Computer Vision Based General Object Following for GPS-denied Multirotor Unmanned Vehicles

Jesús Pestana, Jose Luis Sanchez-Lopez, Srikanth Saripalli and Pascual Campoy

Fig. 1. (left) Image of the AR Drone 2.0 during one of the object tracking and following experiments. The drone is equiped with the outdoors hull to improve the system's wind disturbance rejection.
(right) Modified front image of the drone to show the controller (green) depth reference, (blue) feedback, and (orange) the altitude and lateral movement control error. The drone is controlled from an off-board computer through a WiFi link, and the target object is selected by the experimenter. The utilized tracker features online learning which causes problems if the target's tracked part includes background. This is the reason why, for person following tasks, a logo on the person's outfit is selected as the tracker's target.

I. Introduction

This research showcases visual object tracking for Unmanned Air Vehicles (UAV) to perform GPS-denied object following tasks, on a big variety of objects in outdoors suburban environments. Navigation is more challenging for a flying robot than for ground robots because it requires feedback to stabilize itself. This fact provides a second objective, which is to show that visual servoing or visual based object following is possible and reliable 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. So far, this work has been a feasibility project to showcase the possibilities of visual servoing to operate in relatively spacious unknown outdoors environments in suburban areas. The proposed architecture addresses these objectives through the knowlegeable choice of robust reliable components, namely: the open-source object tracker OpenTLD [1], [2], and the AR Drone 2.0 (see Fig. 1-left).

The Vertical Take-Off and Landing (VTOL) UAV platform used in the experimental work is the multirotor Parrot AR.Drone 2.0, shown in Figs. 1 & 2. Recent research has demonstrated that the the AR Drone is a realiable platform for VTOL UAV vision based navigation algorithm prototyping. For instance, the AR Drone has been used on the following research: autonomous navigation of hallways and stairs [3], visual SLAM based navigation [4] and reactive obstacle avoidance in natural environments [5].

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 [6] or based on computing image moments on a group of points on the target [7]. Recent research has included non-overlaping multi-camera robotic systems [8]. More specific to our work the research [9] 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 [10]; or to balloons [11], [12]. However, our architecture is not able to estimate the depth at which the target is located as in [13], or the relative attitude of the target with respect to the drone, as in [14].

This work is a continuation of previous work by the CVG group [11] where visual based GPS-dependent object following was achieved using a more expensive multirotor. In addition to performing GPS-denied visual servoing the

system presented in this paper can follow a larger variety of objects.

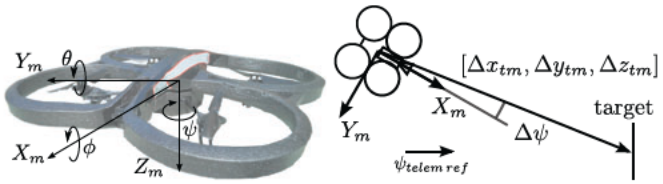

Fig. 2. (left) Parrot AR.Drone 2.0 and its body reference frame, $\{X_m, Y_m, Z_m\}$. X_m points towards the front of the vehicle, Y_m points towards the right and Z_m points downwards, obtaining an orthonormal right-handed reference frame. The attitude is defined using euler angles, which are denoted $\{\phi, \text{roll}\}$, $\{\theta, \text{pitch}\}$ and $\{\psi, \text{yaw}\}$.
(right) While performing the visual servoing tasks the drone tracks the target from a constant distance. The target's relative position, $[\Delta x_{tm}, \Delta y_{tm}, \Delta z_{tm}]$, is estimated from the image feedback using an expected target's size value, A_{exp}. $\psi_{telemref}$ is a variable internal yaw reference which specifies the prefered relative tracking direction.

II. System Description

Our system consists of several modules which communicate with each other under the Robot Operating System (ROS) framework [15]. The main modules of our system are an object tracker and an Image Based Visual Servoing IBVS controller. As shown in Fig. 3, the AR.Drone 2.0 is commanded from a computer via WiFi link using the ardrone_autonomy ROS package [16].

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 [17]. The OpenTLD tracker was originally developed by Z. Kalal at the University of Surrey during his PhD Thesis [1], [2]. The repositories related to this library can be found in [18].

