based on the use of ArUco markers. In our solution, a UAV equipped with a camera is able to detect ArUco markers from an altitude of 20 meters. Once the marker is detected, the UAV changes its flight behavior in order to land on the exact position where the marker is located. We evaluated our proposal using our own UAV simulation platform (ArduSim), and validated it using real UAVs. The results show an average offset of only 11 centimeters, which vastly improves the landing accuracy compared to the traditional GPS-based landing, that typically deviates from the intended target by 1 to 3 meters.**

I. Introduction

Recently there has been a growing interest in unmanned aerial vehicles (UAVs). Their applications are diverse, ranging from surveillance, inspection and monitoring, to precision agriculture and package retrieval/delivery.

Landing an UAV is the last and most critical stage of navigation [1]. According to statistics, the number of accidents associated to UAV landing process represent 80% of the hazard cases [2]. Therefore, improved landing techniques are being intensely explored. Furthermore, some of the applications mentioned above, such as package retrieval, require a high level of accuracy so as to make sure the UAV lands exactly on the desired target.

Previous proposals heavily rely on the Global Positioning System (GPS) and inertial navigation sensors (INS) as the main positioning systems. However, altitude data provided by the GPS is typically inaccurate, and needs to be compensated with a close-range sensor, such as a barometric pressure sensor or a radar altimeter. Despite such compensation, these methods still remain inaccurate, especially in the horizontal plane, resulting in a landing position that typically deviates from the intended one by 1 to 3 meters. Furthermore, the GPS cannot be used indoors. For these reasons, GPS and INS systems are mostly used for long range, outdoor flights having low accuracy requirements [3].

Taking the aforementioned issues into consideration, the aim of this work is to develop a novel vision-based landing system that is able to make a UAV land in a very specific place with high precision. In this paper, we address this problem by developing a solution that combines the use of a camera and ArUco markers [4], [5]. This way, the relative offset of the UAV towards the target landing position is calculated using the ArUco library [6] (based on OpenCV). After computing its relative offset, the UAV adjusts its position so as to move towards the centre of the marker and start descending, performing additional adjustments if needed.

The rest of this paper is structured as follows: in the next section we present some related works on UAV landing strategies. In section III we provide some technical details about the UAV platform used for development, highlighting the main issues and restrictions to be taken into consideration in the design of our system. Then, in section IV, we detail the methodology followed in order to track the target for landing, and how to perform the necessary calculations to adjust the UAV position. Section V describes how the different experiments were made. The main results are then presented in section VI, with appropriate discussion. Finally, section VII concludes this paper, and refers to future works.

II. Related work

Recently, different UAV landing approaches have been studied. These studies can be categorized based on the different types of landing platforms adopted. According to [7] the vast amount of landing platforms belong to one of these tree categories:

1) *Category I - fixed platforms:*
   Fixed platforms include all platforms that are stationary. This type of platforms are the easiest to land on, since the target remains stationary.
2) *Category II - moving platforms:*
   Moving platforms are defined by the ability to move with two degrees of freedom. In this case, the UAV needs to track the platform first, and then land on it.
3) *Category III - Landing on a ship:*
   The ultimate landing platform is on a moving ship. In this case there are six degrees of freedom (heave, sway, surge, yaw, roll, pitch) and, therefore, developing a safe landing approach becomes a troublesome task.

It is also possible to categorize the different landing approaches based on the sensor(s) used in the process. Many different sensors can be used such as: sonar, infrared, LIDAR, cameras, or a combination of those. In the following some approaches that rely on computer vision will be discussed.

In [8], authors succeeded in landing a real UAV on an object moving with a speed of 1 m/s (category II). A camera was used to track the position of the landing platform (xy-coordinates) and a LIDAR sensor provided detailed information about the altitude. This research work introduced a robust method to track and land on a moving object. However, the use of a LIDAR sensor discourages the solution, as it tends to be too expensive when scaling to a high number of UAVs.

