Vision based GPS-denied Object Tracking and Following for Unmanned Aerial Vehicles

Jesús Pestana
José Luis Sanchez-Lopez
Pascual Campoy

Computer Vision Group
Centro de Automática y Robótica, CSIC-UPM (Spain)
Email: jesus.pestana@upm.es, pascual.campoy@upm.es
Web: www.vision4uav.com

Srikanth Saripalli

ASTRIL Lab, SESE, ASU
Arizona State University (USA)
Email: srikanth.saripalli@asu.edu
Web: http://robotics.asu.edu/

***Abstract*—We present a vision based control strategy for tracking and following objects using an Unmanned Aerial Vehicle. We have developed an image based visual servoing method that uses only a forward looking camera for tracking and following objects from a multi-rotor UAV, without any dependence on GPS systems. Our proposed method tracks a user specified object continuously while maintaining a fixed distance from the object and also simultaneously keeping it in the center of the image plane. The algorithm is validated using a Parrot AR Drone 2.0 in outdoor conditions while tracking and following people, occlusions and also fast moving objects; showing the robustness of the proposed systems against perturbations and illumination changes. Our experiments show that the system is able to track a great variety of objects present in suburban areas, among others: people, windows, AC machines, cars and plants.**

***Keywords*—multirotor control, Visual Servoing, Object Following**

I. Introduction

The motivation of this work is to show that Visual Object Tracking can be a reliable source of information for Unmanned Air Vehicles (UAV) to perform visually guided tasks on GPS-denied unstructured outdoors environments. Navigating populated areas is more challenging to a flying robot than to a ground robot because it requires to stabilize itself at all moments; in addition to the other usual robotics operations. This provides a second objective to the presented work to show that Visual Servoing, or positioning a Vertical Take-Off and Landing (VTOL) UAV relative to an object at an approximate fixed distance, is possible for a great variety of objects. The capability of autonomous tracking and following of arbitrary objects is interesting by itself; because it can be directly applied to visual inspection among other civilian tasks.

The VTOL UAV platform used in the experimental work is the multirotor Parrot AR.Drone 2.0, shown in Fig. 1. Recent work has demonstrated that the the AR Drone is a realiable platform for VTOL UAV vision based navigation algorithm prototyping for GPS-denied environments, for example: autonomous navigation of hallways and stairs [1], visual SLAM based indoors navigation [2], reactive obstacle avoidance in natural environments [3] and floor optical flow based navigation [4].

978-1-4799-0880-6/13/$31.00 ©2013 IEEE

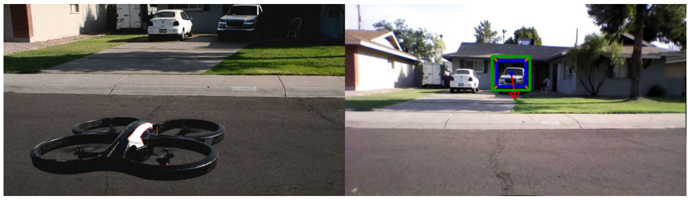

Fig. 1. (left) Image of the AR Drone 2.0 during one of the Visual Servoing experiments. (right) Modified front image of the drone to show the cotroller (green) references, (blue) feedback, and (red) control error.

The contributions of this paper are the integration of a system that: performs Visual Servoing on a great variety of targets, does not depend on GPS, and is able to achieve person following while handling occlusions. This work has been so far a feasibility project to showcase the possibilities of Visual Servoing to operate in relatively spacious environments in suburban areas. Its success has been achieved through the knowlegeable choice of robust and reliable components, namely: the open-source object tracker OpenTLD [5], [6], and the AR Drone 2.0 (see Fig. 1-left). This project will be improved in future work but the authors believe that the current results already have value to the scientific community. Several videos with descriptions about the presented work on Visual Servoing using the AR Drone 2, a cheap robotics platform, can be found on the following website: http://robotics.asu.edu/ardrone2_ibvs/.

II. Related Work

Recent research on Visual Tracking has been focused on online learning and detection [6], [7]. Such trackers have demonstrated to be able to track general targets online, which is the reason why they were selected to be tested in our project.