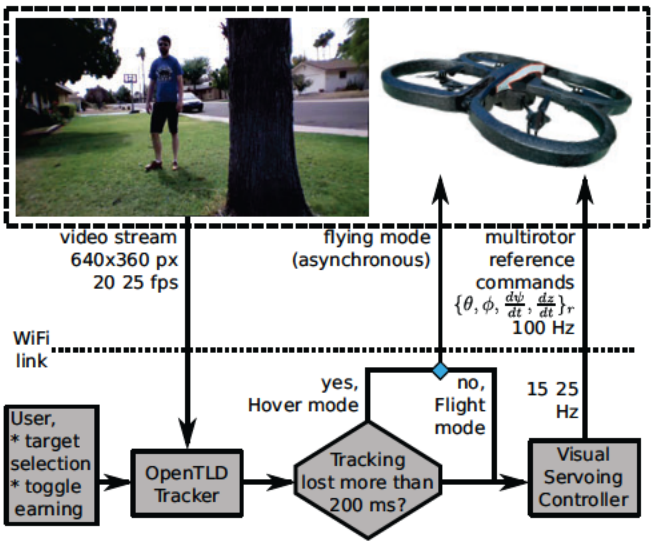

Fig. 3. System overview, the AR.Drone 2.0 is commanded from a computer over a WiFi link. The main components of the system are: the drone, the drone driver (ardrone_autonomy ROS package), the image rectification stage, the object tracker (OpenTLD tracker) and the controller.

OpenTLD can robustly track objects in the drone's video stream without any previous knowledge of the target. The tracker feeds back a bounding box (location, height and width) around the tracked object along with a confidence ratio. OpenTLD was tested during flight with a great variety of objects, see the objects shown in Figs. 6 & 7 & 8. The only constraints that were determined to be important to get high repeatibility during testing were that: the tracker's learning feature has to be switched off to enable the target occlusion capability, and small targets require the bounding box not to include any background.

2) IBVS controller: the controller closes four feedback loops based on three image features, which are the target's bounding box centroid and size, see Fig. 4. The references to the controller are its desired values. The resulting system's behaviour is that the drone will turn to look at the target and approximately control its relative position with regards to it.

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 off-board software uses the front camera image and part of the IMU telemetry data. Since the AR Drone is executing its internal "flying mode", it does not use optical flow based speed estimation during the execution of this operation mode of our architecture.
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 the internal "hovering mode". As a result, the AR Drone uses the on-board optical flow speed estimation to self-stabilize.

A. Image Based Visual Servoing (IBVS) Controller

An overview of the system focused on the description of the visual servoing controller is shown in Fig. 4. The tracker provides the horizontal, x_{bb}, and vertical, y_{bb}, location of the upper-left corner, and the width, w_{bb}, and height, h_{bb}, of the target on the image plane. The image features that are provided as feedback to the controller are calculated as follows, see Eq. 1 and Figs. 2 & 4:

$$\begin{aligned} f_u &= \frac{x_{bb} + (w_{bb}/2)}{w_{im}} \\ f_v &= \frac{y_{bb} + (h_{bb}/2)}{h_{im}} \\ f_\Delta &= \sqrt{\frac{w_{im} \cdot h_{im}}{w_{bb} \cdot h_{bb}}} \propto x_{tm} \end{aligned} \tag{1}$$

Note that the image feature f_Δ is approximately proportional to x_{tm}, the frontal distance from drone to target, which results in better visual servoing performance [6].

The utilization of a fixed camera on a visual servoing multi-rotor system naturally couples the controled variables