Nowak et al. [9] proposed a system in which a UAV could land both at night, as well as during the day. The idea is simple yet robust and elegant: a beacon is placed on the ground. The light emitted from the beacon is then captured by a camera (without a infrared filter), and the drone moves (in the xy-plane) such that the beacon is in the centre of the picture. Once centred, the height is estimated based on the image area occupied by the beacon, and the drone's altitude is decreased in order for it to land safely.

The authors of [10] suggested another approach: reinforcement learning. In this approach, the UAV agent learns and adapts its behaviour when required. Usually, reinforcement learning takes a lot of time. To accelerate this process, a technique called Least-Squares Policy Iteration (LSPI) is used. With this method, a simulated UAV (AR100) was able to achieve a smooth landing trajectory swiftly.

In [11] an approach to landing and position control, similar to our work, was developed. Their approach was also based on OpenCV, and on recognizing a landing pattern. However, their landing pattern was not build with the use of ArUco markers. In fact, the landing pattern used, with a diameter of 45 cm, was only detected from a distance of 70 cm. Therefore, this strategy cannot be used in an outdoor environment where the altitude is typically much higher. However, in this approach, the UAV does not need to see the entire marker, which is an advantage of this scheme.

A system that can land on, and track a slow moving vehicle (180 cm/s), was developed in [12]. Indoor experiments show that the UAV used was able to successfully land on the target landing platform (which also consists of multiple ArUco markers) from a height of approximately 80 cm.

Our work differs from the former ones as we want the UAV to detect the landing area when high above the ground (height>20 meters) to compensate for possibly high GPS error values, while using cheap sensors (only a Raspberry Pi camera is needed), and still achieving very low errors in terms of landing accuracy (<20 cm).

III. UAV SPECIFICATION

Due to the vast amount and diversity of UAV models available, it is worth mentioning the actual characteristics of the UAV used for our experiments. The UAV adopted belongs to the Vertical Take-Off and Landing (VTOL) category, more commonly known as a multirotor UAV. In the experiments described in this paper, a hexacopter model is used (see Figure 1), being equipped with a remote control operating in the 2.4 GHz band, a telemetry channel in the 433 MHz band, a GPS receiver, a Pixhawk flight controller, and a Raspberry Pi with external camera (see Figure 2). The Raspberry Pi creates an ad-hoc WiFi connection in the 5 GHz band, which is used to communicate with a groundstation or with other UAVs. Below we detail the purpose and the connections between these different devices.

Fig. 1. Hexacopter used in our experiments.

A. 2.4 GHz remote control

The FrSky X8r receiver provides communication between the UAV and the remote control. This device makes it possible to fly the UAV manually. It receives signals from the remote control, and passes them to the flight controller. Furthermore, it sends basic information about the UAV to the remote control e.g. flight mode, so that the pilot is informed about the UAV state. It accomplishes these tasks by using the entire 2.4 GHz ISM band. Therefore, it becomes nearly impossible to receive/send any (2.4 GHz) WiFi signal in this band, as shown

in [13]. Hence, our communications with the ground station rely on the 5 GHz band instead.

B. 433 MHz telemetry

The telemetry channel operates at a lower frequency (433 MHz). Its purpose is sending UAV information from the flight controller to a ground station (typically to a smartphone), including data such as: heading, tilt, speed, battery lifetime, flight mode, altitude, etc. It is also able to receive instructions to follow a specific mission i.e, return to home, or perform an emergency landing.

C. Flight controller

The flight controller, in this case the Pixhawk 4, is a device that receives information from sensors, and that processes this information in order to control (low-level) the UAV's engines. In particular, information from different sensors such as GPS, barometer, magnetometer, accelerometer and gyroscope are combined in order to provide an accurate representation of the UAV state. This information is then used in order to stabilize the UAV, and make it controllable. Some of this information can also be sent via a serial link towards the Raspberry Pi. For the communication between the Raspberry Pi and the Pixhawk, the open source MAVLink protocol [14] is used. It is a lightweight messaging protocol for communicating with most open-source flight controllers, as is the case of the Pixhawk.

