Aalborg Universitet

AALBORG UNIVERSITY
DENMARK

UAV Visual Servoing Navigation in Sparsely Populated Environments

Durdevic, Petar; Ortiz Arroyo, Daniel; Li, Shaobao; Yang, Zhenyu

Published in:
IFAC-PapersOnLine

Publication date:
2020

Document Version
Accepted author manuscript, peer reviewed version

Link to publication from Aalborg University

Citation for published version (APA):
Durdevic, P., Ortiz Arroyo, D., Li, S., & Yang, Z. (2020). UAV Visual Servoing Navigation in Sparsely Populated Environments. Manuscript submitted for publication.

UAV Visual Servoing Navigation in Sparsely Populated Environments

Petar Durdevic[1], Daniel Ortiz-Arroyo, Shaobao Li, and Zhenyu Yang

Aalborg University, Esbjerg DK-6700, Denmark,
pdl@et.aau.dk,
WWW home page: https://vbn.aau.dk/en/persons/125324

Abstract. This paper presents a novel approach based on deep neural networks for autonomous navigation of a quad-copter UAV in a sparsely populated environment. Images from the video camera mounted on the UAV are split, emulating a compounded eye, and processed by deep neural networks that calculate the probability that each image contains a wind turbine object. Then, these probabilities are used as inputs to a vision servoing system that controls the drone's navigation movements. Our experiments show that our approach produces relatively stable movements in the UAV, allowing it to find and navigate autonomously towards a wind turbine.

Keywords: Quad-copter UAV, Control, Visual Servoing, Compounded Eye, Artificial Intelligence, Vision, Monocular Camera

1 Introduction

The increasing demand of energy and the threat of climate change is continuously expanding the production of energy with the use of green technologies. One of the most effective green technologies for the production of electricity is wind turbines.

Wind turbine technologies delivers energy that is cleaner than fossil fuels, but they are not completely CO_2 neutral given that their end-of-service life span must be considered in the whole life-cycle analysis of energy production. One key factor in keeping the wind turbines operational for a longer time is proportioning them a proper maintenance. This requires continuous monitoring of the wind turbine's state. Currently, this task is performed using a diversity of methods such as: drone-based inspections, ground-based camera inspections and manual inspections, each with a market share of 7%, 10% and 83%, respectively Jamieson and Hassan (2011). Eventually manual inspections may no longer be able to keep up with the increase in demand, and thus automating this process to a high degree is beneficial.

Drones and other unmanned autonomous vehicles (UAVs) have played an increasingly important role in automating inspections, due to their flexibility and low cost. This trend started as early as 1980s as reported in Lattanzi and Miller (2017). Currently, drones are used for a variety of inspection tasks. For instance, to inspect river channels for pollution (Rusnák et al (2018); Flener et al (2013)), detecting fractures in bridges and buildings (Rakha and Gorodetsky (2018)), detecting faults in high voltage power transmission lines (Day (2017)), finding defects and leakages in pipelines or building facade defects (Lattanzi and Miller (2017); Shakmak and Al-Habaibeh (2015); Morgenthal and Hallermann (2014)), and to inspect the state of the blades on on-shore and off-shore wind turbines (Wang and Zhang (2017); Stokkeland et al (2015); Nikolov and Madsen (2017); Moolan-Feroze et al (2019)), among other applications. Many of these inspection tasks are performed by an operator who manually controls the movements of a drone equipped with a camera or other sensory equipment. However, performing autonomous inspections with an UAV, still remains a challenge. To accomplish this task, one of the main features needed in a drone is the use of computer vision techniques. Latest advancements in Deep Neural Networks (DNNs) now allow drones to perform a variety of common computer vision tasks such as recognizing objects from captured images, performing localization, image segmentation and motion tracking.