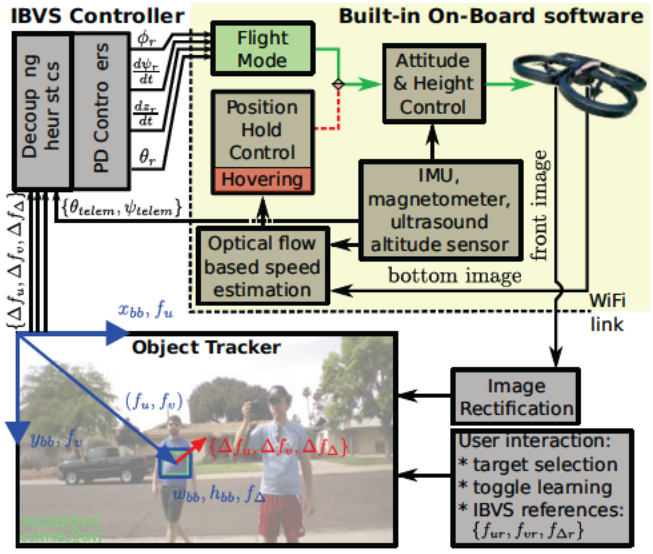

Fig. 4. Diagram of the Image Based Visual Servoing (IBVS) Controller. Since the tracker is working properly at this moment, the drone is operating under the flight mode and following the drone reference commands that the off-board computer is sending via WiFi. During flight mode, the optical flow speed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

through the target's movement on the image plane, which has required the design of decoupling control heuristics for the altitude and lateral movement degrees of freedom. Thus, the correction shown on Eq. 2 on the tracking error is calculated, which greatly decreases the mentioned DOF couplings.

$$\begin{aligned} \Delta f_{u\,\psi} &= \Delta f_u \\ \Delta f_{u\,y} &= \Delta f_u - \frac{\psi_{telem\,ref} - \psi_{telem}}{FOV_u} \\ \Delta f_{v\,z} &= \Delta f_v - \frac{\theta_{centroid\,ref} - \theta_{telem}}{FOV_v} \\ \Delta f_{\Delta\,x} &= \Delta f_\Delta \end{aligned} \tag{2}$$

θ_{telem} and ψ_{telem} are the pitch and yaw angles of the vehicle respectively, as shown in Fig. 2, which are obtained from the AR Drone telemetry data. FOV_u and FOV_v denote the horizontal and vertical field of view of the camera. $\psi_{telem\,ref}$ is a yaw heading reference which is updated to ψ_{telem} everytime that the drone is commanded to enter the Flight Mode or whenever $|\psi_{telem\,ref} - \psi_{telem}| > 25°$. $\theta_{centroid\,ref}$ is always 0.0, and θ_{telem} is obtained from the drone's telemetry data.

B. IBVS PD Controllers Parameter Breakdown

In this subsection the controller gains are decomposed, see Fig. 5, in parameters related to various system components: process of image formation on the camera, a previously tunned position controller; and the stationary relationship between speeds and drone reference commands. This decomposition allows to understand the effect that each parameter can have on the overall behaviour of the system during the Visual Servoing task.

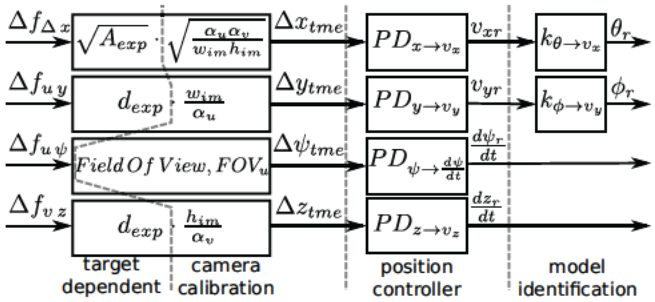

Fig. 5. IBVS Controller Breakdown, the PD controllers shown in Fig. 4 can be decomposed to show how they relate to a regular position controller. The shown parameters come from: the camera parameters, the expected target size A_{exp} and distance to the target d_{exp}, the position controller gains and the aerodynamic friction of the vehicle. The obtained PD gains are: $\{PD_{f_u \to y_{tm}},\ Kp = -0.298,\ Kd = -0.145\}$, $\{PD_{f_u \to \Delta\psi},\ Kp = -0.990,\ Kd = -0.119\}$, $\{PD_{f_v \to z_{tm}},\ Kp = 1.430,\ Kd = 0.371\}$, $\{PD_{f_{\Delta x} \to x_{tm}},\ Kp = 0.0254,\ Kd = 0.0124\}$, for the following AR Drone command gains configuration: $\{\theta_r = 1 \to 12°\}$, $\{\phi_r = 1 \to 12°\}$, $\{\frac{d\psi_r}{dt} = 1 \to 100\frac{\text{deg}}{\text{s}}\}$, $\{\frac{dz}{dt} = 1 \to 1\frac{\text{m}}{\text{s}}\}$.