D. Raspberry Pi

The Raspberry Pi 3 Model B+ serves three purposes in our custom UAV. First, it runs the open source ArduSim [15] program, which controls the UAV at a high-level. This program is capable of coordinating an autonomous flight. Second, it also runs a Python application that processes the camera information (see Figure 2) using the ArUco marker library, and send the resulting information to ArduSim via a TCP connection. Finally, it is used to setup an ad-hoc network in the 5 GHz band. This network can be used to communicate with other UAVs or, in our case, with a ground station (laptop). As an alternative, the Raspberry Pi can be equipped with a 4G LTE dongle so as to operate the UAV from a remote location. However, this option is currently not supported by ArduSim.

IV. Proposed solution

The aim of this work is to make a UAV land on a specific location. This problem can be divided into three steps. In the first step the UAV has to make a course approach to the landing zone. This can be achieved through the GPS system. As stated before, the UAV will typically not be on its exact target location, but rather within an area of 1 to 3 meters near the intended landing position. The second part deals with finding the marker. The ArUco marker library [4], [5] (based on OpenCV) provides a function which takes the camera feed and returns information about the marker(s). ArUco markers resemble the well-known QR-codes, but carry less information than the latter ones (only an id), which makes them easier to detect. A typical ArUco marker consists of a black border and a 6x6 square of black and white smaller squares. There are different types of configurations (e.g. 3x3, 4x4, 7x7) known as dictionaries. A marker from a dictionary with less squares is of course easier to detect, but only a small number of ids can be provided. In this paper, dictionary "DICT_6X6_250" is used. As the name suggests, it provides 250 ids, which is more than enough for our purposes. The last part consists of descending the UAV while trying to keep it centered over the marker.

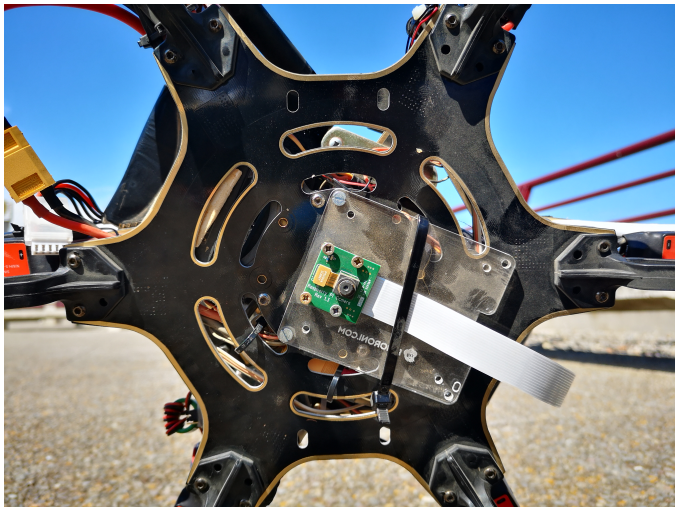

Fig. 2. Raspberry Pi camera attached to the UAV.

TABLE I
Parameter values adopted regarding Algorithm 1.

Altitude threshold z_1	0.30 m
Altitude threshold z_2	13 m
Virtual border angle α	$\{10°, 20°\}$

In order to detect a marker two conditions must be met: the marker must be fully inside of the picture, and each square must be uniquely identified (black or white). In this application, it is possible that the two conditions are not simultaneously met in some cases. For instance, when the drone is at a low altitude (i.e. 0.5m) the marker is too big to fit inside the field of view of the camera; in addition, the shadow of the drone may "corrupt" the image. On the other hand, when the drone is flying at a higher altitude (e.g. 12 m) the image may be too small to be detected. Therefore, we developed a strategy (see Figure 3) that combines markers of different sizes, so that the drone can find its target from a higher altitude. If the UAV is able to detect a smaller marker, it will switch to it, adjusting its course accordingly. Figure 4 shows a real scenario where the UAV is able to see two markers but chooses to move towards the smaller marker. The center of this marker is indicated by the red spot.