Research on Image Based Visual Servoing (IBVS) has shown that the performance of the robot depends on the set of used image features, which should be decoupled [8] or based on computing image moments on a group of points on the target [9]. Recent research has included non-overlaping multi-camera robotic systems [10]. More specific to our work the research [11] discusses "eye-in-hand" systems where the camera is fixed to a rigid body with actuated dynamics.

When compared to prior research, the main advantage of our system is that OpenTLD allows to perform visual servoing with a large number of different targets; which is a big improvement compared to targets marked with blobs of different sizes [12]; or to balloons [13], [14]. However, our architecture is not able to estimate the depth at which the target is located as in [15], or the relative attitude of the target with respect to the drone, as in [16].

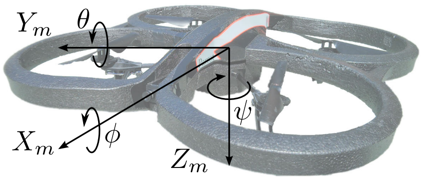

Fig. 2. Parrot AR.Drone 2.0 and its body reference frame, $\{X_m, Y_m, Z_m\}$.

III. System Description

Our system consists of several modules that can communicate with each other under the Robot Operating System (ROS) framework [17]. The AR.Drone 2.0 is commanded from a computer via WiFi link using the ardrone_autonomy ROS package [18] to communicate with the Drone, which is based on official AR-Drone SDK version 2.0. The following section is dedicated to describe the system and its components.

A. System Overview

The main modules of our system are an object tracker and an Image Based Visual Servoing (IBVS) controller. As shown on Fig. 3, the drone is commanded by the controller at 15-25 Hz calculating the drone reference commands based on the bounding box provided by the object tracker. The AR.Drone 2.0 is operated using the flight mode when the object is tracked properly and it will be hovering otherwise. The software sets the drone to hovering mode when the tracker losses tracking for 200 ms or more. The following is a brief description of each of the modules:

1) Object tracker: our software is currently using a C++ open source implementation of the OpenTLD tracker [19]. The OpenTLD tracker was originally developed by Z. Kalal at the University of Surrey during his PhD Thesis [5], [6]. All the open source repositories related to this library are located in [20], including a C++ ROS wrapper.
This tracker can robustly track objects on the drone's video stream. A great advantage of object trackers with learning capability is that they do not require any previous knowledge of the tracked object. It provides a bounding box (location, height and width) around the tracked object along with a confidence ratio. During our tests we have tested the tracker during drone flights on a great variety of objects that can be found in suburban areas, including: windows, AC machines, cars, plants, logos and people (t-shirt logos). The only constraints that were important in order to get high repeatability during tests were: tracker's learning is switched off to better handle object occlusion and during object following of small targets the bounding box should not include any background.

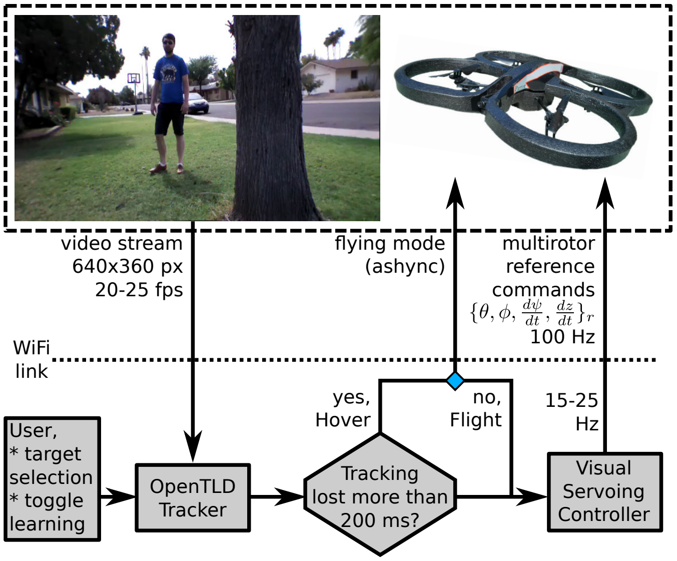

Fig. 3. System overview, the AR.Drone 2.0 is commanded from a computer over a WiFi link. The main components of the system are: the drone, the drone driver (ardrone_autonomy ROS package), the image rectification stage, the object tracker (OpenTLD tracker) and the controller. The user only interacts with the system to set the target to be visually tracked and to toggle the model learning of the tracker, in order to attain improved performance handling occlusions.