The meaning of the the constants in Fig. 5 is:

- The target dependent parameters are the size of the tracked object's target surface, $A_{exp} = 40{\times}30$ cm; and the expected distance to the target $d_{exp} = 3\ m$.
- The camera dependent parameters are: the image resolution along width w_{im} and height h_{im}; α_u, α_v and the horizontal field of view of the camera, FOV_u, obtained from the rectified image projection matrix P, see Eq. 3:

$$P = \begin{bmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3}$$

- $\{\Delta x_{tme}, \Delta y_{tme}, \Delta z_{tme}, \Delta\psi_{tme}\}$ is the estimated relative position of the target with respect to the drone, expressed in the drone's reference frame, as shown in Fig. 2. $\{\Delta v_{xmr}, \Delta v_{ymr}\}$ are speed commands in the same reference frame.
- $PD_{var_1 \to var_2}$ are the PD controllers of an already tunned position controller which outputs speed commands. For the experimental work the position controller presented in [19] was used.
- $\left\{k_{\theta \to v_x}, k_{\phi \to v_y}\right\}$, 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. Using the methodology presented in [20] for the AR Drone with the outdoors hull, they were estimated to be $k_{tilt \to v_{xy}} \approx 5 - 7\frac{\text{m/s}}{24°}$ for angles lower than 12°.
- $\left\{\theta_r, \phi_r, \frac{d\psi_r}{dt}, \frac{dz}{dt}\right\}$ are the reference commands, Fig. 4.

III. Experiments and Results

The main focus of the experimental work was to demonstrate that our system performs the object following task, as described in Sec. II, successfuly. Note that the presented architecture does not rely on GPS at any level. A series of real-world experiments were conducted on suburban areas, of

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. The red arrow shows the target centroid tracking error on the image plane.

Fig. 7. Selection of on-board camera images, showing a person following test and how the system handles target occlusion. When target occlusion occurs the AR Drone 2.0 starts hovering until the target is detected again, and then it proceeds with the visual servoing task. The first three images show target occlusion by a tree, and the second three images show occlusion by another person. The learning feature of the tracker is used at the beginning of every experiment and then it is switched off. Otherwise the tracker may not be able to recognize the occlusion event appropriately.

Fig. 8. Selection of on-board camera images from another person following test, which is explained in subsection III-A. The robot tracks and follows a person along a street corridor in a suburban area. The experiment lasted 45 seconds where the AR Drone 2.0 covered a distance of about 120-140 m, thus navigating at an average speed of 2.65-3.10 m/s. The orange arrow shows the decoupled altitude and lateral movement control actions.

which experimental flight videos can be watched online at the ASTRIL lab website: http://robotics.asu.edu/ardrone2_ibvs/. The available videos show: two tests where the target matches the A_{exp} and d_{exp} parameters used to calculate the controller gains (Fig. 5); testing against various objects present on suburban areas; a car and a person following tests along suburban area streets; two tests where people were followed from close distances showing occlusion handling; the videos in the section 3.5 of the website show people following tests where the outdoors hull and the decoupling heuristics were utilized including the test presented in Sec. III-A.

Various tests were performed to ascertain what kind of objects could be tracked visually. A selection of images acquired during test flights 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 including some background and the tracker and system will still work successfuly. The reason for this is that big targets tend to evolve slowly in the image plane, which accounts for the tracker's better performance.

People following was highly succesful. As shown in Fig. 7, our solution can handle occlusion by objects such as trees and also by other people. In this kind of situation, the system will automatically switch to hovering mode until the target is detected again; and then it will proceed with the visual servoing task. On these experiments, the learning feature of the tracker was switched off after a 10-30 seconds period of learning. Occlusion handling was highly degraded if the tracker still had the learning feature enabled.

The flight time, when including target loss, second detection, etc; can reach battery depletion provided that the outdoors environment is spacious.

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. First, the tracker was trained to learn the target. Then, the learning feature of the tracker was switched off before the experimenter started running.