Once the marker is detected, the drone has to move towards the centre of the marker, and descend from there. Due to the effect of the wind, and to the inherent instability of the UAV

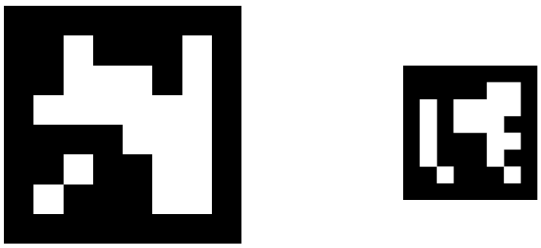

Fig. 3. Two examples of ArUco markers of different sizes.

Fig. 4. Image retrieved by the UAV camera after processing using OpenCV/ArUco libraries.

itself, the drone will also move in the horizontal plane while descending. This unwanted movement should be compensated in order to land the drone more precisely. To achieve this behavior, we propose the strategy described in Algorithm 1. In line 3, the UAV searches for an ArUco marker. If there is no marker detected. The flight mode of the UAV is changed to loiter. If this is the case for 30 consecutive seconds, the mission is aborted, and the UAV will land using GPS only. Otherwise, from the potential list of detected markers, the marker with the highest ID (i.e., the smallest marker) is selected (line 11). With the use of the ArUco library, the location of the marker with respect to the drone is estimated. If the altitude of the UAV is greater than z_2, we set α to 20 degrees; otherwise, we set it to 10 degrees (the actual parameter values, which were determined in an empirical manner, are detailed in Table I). In line 20 we check if the marker is within the virtual border (explained later). If so, the UAV descends; otherwise, it moves towards the target position. This algorithm will be executed continuously as long as the altitude of the UAV is

Algorithm 1 Vision based landing approach

1: Start timer 30 s
2: **while** altitude $> z_1$ **do**
3: Search
4: **if** $\neg$ detected **then**
5: Loiter
6: **if** timer exceeded **then**
7: Abort
8: **end if**
9: **else**
10: reset timer
11: ID $\leftarrow$ highest detected ID
12: Get $P(x, y, z)_{id}$
13: **if** z $> z_2$ **then**
14: α = 20°
15: **else**
16: α = 10°
17: **end if**
18: $\beta_1 = |\arctan(x/z)|$
19: $\beta_2 = |\arctan(y/z)|$
20: **if** $\beta_1 > \alpha$ **or** $\beta_2 > \alpha$ **then**
21: Move(x,y)
22: **else**
23: Descend(speed)
24: **end if**
25: **end if**
26: **end while**
27: Descend and disarm UAV

greater than z_1. From the moment the UAVs altitude drops below z_1 (very near to ground), the control will be handed over to the flight controller, which will land the UAV in a safe manner, and disarm the engines

In the description above, a virtual border is mentioned. This border defines an area which should enclose the marker (illustrated in Figure 5). The size a of this square is defined as:

$$a = 2 \times \tan(\alpha) \times h$$

where h refers to the relative altitude of the UAV.

The main advantages of defining the area in this way is that it will decrease as the drone lowers its altitude. Therefore, the drone will be more centred above its target position when it flies at a low altitude. However, when flying at a higher altitude, the drone should descend whenever possible to avoid excessive landing times. For this reason, α is increased to 20° if the UAV is flying above 13 meters.

There are multiple ways to move the UAV towards the target point. In ArduSim, a UAV can be moved by overriding the remote control with function "*channelOverride*". With the use of this function the UAV is moved, at a constant speed, along the roll axis (left/right) until the target point is met, and afterwards it moves along the pitch (forward/backward) axis.

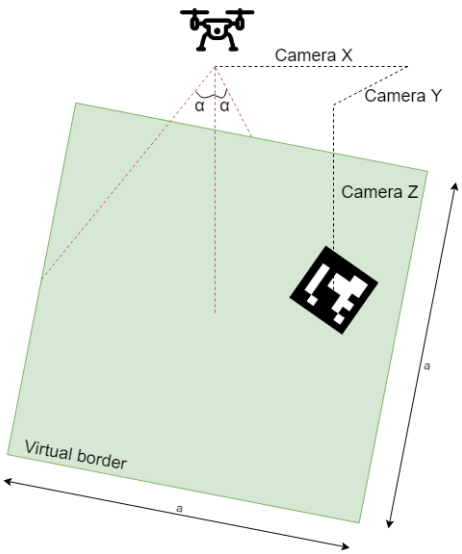

Fig. 5. Visual representation of the virtual border.