2) IBVS controller: the controller closes four feedback loops based on image features, which are the bounding box location and size, at 15-25 Hz. The references to the controller are desired location of the centroid of the bounding box in the image, and the size of bounding box. The resulting behaviour of the system is that the drone will turn to look to the target and approximately control its relative position with regards to the target.

As a result of the above mentioned system architecture, the sensor information required during the experiments is (see Fig. 4):

1) During succesful object tracking: the built-in operation of the drone requires, at all times, to use the IMU and the ultrasound altitude sensor. Additionally, our offboard sofware uses only the front camera image to control the vehicle. Note that the optical flow based speed estimation is not used, either by the IBVS controller or by the AR Drone 2.0 itself, during this operation mode.
2) Whenever the object tracking is lost or when the object is out of the image frame: the AR Drone 2.0 is automatically commanded to enter hovering mode. As a result the optical flow speed estimation, in addition to the previous sensors, is internally used to stabilize the vehicle.

The following subsections explain the architecture of the controller and the obtention of the controller gains.

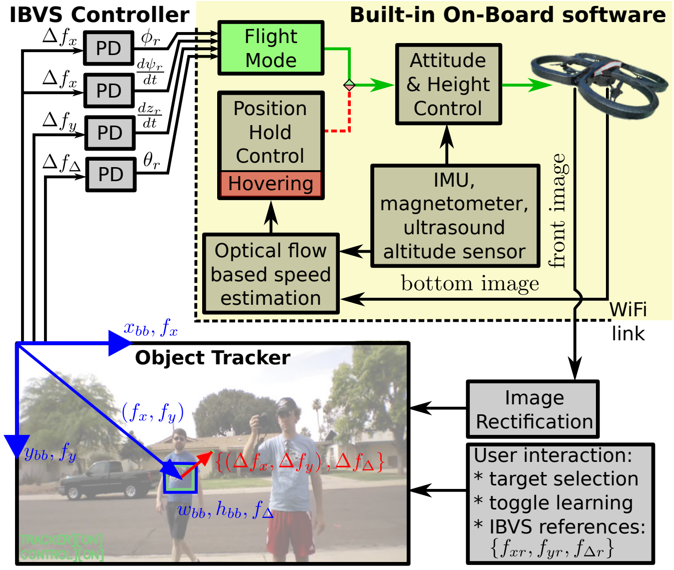

Fig. 4. Diagram of the Image Based Visual Servoing (IBVS) Controller. Since the tracker is working properly at this moment, the drone is operating under the flight mode and following the drone reference commands that the off-board computer is sending via WiFi. During flight mode, the optical flow speed estimation is unused. The figure shows the image features utilized by the controller, which are: $\{f_x, f_y \in [0, 1]\}$ related to the centroid position and $\{f_\Delta > 0\}$ to the size of the bounding box.

B. Image Based Visual Servoing (IBVS) Controller

An overview of the system focused on the description of the controller is shown in Fig. 4. The user interacts with the system only to: take off, land, start the controller, change the IBVS controller references, select the targets to be tracked and to toggle, on and off, the learning feature of the tracker. As shown, the feedback measurements and the references used by the controller are directly related to the target's bounding box on the image plane. Thus, the controller, as implemented during the current experimental work is purely a Visual Servoing controller. As depicted on Fig. 4, the drone reference commands calculated by the controller are only taken into account when the drone is operated on "Flight Mode".

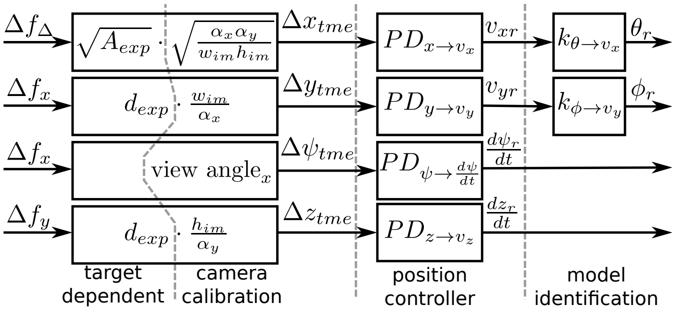

Fig. 5. IBVS Controller Breakdown, the four PD controllers shown in Fig. 4 can be broken down to show how they relate to a regular position controller.