One application of image-based servoing is in wind turbine inspection. In this application, an autonomous drone equipped with a high resolution camera may be used to reducing the burden on the operator for continuously keeping track of drone's position and trajectory. Instead, the operator may focus on capturing the images required for the inspection. The ultimate goal is to reduce overall inspection time and cost by fully automating the inspection process with a drone.

The DNNs used in computer vision employ a stack of convolutional networks, pooling layers, normalization and fully interconnected layers. Each of these layers is designed to recognize from simple features such as lines and edges to contours and other more complex forms, comprising these features.

The output of a computer vision system may be used as one of the multiple inputs that UAVs use to navigate autonomously. In visual servoing, the position and orientation of a UAV is controlled using one or more visual features. These features could be points, lines or other geometric shapes (Kanellakis and Nikolakopoulos (2017)). Visual servoing algorithms may be classified into position-based visual servoing, image-based servoing or a hybrid of the two techniques, Kanellakis and Nikolakopoulos (2017); Espiau et al (1992). Position-based servoing requires a 3D model to be built of the target, so that its position could be estimated from the image features. In image-based servoing the input to the control system is computed from a 2D image space.

Nicolas Guenard (2008); Mondragón et al (2010); Ho and Chu (2013); Sa et al (2014); Metni and Hamel (2007) used traditional pattern recognition techniques in their vision systems such as feature detection algorithms for edges and blobs. However, image based visual servoing in UAVs may be also implemented with supervised machine learning techniques based on shallow or deep neural networks as reported in Pomerleau (1989, 1991); Bojarski et al (2016); Giusti et al (2016); Smolyanskiy et al (2017); Loquercio et al (2018); Drews et al (2018);

Goldfain et al (2019); Drews et al (2019); Durdevic et al (2019).

Similar to the aforementioned approaches, we used image-based visual servoing to navigate towards our target object. However, we do not use trails or blobs to find our target. As Giusti et al (2016); Smolyanskiy et al (2017); Drews et al (2019); Loquercio et al (2018) do, we use DNNs but our DNNs are pre-trained to recognize not trails, but more complex objects such as wind turbines. The challenge we face is the absence of clearly defined paths, as we wish to deploy the UAV in an offshore environment where no clear landmarks may be found. Instead we identify and localize the object within an image that we wish to navigate towards and use this information for navigation.

Our system segments images, captured by a RGB camera mounted on the UAV, into what we call the compounded eye, currently consisting of four segments. The DNNs analyze, in parallel, the four segments in which an image has been segmented, to determine which segment has the highest probability for containing a wind turbine. This data is constantly fed into the controller that computes the yaw angle required to navigate towards the wind turbine, the forward velocity of the vehicle and when to stop the vehicle to avoid colliding with the wind turbine.

This paper is a continuation of our previous work in Durdevic et al (2019), where the drone scanned the environment searching for a wind turbine using a convolutional DNN and stopped if the wind turbine was detected in an image. This was a successful experiment performed in a laboratory environment.

The contribution of this work is a visual navigation system that gives a UAV the ability to navigate autonomously towards an object, in an environment with no visible tracks nor landmarks, other than the object itself. To our knowledge, no other research work reported in the literature uses this approach for image based visual servoing. Our work is a step in the direction of implementing a fully autonomous system for wind turbine inspection.

2 Perception-Based Navigation

The perception based navigation is a fusion of a DNN image recognition and classical feedback control for navigation of a UAV, with the aim of localizing an object in a sparse environment. Additionally to this, the output from the DNN serves as a reference to the UAV's outer loop feedback controller and to control UAV's yaw and forward movement. The control system is described in the following.

2.1 Image Capturing and Processing (Compounded Eye)

An on-board camera R200 from Intel is used to capture images. This camera is capable of capturing images with a resolution of 1080x1920x3 pixels at 30 fps, where the image size is scaled to 454x454x3 pixels to create the four segments of our compounded eye. Figure 1 shows an image collected during the experiment presented in this work. Using this image, the DNN calculated the probability the upper left segment contained a wind turbine, triggering the **Object Found** event.