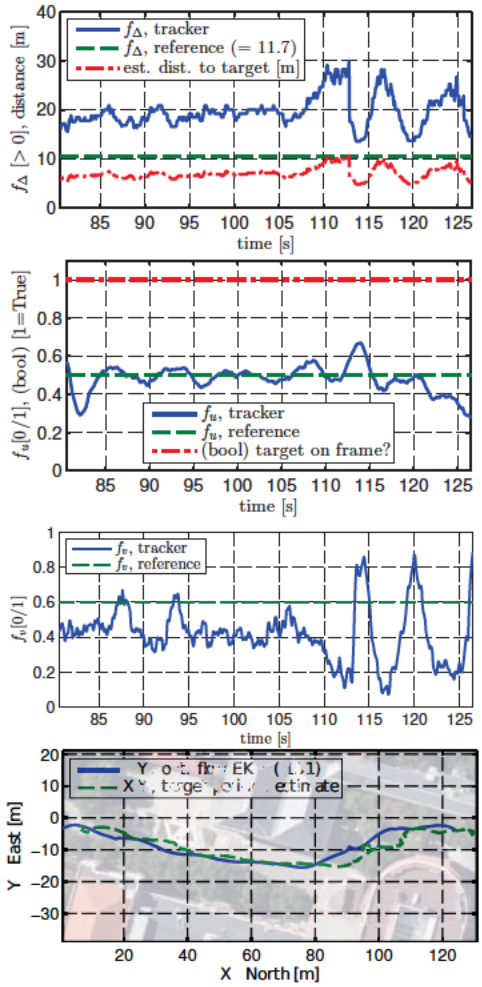

Fig. 9. Measured and reference values of the image features during the experiment shown in Fig. 8. The f_Δ plot shows that the distance to the target varies slowly and it was estimated to be on the 5-10.5 m range. Due to the target being 2.2 times smaller than expected; the real distance was about $sqrt(2.2)$ smaller, thus, in the 3.5-7 m range. We note that the object tracker does not estimate f_Δ, the size of the target's bounding box, smoothly all the time; the target bounding box sometimes jumps to another estimate on the image plane. f_u has been well controlled around the reference value; it's graph also shows (red, dash-dot line) that the tracking was not lost during this test. There are big variations in f_v near the end of test which might be due to this feature being coupled to the vehicle's pitch. The navigation speed of the vehicle or an estimation error on the vehicle's pitch might be the cause of the decoupling's heuristic degradation on that part of the test. The last plot shows an estimate of the vehicle's trajectory based on the vehicle's telemetry using the EKF presented on [20]. The target's position estimate is obtained adding the estimate $\{\Delta x_{tme}, \Delta y_{tme}, \Delta z_{tme}\}$, shown in Fig. 5, to the EKF's drone's position estimate.