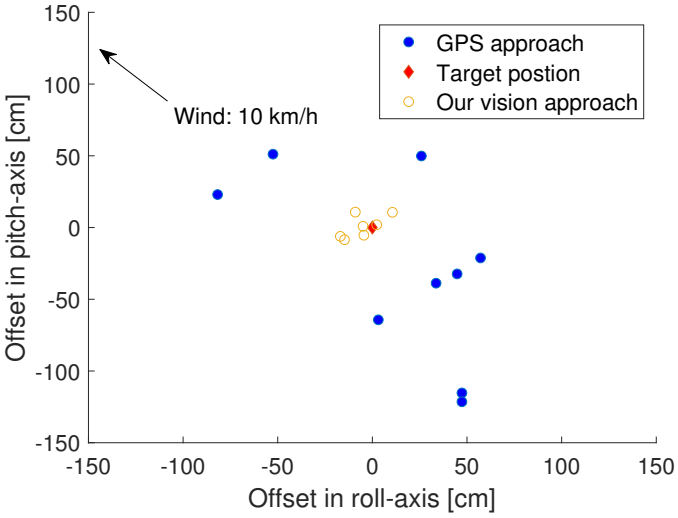

Fig. 6. UAV landing position

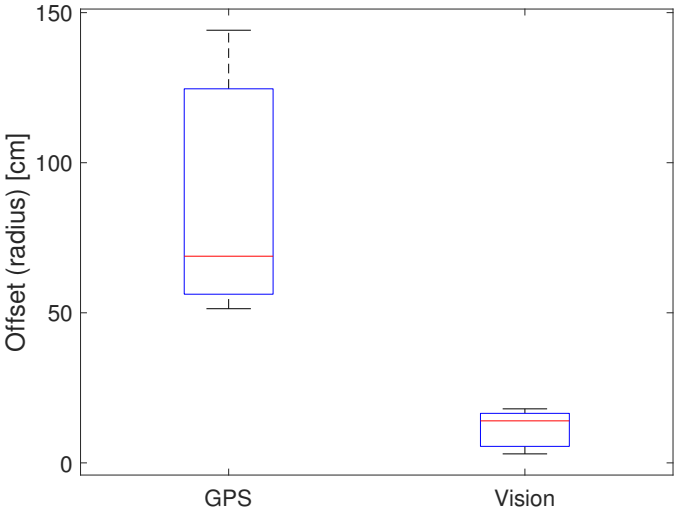

Fig. 7. Landing offset GPS vs visual based approached.

V. Experimental Settings

We performed two sets of 10 experiments in order to assess the effectiveness of our proposed solution. In the first set of experiments, the UAV was instructed to fly up to an altitude of 20 meters, to move toward a specific GPS location, and to land automatically (by giving the flight controller full control) once that position was reached. During these experiments, the landing time was recorded, as well as the actual landing position.

In the second set of experiments, again the UAV took off until an altitude of 20 meters was reached, and then flew towards the target GPS location. The largest available marker (56×56 cm) was placed at that location, and the UAV used this marker as the initial reference point for landing. When the UAV was able to detect the smaller marker (18×18 cm), it used that marker as the reference point instead. For these experiments, the descending speed was defined by lowering the throttle by 10%, and the roll and pitch values were set to a value of 5%. After each experiment, the distance between the marker and the actual landing position was recorded, along with the flight time (starting from the moment the UAV detected the largest marker until the time landing was completed); both of them were used as performance metrics.