The following variables, measured in pixels, specify the estimate of the target's bounding box as returned by the object tracker: horizontal x_{bb} and vertical y_{bb} location of its upper-left corner, and its width w_{bb} and height h_{bb}. Additionally the size constants of the image in pixels are: width $w_{im} = 640$ px, and height $h_{im} = 360$ px. The image features that are provided as feedback to the controller are calculated as follows, see Eq. 1 and Fig. 4, where x_{tm} is the frontal distance from drone to target:

$$\begin{aligned} f_x &= \frac{x_{bb} + (w_{bb}/2)}{w_{im}} \\ f_y &= \frac{y_{bb} + (h_{bb}/2)}{h_{im}} \\ f_\Delta &= \sqrt{\frac{w_{im} \cdot h_{im}}{w_{bb} \cdot h_{bb}}} \underset{\sim}{\propto} x_{tm} \end{aligned} \qquad (1)$$

Note that the image feature f_Δ is approximately proportional to x_{tm}, the frontal distance from drone to target, which results in simpler and better performance on Visual Servoing controllers [8], [21].

The behaviour of the controller is succesful, but the controller's behaviour is affected by the following factors:

- The controller is tuned to follow targets of surface size $A_{exp} = 40 \times 30$ cm, at an expected distance of $d_{exp} = 3$ m.
- Multirotor's degrees of freedom are not dynamically coupled, but in our system they are coupled by the controller architecture. The main couplings are the following:
- Pitch reference commands intended to control the distance to the target will cause the altitude controller to react unnecessarily.
- Yaw speed and roll reference commands cause similar movements on the horizontal centroid coordinate, f_x. Also, the dynamics of f_x are dominated by the yaw command. The effect on the system is that the PD controller that generates the roll commands is mainly stabilizing the platform laterally, but its action is not enough to counteract moderate winds.
- Due to the fact that the AR Drone 2.0's frontal camera is fixed to the vehicle's body, the yaw IBVS PD controller has to be strong enough to prevent the target of moving out of the image frame. A pan-tilt camera would allow to avoid this situation.

At its current stage this project is mainly a feasibility project, but the disadvantages of the current controller architecture will be adressed in the near future. The next subsection introduces a simple methodology to tune the IBVS controller gains.

C. Simple IBVS Controller tunning

As shown it is explained in this section, the IBVS Controller gains are related to those of a position controller through the relationship shown in Fig. 5, where each gain is shown to be the result of the multiplication of three terms: process of image formation on the camera, a previously tunned position controller; and the stationary relationship between speeds and drone reference commands.

The meaning of the variables and constants in the Fig. 5 are explained in the following enumeration:

- The target dependent parameters are the size of the tracked object's target surface, A_{exp}; and the expected distance to the target d_{exp}.