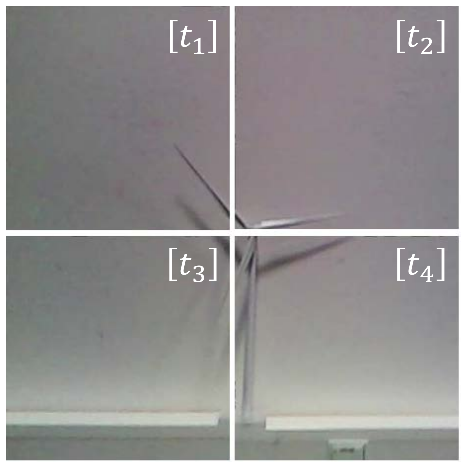

Fig. 1. Segmentation of the image into 4 segments, each has a resolution of 227x227x3 pixels. The recognition of this image, produced the **Object Found** event in the experiments presented in this paper.

Images form the on-board camera in the UAV are processed and segmented, creating a vector that represents the compounded eye as $[I_i, t_i]$, where I_i are the concatenated image pixels with size (227*227*3) and t_i is the *tag* of the image. The resulting vector size is 227*227*3+1=154588. The tags, $t = [t_1, t_2, t_3, t_4]$ represent the four segments: $\left[\begin{array}{c|c} t_1 & t_2 \\ \hline t_3 & t_4 \end{array}\right]$. This vector is then sent to the Ground Control Station (GCS), where it is unpacked and converted into images, with sizes (227x227x3 pixels) and their corresponding tags. The amount of segments is arbitrary and only limited by the camera's resolution. Therefore, in the general case, tags can be represented by $t = [t_1, t_2, ..., t_n]$. The unpacked images are then processed by a pretrained DNN.

2.2 Image Detection Using DNN

In this work a pre-trained DNN namely AlexNet was used (Krizhevsky et al (2012)). The DNN was retrained using transfer leaning to recognize wind turbines.

This same network was used in our previous work, Durdevic et al (2019), where the last layer consisted of two outputs [*windturbine*, *environment*]. In this work it was extended to include four outputs representing the following classes, [*curtain*, *net*, *wall*, *windturbine*]. These classes were chosen as these are the views that the UAV most often will face in our lab environment, where the class [*windturbine*] is used for visual servoing. The network was retrained using a total of 3036 images collected in our lab using a still camera (Canon 5ds). 70% of the images were used for training and 30% for validation. Images were preprocessed in memory by applying synthetic transformations, such as reflexions and translations with the goal of increasing the number of training images and to avoid over-fitting. A low learning rate of 0.0001 was used to train the first layers. For the last layers, a fastest weight learning rate of 20 was used so that they could quickly learn the main new features that characterize a wind turbine.

2.3 Visual Servoing

The perception based Visual Servoing controller is based on the hypothesis that when the wind turbine appears in one of the segments, the probability computed by the DNN will be largest in the corresponding segment. This probability can thus be used to adjust the orientation of the UAV, such that it has the desired orientation, i.e. the center most segments to have the largest probability of seeing the object.

If we simply this problem to two segments in the horizontal plane, we will get a left and right segments, whose center point is located in the Cartesian coordinates, denoted as $[S_l = [x_l, y_l], S_r = [x_r, y_r]]$. The image in each of these segments, has a probability of containing the object in question, represented by the two probabilities $[P(S_l), P(S_r)]$.