VI. Results

Without the use of our approach, the UAV was able to land consistently within a time span ranging from 27 to 30 seconds. Nonetheless, this rapid landing comes at a price. As shown in Figures 6 and 7, the actual landing position varies substantially, ranging from a maximum error of 1.44 meters to a minimum error of 0.51 meters; the mean value for our experiments was of 0.85 meters. Notice that these errors are smaller than expected (1-3 meters). This is most likely due to the small travelled distance between the takeoff and landing locations. In fact, in these experiments, the UAV flew for only 14 meters, and flight time was of about 52 seconds. Longer flights will introduce higher errors, as reported in the literature [16].

As shown in Figure 6, the accuracy of the landing position increased substantially when the proposed solution was adopted. In particular, our experiments showed that the error ranged from 3 to 18 cm, with a mean value of 11 cm (see Figure 7). Overall, this means that our landing approach is able to reduce the landing error by about 96%. However, in three of the experiments performed, the UAV moved away from the marker due to the effect of wind. Since at these moments the altitude was already quite low, the UAV could no longer detect the marker, causing the mission to be stopped after 30 seconds. Furthermore, the average landing time was increased to 162 seconds. This is due the fact that, during the transition from one marker to another, the algorithm experienced problems at detecting the smaller marker during some time periods, as

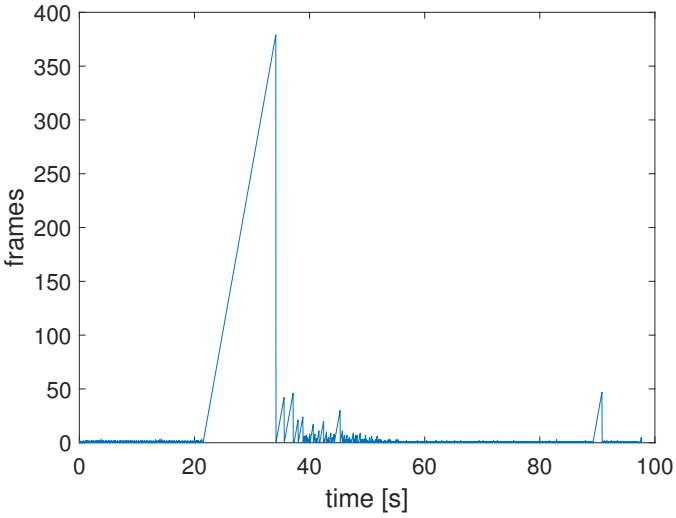

Fig. 8. Cumulative dropped frames since last detection.

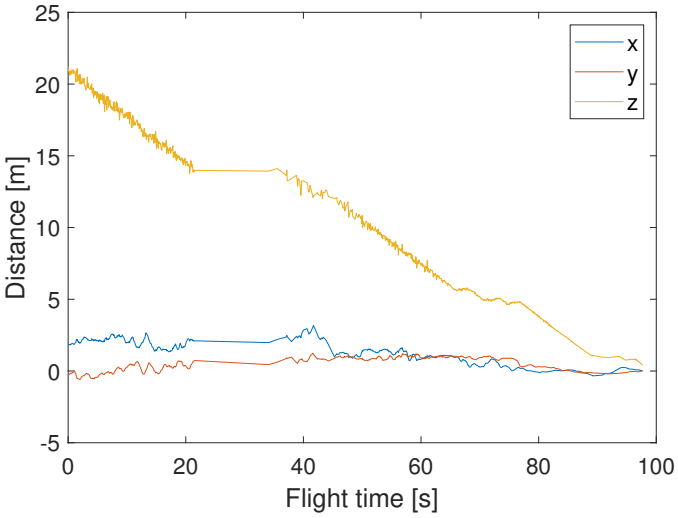

Fig. 9. Estimated distance between UAV and marker w.r.t. time.

illustrated in Figure 8.

Besides this malfunction situations, the UAV showed a smooth landing trajectory (see Figure 9). Notice how the UAV makes more aggressive adjustments in the X axis when the altitude drops below 13 meters; this is due to the fact that parameter α becomes smaller, restricting the error range. If the malfunction cases are removed, the average descending speed was of 0.3 m/s, which we find to be too conservative. Furthermore, Figure 9 shows that most of the adjustments are made when the UAV is close to the ground (constant altitude). This can be better observed in Figure 10, where the angle between the estimated altitude and the X and Y offset is plotted (X angle, Y angle). We can see that the drone only moves when the X or Y angle exceeds the value of α. The range of estimated values captured is shown in Figure 11; we can observe that there is a higher variability in the X axis due to wind compensation requirements along that direction during the experiments, something that occurs to a much lower extent for the Y axis.