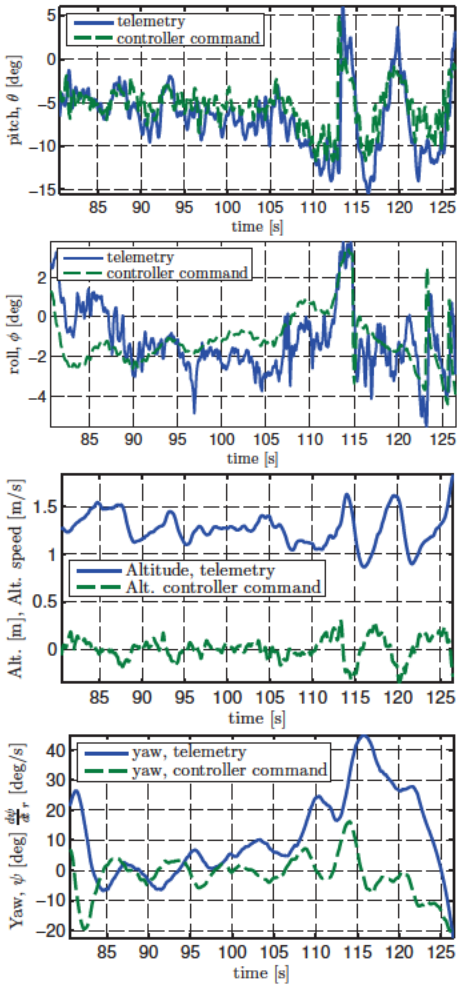

Fig. 10. Controller commands during the experiment shown in Fig. 8. As shown, none of the commands present a high level of noise. The pitch command has a negative average because the drone is following a person who is running and it has to move forward to keep up. The roll command is low and only used during lateral movement or wind disturbance rejection to stabilize the vehicle's lateral movement. The drone stayed at an altitude of 1 to 1.5 m during most of the experiment, as measured by the ultrasound altitude sensor. Near the end of the test, the altitude speed command is coupled with the pitch command, which is caused by the coupling of the f_v image feature with the pitch command. The yaw plot shows a constant heading during the first half of the experiment and then it shows the turn on the last part of the street corridor covered during the experiment, see the last plot of Fig. 9. As explained before, the yaw heading is controlled directly to keep the target in the middle of the image which is important to avoid tracking loss events. The effect of the internal yaw reference $\psi_{telem\,ref}$ is that the vehicle tends to look at the target from the same direction.

The run lasted 45 seconds where the AR Drone 2.0 covered a distance of about 120-140 m. Thus, the drone navigated at an average speed of 2.65-3.10 m/s. Thus far, these values show the peak performance of our IBVS controller.

The Figs. 9 & 10 show the main variables of the controlled system during the experiment. There is low noise in the controlled image features, the drone commands and the attitude and altitude of the drone. In spite of the decoupling heuristics, the coupling between pitch and altitude through the f_v image feature near the end of the test is still noticeable. The reason why there are big variations in f_v 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. f_v variations are then introduced in the altitude speed command through the calculations of the PD altitude controller. Another improvement that was introduced lately in our controller was the utilization of the outdoors hull and the yaw-lateral movement decoupling heuristic. Again, it is difficult to quantify the wind disturbance rejection improvement in any other way than investigating the external

videos of the tests, which in our opinion show a clear improvement.

As discussed in the paper 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 OpenTLD algorithm, to the extent of our limited group of experiments, has shown to be very reliable for target tracking and detection and it has only rarely detected a wrong object.

IV. Future Work

There are two main research lines to improve the performance of the system. The first line is to use another tracking algorithm like [14] which provides the projective transformation in the image plane; instead of OpenTLD which provides the position and size of the object in the image. The second line is to improve the reliability of the architecture by implementing an active target recovery scheme. For instance a target and drone 3D position estimation could be used to feedback a position controller in order to recover the target.

V. Conclusions

In this paper, a visual based object tracking and following architecture for multirotor vehicles is presented. The experimental work was performed using an AR Drone 2.0, and the algorithms where run on an off-board laptop computer via a WiFi link. Our system is able to follow and stabilize itself and it is able to track a large variety of different objects. Additionally, safety is assured even when the wireless connection is suddenly degraded, the tracking is lost or the target is occluded; by using a multirotor platform that can attain on-board autonomous hovering using floor optical flow based odometry. Our system has been able to perform visual servoing with targets of varying size, from a quarter to more than ten times the tunned target size, at varying distances from 1-2 m to 10-15 m of distance from the target, and it has has achieved person following at speeds up to 2.5-3.0 m/s for a period of 45 seconds.

The contributions of this paper are two-fold. First, it has been demonstrated that current tracking algorithms, such as OpenTLD, can reliably work on a fixed camera multirotor vehicle to feedback an Image Based Visual Servoing (IBVS) controller, even during high velocity autonomous navigation. Second, our architecture has been able to follow a large variety of unmarked targets of different sizes and from a wide range of distances. Moreover, the algorithm is validated using a low-cost platform, the Parrot AR Drone 2.0, in outdoor conditions while tracking and following people. The system has succesfuly and repeatedly handled occlusion events and tracked fast moving targets, such as a person running; showing the robustness of our system against wind disturbances and illumination changes.

References

[1] 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

[2] 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.

[3] 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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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

[18] "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."

[19] J. Pestana, I. Mellado-Bataller, J. L. Sanchez-Lopez, C. Fu, I. F. Mondragón, and P. Campoy, "A general purpose configurable controller for indoors and outdoors gps-denied navigation for multirotor unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 387–400, 2014.

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