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 knowledgeable 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 AR Drone is a reliable 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-overlapping 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.

![Diagram](image)

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 \( \{\phi, \text{roll}\}, \{\beta, \text{pitch}\} \) 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, \( A_{exp} \). \( \psi_{rel \text{e}m ref} \) is a variable internal yaw reference which specifies the preferred 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].

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 repeatability 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 successful 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:

\[
\begin{align*}
  f_u &= \frac{x_{bb} + \left(\frac{w_{bb}}{2}\right)}{w_{im}} \\
  f_v &= \frac{y_{bb} + \left(\frac{h_{bb}}{2}\right)}{h_{im}} \\
  f_A &= \sqrt{\frac{w_{im} \cdot h_{im}}{w_{bb} \cdot h_{bb}}} \\ &= \infty \quad x_{tm}
\end{align*}
\]

Note that the image feature \( f_A \) 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 centered variables
The meaning of the the constants in Fig. 5 is:
- The target dependent parameters are the size of the tracked object’s target surface, \( A_{\text{exp}} = 40 \times 30 \) cm; and the expected distance to the target \( d_{\text{exp}} = 3 \) m.
- The camera dependent parameters are: the image resolution along width \( w_{\text{im}} \) and height \( h_{\text{im}} \); \( \alpha_{\text{im}}, \gamma_{\text{im}} \) and the horizontal field of view of the camera, \( FOV_{u} \), obtained from the rectified image projection matrix \( P \), see Eq. 3:

\[
P = \begin{bmatrix}
\alpha_{u} & 0 & u_{0} & 0 \\
0 & \alpha_{v} & v_{0} & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]  

(3)

- \{\Delta x_{\text{time}}, \Delta y_{\text{time}}, \Delta z_{\text{time}}, \Delta \psi_{\text{time}}\} 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 x_{\text{arm}}, \Delta y_{\text{arm}}\} are speed commands in the same reference frame.
- \(PD_{\text{var1}} \rightarrow \text{var2}\) are the PD controllers of an already tuned position controller which outputs speed commands. For the experimental work the position controller presented in [19] was used.
- \{\kappa_{\phi_{\rightarrow v_{x}}}, \kappa_{\phi_{\rightarrow v_{y}}}\}, are the static relationship between the multicopter tilt and the attained horizontal steady-state speed. In multicopters 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 \( k_{\text{tilt}} \approx 5 - 7 \) m/s for angles lower than 12°.
- \{\theta_{r}, \phi_{r}, \psi_{r}, \kappa_{r}, \kappa_{y}, \kappa_{x}, \kappa_{z}, \kappa_{dy}, \kappa_{dz}, \kappa_{dx}\} 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. successfully. 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
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 tuned size to more than ten times the tuned 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 successfully. 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 successful. 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.
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 sucessfully 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