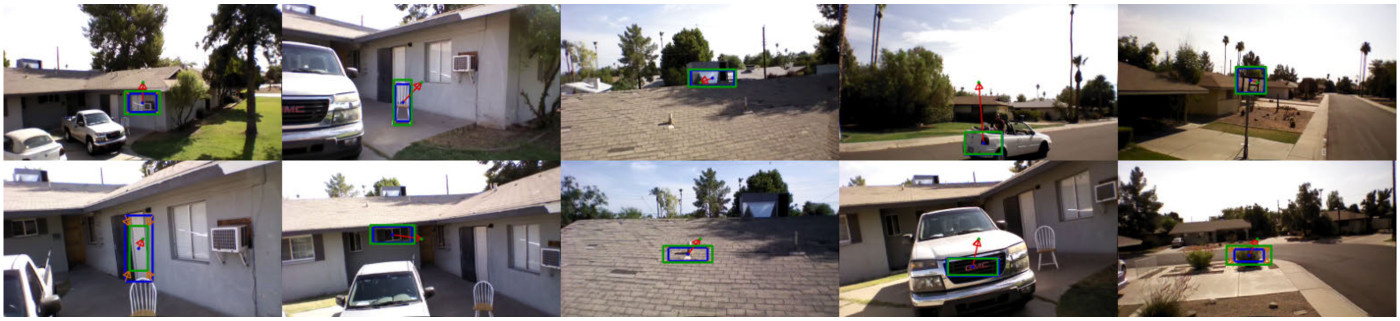

Fig. 6. Selection of on-board camera images showing targets upon which our system was tested: (house elements) a window with an AC machine, a chair, a door and a small window; (roof elements) AC machinery on the roof and a roof part; (car elements) a moving car and a car logo; (street elements) a basketball basket and a plant.

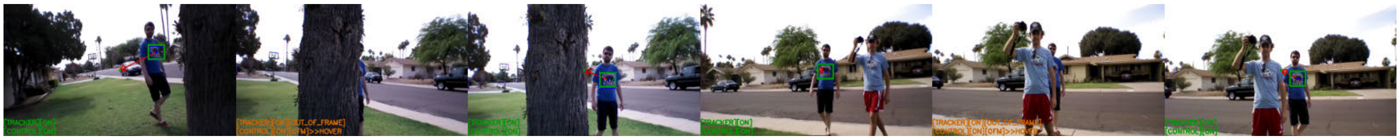

Fig. 7. Selection of on-board camera images, showing a person following test and how the system handles occlusions. The AR Drone 2.0 starts hovering until it locates the target again and proceeds with the Visual Servoing task. The first three images show target occlusion by a tree, and the second three images by another person. In our experience the learning feature of the tracker must be switched off in order to handle occlusions successfuly.

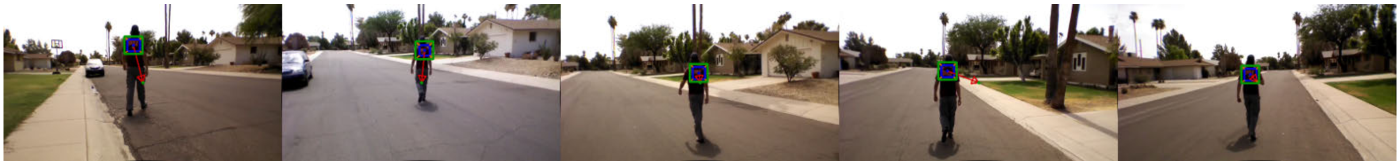

Fig. 8. Selection of on-board camera images from another person following test, which is explained in subsection IV-A. The drone follows a person along a street in a suburban area, the experiment lasted 1.5 minutes and the tracking had to be recovered by hand only once.

- The camera dependent parameters are: the image resolution along width w_{im} and height h_{im}; α_x, α_y and the horizontal view angle of the camera, "view angle$_x$", which are obtained from the rectified image projection matrix P:

$$P = \begin{bmatrix} \alpha_x & 0 & x_0 & 0 \\ 0 & \alpha_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2}$$

- $\{\Delta x_{tme}, \Delta y_{tme}, \Delta z_{tme}, \Delta\psi_{tme}\}$ is the estimated relative position of the target to the drone, expressed in the drone's reference frame, as shown in Fig. 2. $\{v_{xr}, v_{yr}\}$ are speed commands in the same reference frame.
- $PD_{var_1 \rightarrow var_2}$ are the PD controllers of an already tunned position controller which outputs speed commands. For this experimental work the controller presented in [4] was used.
- $\{k_{\theta\rightarrow v_x}, k_{\phi\rightarrow v_y}\}$, are the static relationship between the multirotor tilt and the attained horizontal steady-state speed. In multirotors this constant is related to the aerodynamic profile of the vehicle. For the AR Drone with the indoors hull, they were estimated to be $k_{tilt\rightarrow v_h} \approx 7\frac{\text{m/s}}{24^\circ}$ for angles lower than 8° [22].
- $\left\{\theta_r, \phi_r, \frac{d\psi_r}{dt}, \frac{dz}{dt}\right\}$ are the reference commands, Fig. 2.