Finally, an illustrative video [1] has been made available to show how our solution performs in real environments.

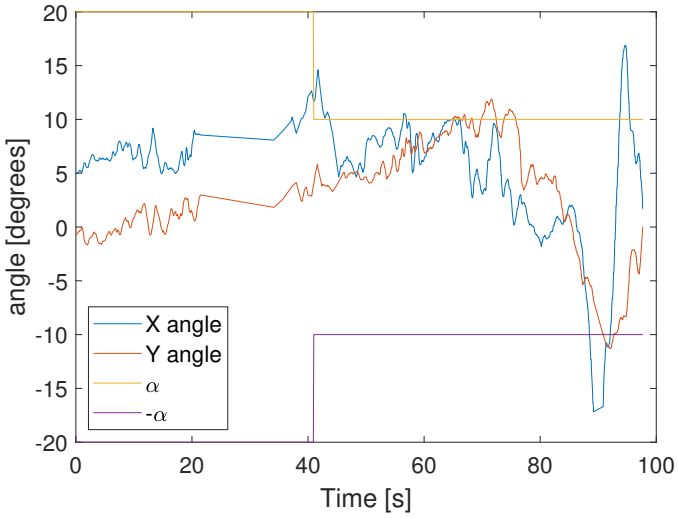

Fig. 10. X and Y angle variations vs. flight time.

VII. Conclusions and Future Work

Achieving accurate landing of multirotor UAVs remains nowadays a challenging issue, as GPS-based landing procedures are associated with errors of a few meters even under ideal satellite reception conditions, performing worse in many cases. In addition, GPS-assisted landing is not an option for indoor operations. To address this issue, in this paper we proposed a vision-based landing solution that relies on ArUco markers. These markers allow the UAV to detect the exact landing position, paving the way for sophisticated applications including automated package retrieval, or the landing of large UAV swarms in a very restricted area, among others.

We have equipped a hexacopter with a Raspberry Pi that connects to a camera and to the flight controller through a serial port. We developed our proposal for the Raspberry Pi that relies on OpenCV libraries to process in real time the images acquired by the camera. This way it is able to detect the exact landing position, instructing the flight controller on how to move so as to achieve precise landing.

Experimental results using a real UAV have validated our proposed approach, showing that accurate landing (mean error of 0.18m) can be achieved while introducing an additional time overhead in the landing procedure compared to the standard landing command. As future work we plan to improve the overall efficiency of the protocol. First and foremost, the transition between one marker and another should be made instantaneous. Besides, flight behaviour can also be improved. In particular, the following adjustments are proposed: varying the descending speed as a function of the UAV's altitude, and

[1] https://youtu.be/NPNi5YC9AeI

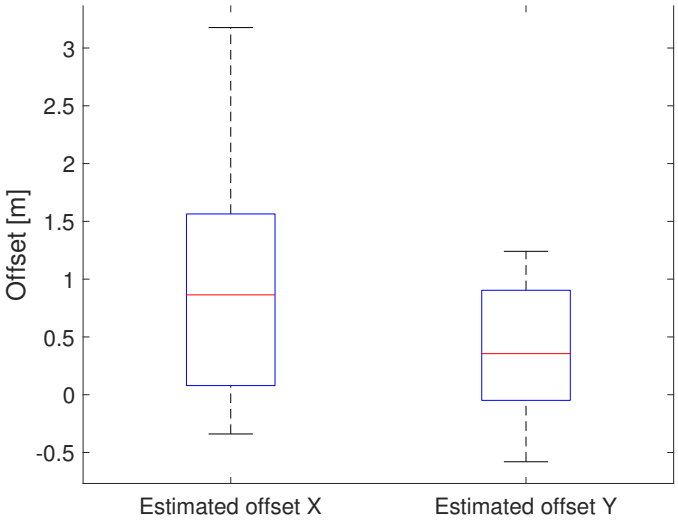

Fig. 11. Estimated X and Y variations associated to UAV position during landing.

make simultanesouly use of pitch and roll so that the UAV is able to move in diagonal lines. Finally, the algorithm can be further optimized so that the UAV is able to land under less favourable weather conditions.