The process to compute these probabilities is not straight forward, as each segment consists of a matrix of pixels, each representing an integer value with eight bits. One way of computing this is using DNN. The final output of our model in the visual servoing system is to use a DNN to perform inferences and produce a probability, i.e. the probability that a wind turbine appears in either, segment S_l or S_r. The next step is to link the probabilities $[P(S_l), P(S_r)]$ to an angle, ψ, which is used as a reference to our feedback loop. In the classical sense this could be done using the $atan2$ function, as described in equation 1.

$$\psi_{ref} = atan2(y_S - y_{UAV}, x_S - x_{UAV}) \tag{1}$$

Where $[x_S, y_S]$ is the segment's center location and $[x_{UAV}, y_{UAV}]$ is the UAV's camera focal point in the Cartesian coordinate system and ψ_{ref} is the reference angle. This method works well if we have a deterministic system, but in our case $[P(S_l), P(S_r)]$ might be change randomly and not being necessary 0 in the case of not 'seeing a wind turbine'.

In this work we propose an alternative method, where the probabilities of the two segments are subtracted as shown in equation 2, we call this the augmented angle $\hat{\psi}$. The three resulting yaw angle ψ cases are shown in equation 3.

$$\hat{\psi} = P(S_l) - P(S_r) \tag{2}$$

$$\begin{cases} \hat{\psi} \to 1 & \text{for} \quad P(S_l) > P(S_r) \\ \hat{\psi} \to (-1) & \text{for} \quad P(S_l) < P(S_r) \\ \hat{\psi} = 0 & \text{for} \quad P(S_l) = P(S_r) \end{cases} \tag{3}$$

The augmented angle $\hat{\psi}$ is used as the reference value, i.e. (ψ_{ref}), for the outer loop controller as shown in figure 5.

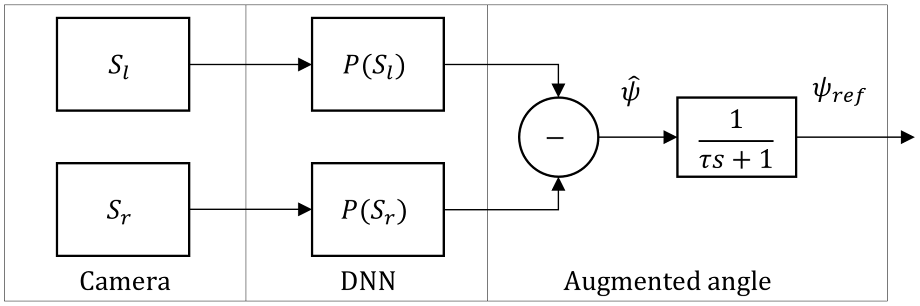

Fig. 2. Block diagram of the visual servoing, which outputs the reference for the outer loop's yaw $[\psi]$ control.

As the probabilities $P(S_l)$, $P(S_r)$ have significant fluctuations in the implementations, a low pass filter is added. The structure of the Visual Servoing is shown in figure 2. Figure 3, shows the raw $\hat{\psi}$ and the filtered $\hat{\psi}$, which is referred to as ψ_{ref} in equation 4.

$$\psi_{ref} = \hat{\psi} \cdot \frac{1}{\tau s + 1} \tag{4}$$

This problem can increase if more segments are included, as the problem is reformulated to sum the probabilities on either side of the image, see equation 6. Where $(\cdot)^n$ indicates n^{th} segment.

$$\hat{\psi} = \sum P(x_l^n, y_l^n) - \sum P(x_r^n, y_r^n) \tag{5}$$

In broader terms our approach increases the level of autonomy, since the DNN is used to compute the trajectory of the UAV.

Practical Approach of This Paper The visual servoing controller uses the DNN's probabilities associated to the *windturbine* class. In this work we used four segments, where the DNNs calculate the probabilities for segments 3 and 4, i.e. the segments with the tags $[t_3, t_4]$ which have the corresponding probabilities $[P_3, P_4]$, and use them to compute the yaw reference $[\psi_{ref}]$, as shown in figure 2. The yaw reference is used to compute the error signal for the yaw controller, as shown in figure 5.