IV. Experiments and Results

Several experimental flight videos are avaiable online on our website: http://robotics.asu.edu/ardrone2_ibvs/. The available videos show the system performing the following Visual Servoing tasks: two tests where the target matches the A_{exp} and d_{exp} parameters of the gain tunning; two tests where various urban objects were used as targets, one of them with selected targets moving along a street; a car and a person following tests along suburban area streets; and two tests were people were followed from close distances showing occlusion handling.

Various tests were performed to ascertain what kind of objects could be tracked visually. A selection of images that includes house, roof, car and street elements is shown in Fig. 6. The selected targets ranged from a quarter of the tunned size to more than ten times the tunned target surface, A_{exp}. The drone was able to visually track all these targets even when the objects were at a distance, relatively far from the stable visual tracking position.

A second battery of tests was performed to showcase moving object following, mainly including people following and some car following tests. For small moving targets, such as logos on people's t-shirts, the best performance was achieved when no background is included in the bounding box. However, for big moving targets, the bounding box can be chosen with background on it and the tracker and system will still work successfuly. The reason for this is that big targets tend to move slowly in the image plane, which accounts for the tracker's higher performance.

People following was highly succesful, which is the reason why it is the main focus on this article. As shown on Fig. 7, our solution can handle occlusion by objects such as trees and also by other people.

A. Quantitative performance during a person following task

The experimental test corresponding to the images shown in Fig. 8 was selected to showcase the performance of our Visual Servoing controller. The test lasted for about 1 minute and 30 seconds, where the drone followed a person in a suburban area. The main variables of the controller are plotted on Figs. 9 & 10.

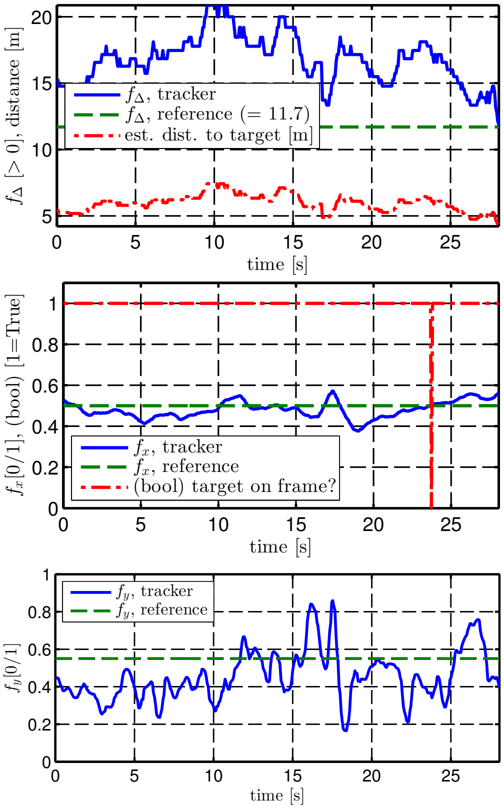

Fig. 9. Measured and reference values of the image features during a 30 seconds period of the experiment shown in Fig. 8. The $f_\triangle$ plot shows that the distance to the target varies slowly and it was estimated to be on the 5-7.5 m range, due to the target being 2.2 times smaller than expected; the real distance was about $sqrt(2.2)$ smaller, thus, in the 3.4-5 m range. Note that the object tracker does not estimate $f_\triangle$, the size of the target's bounding box, smoothly. f_x has been well controlled around the reference value; its graph also shows (red, dash-dot line) that the tracking was lost for a very small period of time once. The reason why there are big variations on f_y is that this image feature is tightly coupled to the vehicle's pitch, because the camera is fixed to the vehicle's body and the pitch commands are required to follow the moving target.

Figs. 9 & 10 show the main variables of the controlled system and show an overall succesful behaviour. There is no noise in the image features, the drone commands or the attitude and altitude of the drone. The coupling between pitch and altitude through the f_y image feature is noticeable.