ACKNOWLEDGMENTS

This work was partially supported by the "Ministerio de Ciencia, Innovación y Universidades, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2018", Spain, under Grant RTI2018-096384-B-I00.

COPYRIGHT INFORMATION

REFERENCES

[1] X. Pan, D. Q. Ma, L. L. Jin, and Z. S. Jiang, "Vision-based approach angle and height estimation for UAV landing," *Proceedings - 1st International Congress on Image and Signal Processing, CISP 2008*, vol. 3, pp. 801–805, 2008.

[2] D. Tang, F. Li, N. Shen, and S. Guo, "UAV attitude and position estimation for vision-based landing," *Proceedings of 2011 International Conference on Electronic and Mechanical Engineering and Information Technology, EMEIT 2011*, vol. 9, pp. 4446–4450, 2011.

[3] A. Gautam, P. B. Sujit, and S. Saripalli, "A survey of autonomous landing techniques for UAVs," in *2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings*, pp. 1210–1218, 2014.

[4] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming," *Pattern Recognition*, vol. 51, no. October, pp. 481–491, 2016.

[5] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image and Vision Computing*, vol. 76, no. June, pp. 38–47, 2018.

[6] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "ArUco: Augmented reality library based on OpenCV." https://sourceforge.net/projects/aruco/. Accessed: 2019-06-07.

[7] S. Jin, J. Zhang, L. Shen, and T. Li, "On-board vision autonomous landing techniques for quadrotor: A survey," *Chinese Control Conference, CCC*, vol. 2016-Augus, pp. 10284–10289, 2016.

[8] X. Chen, S. K. Phang, M. Shan, and B. M. Chen, "System integration of a vision-guided UAV for autonomous landing on moving platform," *IEEE International Conference on Control and Automation, ICCA*, vol. 2016-July, pp. 761–766, 2016.

[9] E. Nowak, K. Gupta, and H. Najjaran, "Development of a Plug-and-Play Infrared Landing System for Multirotor Unmanned Aerial Vehicles," *Proceedings - 2017 14th Conference on Computer and Robot Vision, CRV 2017*, vol. 2018-Janua, pp. 256–260, 2018.

[10] X. Chen, S. K. Phang, M. Shan, and B. M. Chen, "System integration of a vision-guided UAV for autonomous landing on moving platform," *IEEE International Conference on Control and Automation, ICCA*, vol. 2016-July, pp. 761–766, 2016.

[11] S. Lange, N. Snderhauf, and P. Protzel, "Autonomous landing for a multirotor uav using vision," in *In SIMPAR 2008 Intl. Conf. on Simulation, Modeling and Programming for Autonomous Robots*, pp. 482–491, 2008.

[12] O. Araar, N. Aouf, and I. Vitanov, "Vision Based Autonomous Landing of Multirotor UAV on Moving Platform," *Journal of Intelligent & Robotic Systems*, 2016.

[13] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "On the impact of inter-UAV communications interference in the 2.4 GHz band," *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*, pp. 945–950, 2017.

[14] L. Meier and QGroundControl, "MAVLink Micro Air Vehicle Communication Protocol." http://qgroundcontrol.org/mavlink/start. Accessed: 2019-01-30.

[15] F. Fabra, C. T. Calafate, J.-C. Cano, and P. Manzoni, "ArduSim: Accurate and real-time multicopter simulation," *Simulation Modelling Practice and Theory*, vol. 87, pp. 170–190, sep 2018.

[16] M. A. A. Careem, J. Gomez, D. Saha, and A. Dutta, "Hiper-v: A high precision radio frequency vehicle for aerial measurements," in *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–6, June 2019.