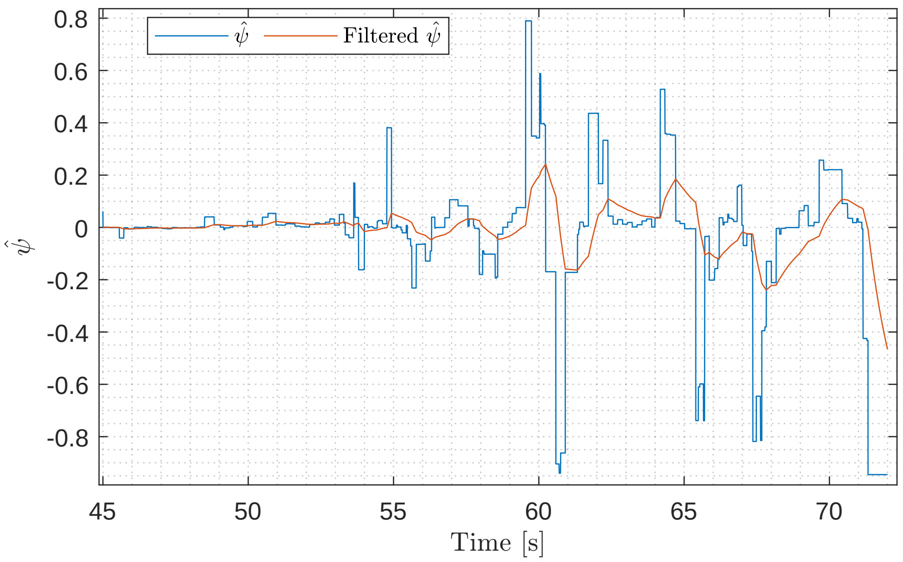

Fig. 3. Raw and filtered $\hat{\psi}$, data stems from the experiment, from **Found** state to **Land** state, refer to figure 9.

2.4 Perception Based Collision Avoidance and Forward Movement Control

The perception based collision avoidance is based on the hypothesis that the closer the UAV is to the object, the larger the sum of probabilities will be (see equation (6)).

$$P_{sum} = \sum_{i=1}^{q} P_i \tag{6}$$

where q is the number of segments, (in this work four segments were used). A threshold for the probability, P_t^{serv}, is defined by the user. This is used to stop the forward movement of the UAV. The forward movement of the UAV is equally controlled using the sum of probabilities, where an increase of the sum is used to decrease the forward velocity of the UAV.

The forward velocity $v = \dot{x}$, i.e. in the x direction as shown in figure 4, is calculated as a function of the sum of probabilities of all segments multiplied by a control parameter b and subtracted from the maximum forward velocity v_{max}, see equation (7).

$$v = v_{max} - b \cdot \sum_{i=1}^{q} P_i \tag{7}$$

$\dot{x}_{max}$ is defined by the user and depends on the maximally allowed velocity for the UAV, the control parameter b is calculated based on equation 8.

$$b = -\frac{v_{max}}{P_t^{serv}} \tag{8}$$

Where v_{max} is the maximum allowable velocity set by the user.

Thus once a wind turbine is detected, the UAV starts navigating towards it with a preset velocity, v_{max}. Then the $P_{sum} = \sum_{i=1}^{q} P_i$ is calculated and used to reduce v_{max}. As the UAV moves nearer to the wind turbine, P_{sum} will increase since wind turbine's apparent size will increase within the field of vision of the camera, eventually reaching P_t^{serv}. The designed control parameter in equation 8, will ensure that the two cases shown in equation 9 hold.

$$v = \begin{cases} > 0 & \text{for } \sum_{i=1}^{q} P_i \quad < P_t^{serv} \\ = 0 & \text{for } \sum_{i=1}^{q} P_i \quad > P_t^{serv} \end{cases} \tag{9}$$

As equation 8

Practical Approach of This Paper In the current implementation the forward movement is controlled by the outer loop controller, which has x_{ref} as the reference as seen in figure 5. The perception based forward movement controller computes a position $x_{ref} = \int v$.

2.5 UAV Dynamics and Navigation Control

In this section, a navigation controller based on visual servoing will be designed for the UAV using PID controllers. The coordinates of the UAV in the Euclidean space is shown in Figure 4.