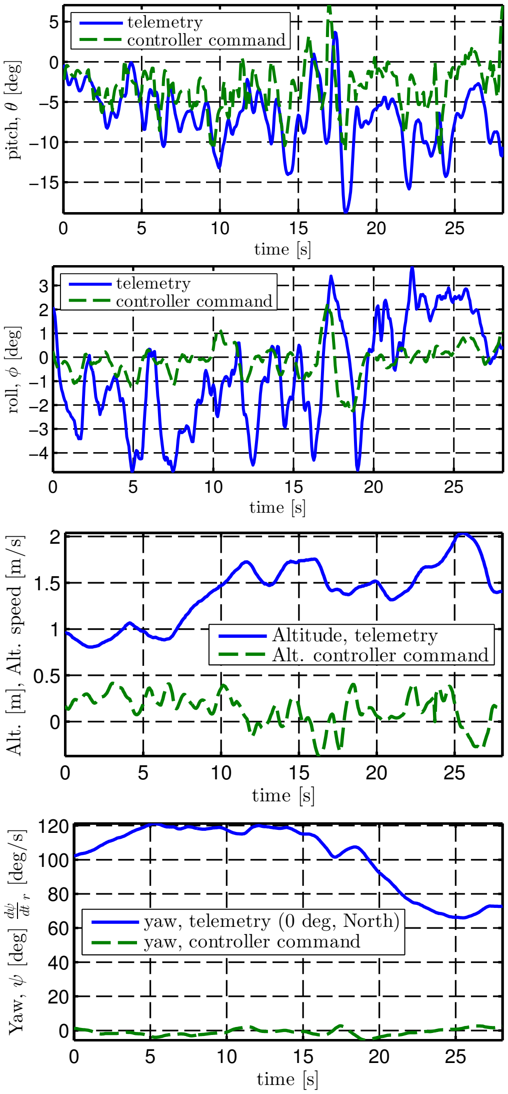

Fig. 10. Controller commands during a 30 seconds period of the experiment shown in Fig. 8. As shown, none of the commands have a noticeable level of noise. The pitch command has a negative average because the drone is following a person and it has to move forward. The roll command is approximately zero, as explained on the article the roll command is used just to stabilize the vehicle laterally. The drone stayed at an altitude of 1 to 2.0 m during the experiment, as measured by the ultrasound altitude sensor. The altitude speed command suffer from the influence of f_y on the pitch commands, because f_y is the image feature used to feedback the altitude controller. The yaw plot shows a constant heading while crossing the street and then it changes when the drone moves parallel to the street.

This flight, shown in Figs. 8 & 9 & 10, lasted one more minute. After that point the tracking was lost due to the tracker increasingly learning the background and finally loosing the target. After this event, the main target was reentered by hand again and the vehicle succesfully followed the target along the street for another 30 seconds. The total flight distance was about 90-110 m, with no user interaction other than target selection. These values allow us to estimate that the drone had to move an average forward velocity of 1.5 m/s.

As discussed on the papers and supported by the experimental videos, the system as a whole has demonstrated to be robust to temporary loss of the visual tracking. This fact is provided by the flying mode switching strategy and by the reliability of the AR Drone 2.0 hovering mode. The system has also been shown to be able to perform visual servoing on a great variety of targets, with robustness to low winds and no dependence on GPS.

V. Conclusions

In this paper, a visual based object tracking and following system is presented. Our flying robot is able to follow a variety of static and moving targets, without any dependence on GPS signals, using a recently developed visual tracking and target model learning algorithm. The system does not require the targets to be marked, and no prior knowledge about the targets is required. Our system has been able to perform Visual Servoing task on targets of varying size, from a quarter to more than ten times the tunned target size, and at varying distances from 1-2 m to 10-15 m of distance from the target. It has also achieved person following at up to 1.5 m/s of speed.

All the experiments have been performed in an unstructured suburban area, in an outdoors environment. Our system has been tested for person following tasks being able to handle occlusion by trees or other people. The computations are performed in an offboard computer that commands the vehicle from a WiFi link. Safety is assured even when the wireless connection is suddently degraded by using a multirotor platform that can attain on-board autonomous hovering using floor optical flow. The main contribution of the paper is to demonstrate that Visual Servoing on a great variety of targets including person following with occlusions handling, on an unstructured suburban area, and without dependence on GPS signals is feasible by a current low-cost but reliable UAV robotic platform.

Acknowledgments

The authors would like to thank the Consejo Superior de Investigaciones Cientificas (CSIC) of Spain for the JAE-Predoctoral scholarships of two of the authors, the UECIMUAVS Project (PIRSES-GA-2010) included in the Marie Curie Program and the Spanish Ministry of Science MICYT DPI2010-20751-C02-01 for project funding.