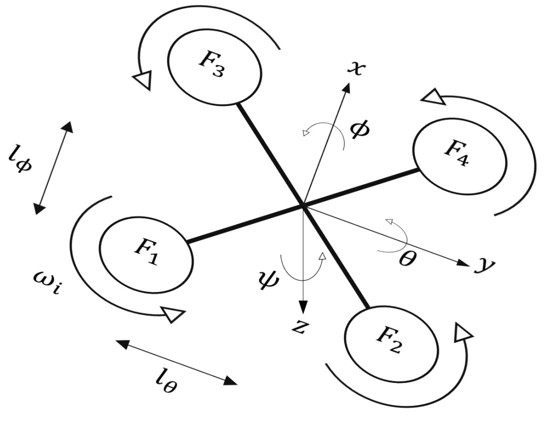

Fig. 4. Kinetic variables and coordinates describing the UAV's dynamic.

The dynamics of the UAV are modeled by a 6-DOF under-actuated nonlinear system as follows (Dikmen et al (2009), Bolandi et al (2013), Choi and Ahn (2015)):

$$m\ddot{\xi} = FRe_3 - mge_3, \tag{10}$$
$$I\ddot{\eta} = \tau - C(\eta, \dot{\eta})\dot{\eta}. \tag{11}$$

where $\xi = [x, y, z]^T$ are the translational positions of the UAV in the inertial frame, $\eta = [\phi, \theta, \psi]^T$ are the Euler angles with ϕ, θ and ψ being the roll, pitch and yaw, respectively, m is the mass of the UAV, R is the rotation matrix from body-fixed frame of the UAV to the inertial frame described as follows

$$R = \begin{bmatrix} c(\theta)c(\psi) & -c(\theta)s(\psi) & s(\theta) \\ c(\phi)s(\psi) + c(\psi)s(\phi)s(\theta) & c(\phi)c(\psi) - s(\phi)s(\theta)s(\psi) & -c(\theta)s(\phi) \\ s(\phi)s(\psi) - c(\phi)c(\psi)s(\theta) & c(\psi)s(\phi) + c(\phi)s(\theta)s(\psi) & c(\phi)c(\theta) \end{bmatrix}, \tag{12}$$

where $s(\cdot) = \sin(\cdot)$ and $c(\cdot) = \cos(\cdot)$, g is the gravitational acceleration, $C(\eta, \dot{\eta})$ is the *Coriolis matrix*, F is the translational force in the inertial frame, I is the inertial matrix and $e_3 = [0, 0, 1]^T$.

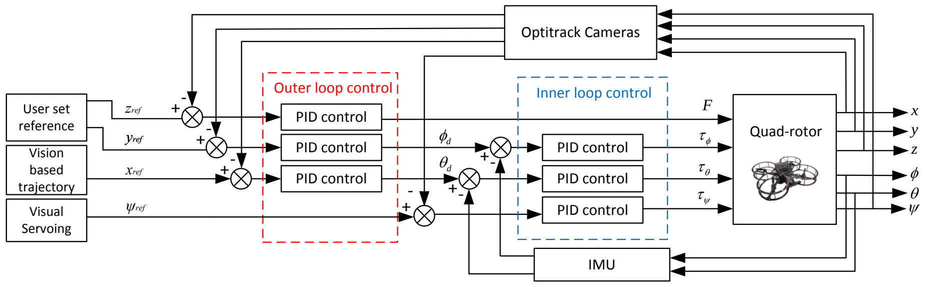

Fig. 5. Structure of navigation control system.

Since the Coriolis matrix is highly nonlinear and thus difficult to determine accurately, we propose to use a series of PID controllers to design the navigation control system, using an inner-outer loop control strategy. See Figure 5. The controller's objective is that the UAV reaches a desired position and attitude $[x_{ref}, y_{ref}, z_{ref}, \psi_{ref}]$, keeping a constant distance away from the wind turbine facing the navigation camera. In practice, this is done to make the UAV reach the starting position for wind turbine inspection.

In the outer loop control, we define the tracking error as $\tilde{\xi} = \xi - \xi_{ref}$. Then, equation (10) can be rewritten as

$$m\ddot{\tilde{\xi}} = FRe_3 - mge_3. \tag{13}$$

We define three virtual control inputs in x, y and z directions as follows:

$$\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} \frac{s(\theta)F}{m} \\ \frac{-c(\theta)s(\phi)F}{m} \\ -g + \frac{c(\phi)c(\theta)F}{m} \end{bmatrix} \tag{14}$$

Here, u_x, u_y and u_z are designed by three PID controllers as follows:

$$u_{x,y,z} = K_p \delta\xi(t) + K_i \int_0^t \delta\xi(\tau) d\tau + K_d \frac{d}{dt} \delta\xi(t). \tag{15}$$

The transfer function of (15) can be written as

$$\frac{u(s)}{\xi(s)} = K_p + \frac{K_i}{s} + K_d s. \tag{16}$$

To get good controller's performance, we tune the control parameters K_p, K_i and K_d through experiments using the PID block in SIMULINK/MATLAB. Then, we can calculate the control force, roll and yaw Euler angles by

$$F = m\sqrt{(u_z+g)^2 + u_x^2 + u_y^2}, \theta_{ref} = \arcsin\left(\frac{mu_x}{F}\right), \phi_{ref} = \arctan\left(\frac{u_y}{u_z+g}\right).$$

Now, the desired Euler angles ϕ_{ref}, θ_{ref} and ψ_{ref} are known. Similarly, other three PID controllers are employed to control the attitude in the inner loop as follows:

$$\tau_{\phi,\theta,\psi} = K_p \delta\eta(t) + K_i \int_0^t \delta\eta(\tau) d\tau + K_d \frac{d}{dt} \delta\eta(t), \tag{17}$$

Control parameters K_p, K_i and K_d are tuned using PID block in the SIMULINK /MATLAB to guarantee the closed-loop stability as well as good control performance.

Finally, we get the 4-dimensional control input (F, τ), which needs to be transformed into PWM pulse and allocated to four motors denoted by $[u_1, u_2, u_3, u_4]$. To balance the motor moments, the linear mapping between control forces and PWM values of motors can be given by

$$\begin{bmatrix} F \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} F_f & F_f & F_f & F_f \\ -F_f l_\phi & -F_f l_\phi & K_f l_\phi & F_f l_\phi \\ F_f l_\theta & -F_f l_\theta & K_f l_\theta & -F_f l_\theta \\ T & -T & -T & T \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}, \tag{18}$$

where F_f is the maximum force of each motor with PWM at 1, l_ϕ is the distance between motors F1 and F2 to the center of the UAV along forward/backward direction as shown in figure 4, l_θ is the distance between motors F1 and F3 to the center of the UAV along sideways direction, and T is the motor torque constant. It is noted that the mapping matrix in equation 18 between the control forces and motor moments is invertible and thus $[u_1, u_2, u_3, u_4]^T$ can be uniquely determined by the control force $[F, \tau_\phi, \tau_\theta, \tau_\psi]^T$.

The feedback parameters x, y, z, ψ are acquired from the opti-track camera system, and the attitude angles ϕ and θ are computed using complimentary filters and the on-board IMU. The full structure is shown in figure 5.

3 Implementation

The algorithms developed in this work were tested in a indoor laboratory environment, with an area of $4.5 \times 4m^2$ with a height of 4 meters. The laboratory set-up, called the 'Autonomous Vehicles Research Studio' (AVRS) is manufactured by Quanser, Quanser (2018), and consists of a quad-copter UAV, GCS and an optical tracking system. In addition a 1:336 scale version of a Vestas V112-3.0 MW wind turbine was used.