References

[1] C. Bills, J. Chen, and A. Saxena, "Autonomous MAV flight in indoor environments using single image perspective cues," 2011, pp. 5776–5783. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5980136

[2] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[3] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments," *CoRR*, vol. abs/1211.1690, 2012.

[4] J. Pestana, I. Mellado-Bataller, C. Fu, J. L. Sanchez-Lopez, I. F. Mondragon, and P. Campoy, "A general purpose configurable navigation controller for micro aerial multirotor vehicles," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, 2013, pp. 557–564.

[5] Z. Kalal, "Tracking Learning Detection," Ph.D. dissertation, University of Surrey, April 2011. [Online]. Available: http://www.ee.surrey.ac.uk/CVSSP/Publications/papers/Kalal-PhD_Thesis-2011.pdf

[6] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[7] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 983–990.

[8] O. Tahri, Y. Mezouar, F. Chaumette, and P. Corke, "Decoupled image-based visual servoing for cameras obeying the unified projection model," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 684–697, 2010.

[9] O. Tahri and F. Chaumette, "Point-based and region-based image moments for visual servoing of planar objects," *Robotics, IEEE Transactions on*, vol. 21, no. 6, pp. 1116–1127, 2005.

[10] A. Comport, R. Mahony, and F. Spindler, "A visual servoing model for generalised cameras: Case study of non-overlapping cameras," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 5683–5688.

[11] T. Hamel and R. Mahony, "Visual servoing of an under-actuated dynamic rigid-body system: an image-based approach," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 2, pp. 187–198, 2002.

[12] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck, "Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle," *Robotics, IEEE Transactions on*, vol. 25, no. 3, pp. 743–749, 2009.

[13] I. Mondragon, P. Campoy, M. Olivares-Mendez, and C. Martinez, "3d object following based on visual information for unmanned aerial vehicles," in *Robotics Symposium, 2011 IEEE IX Latin American and IEEE Colombian Conference on Automatic Control and Industry Applications (LARC)*, 2011, pp. 1–7.

[14] H. Zhang and J. Ostrowski, "Visual servoing with dynamics: control of an unmanned blimp," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, 1999, pp. 618–623 vol.1.

[15] R. Mahony, A. Brasch, P. Corke, and T. Hamel, "Adaptive depth estimation in image based visual servo control of dynamic systems." in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, 2005, pp. 5372–5378.

[16] C. Martínez, I. F. Mondragn, P. Campoy, J. L. Sánchez-López, and M. A. Olivares-Méndez, "A hierarchical tracking strategy for vision-based applications on-board uavs," *Journal of Intelligent & Robotic Systems*, pp. 1–23, 2013.

[17] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS : an open-source Robot Operating System," in *IEEE International Conference on Robotics and Automation (ICRA 2009)*, no. Figure 1, 2009.

[18] Mani Monajjemi, "ardrone_autonomy : A ros driver for ardrone 1.0 & 2.0," https://github.com/AutonomyLab/ardrone_autonomy, 2012, Autonomy Lab, Simon Fraser University.

[19] G. Nebehay, "Robust Object Tracking Based on Tracking-Learning-Detection," Master's thesis, Vienna University of Technology, Austria, 2012. [Online]. Available: http://gnebehay.github.io/OpenTLD/gnebehay_thesis_msc.pdf

[20] "Opentld object image tracker source repositories: (2011) Kalal, Z (PhD Thesis); matlab opentld implementation https://github.com/zk00006/OpenTLD , (2012) Nebehay, G (Msc. Thesis); C++ opentld implementation https://github.com/gnebehay/OpenTLD , (2013) Chauvin, R; C++ ROS opentld wrapper https://github.com/Ronan0912/ros_opentld."

[21] R. Mahony, P. Corke, and F. Chaumette, "Choice of image features for depth-axis control in image based visual servo control," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 1, 2002, pp. 390–395 vol.1.

[22] J. Pestana, "On-board control algorithms for Quadrotors and indoors navigation," Master's thesis, Universidad Politécnica de Madrid, Spain, 2012.