3.1 Program Flow

The algorithms implemented follow an event based control scheme shown in figure 6, the events and states are described below.

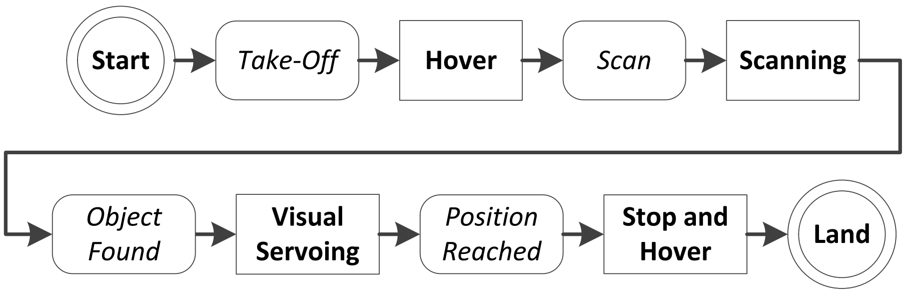

Fig. 6. Flow diagram of the algorithm, consists of six states: 1) **Start**, 2) **Hover**, 3) **Scanning**, 4) **Visual Servoing**, 5) **Stop and Hover** and 6) **Land**, and four events : 1) *Take-off*, 2) *Scan*, 3) *Object Found*, 4) *Position Reached*.

Start Program is initialized, the UAV is stationary on the ground with the motors turned off and the GCS stores the UAV's HOME position. The *take-off* event is activated manually by the operator, which starts the attitude controller and triggers the **Hover** state when the UAV reaches the desired hovering altitude.

Hover The UAV hovers in the HOME position until the *Scan* event is triggered by a timer and the system transitions to the **Scanning** State.

Scanning A scanning algorithm that increments the yaw angle reference $[\psi_{ref}]$ indefinitely, is triggered. During the **Scanning** state the DNN outputs four probabilities $[P_1, P_2, P_3, P_4]$, one for each segment. When the threshold for the probability at quadrant 1 is reached, i.e. $P_1 > P_t^{scan}$, the **Scanning** is halted and the *Object Found* event is triggered and the system transitions into the **Visual Servoing** state. Here P_t^{scan} is the minimum thresold probability for detecting a wind turbine while in the scanning state.

Visual Servoing The UAV starts to move forward in the [x] direction, where the reference x_{ref} is calculated based on the sum of probabilities [P_1, P_2, P_3, P_4]. In addition, [ψ_{ref}] is computed from the two probabilities [P_3, P_4] and the angle [ψ] is controlled by the inner loop controller. When the sum of probabilities have been reached the *Position Reached* event is triggered and the system transitions into the **Stop and Hover** state.

Stop and Hover The **Visual Servoing** continues to keep the UAV true to the wind turbine, and a timer eventually triggers the **Land** State.

Land The UAV lands in front of the wind turbine and the motors are turned off.

4 Experimental Results

The purpose of the experiment was to test the systems ability to scan and find a scaled wind turbine and navigate towards it using visual servoing.

4.1 Experiment Description

The UAV was placed roughly 1.5 m away from the wind turbine, which sat in front of the back wall, as seen in figure 1, and the UAV was angled at 0.8 rad in relation to the OptiTrack reference frame. Figure 7 shows the UAV at four states, where the first image is the **Start** state described above.

The UAV's 3D coordinates from the experiment have been sketched in figure 8, from taking off to after landing. In relation to the 3D plot in figure 8, the back wall is perpendicular to the y axis located at $x \approx 1.5$m, and the net is perpendicular to the back wall and located at $x \approx -2.8$m. The $x, y, z = 0$ is the zeroth coordinate of our lab in the OptiTrack reference frame, which is offset by -0.085 rad from the lab frame.

4.2 Result Description

Figure 9 shows data for ψ angle and x position from the **Visual Servoing** state until the **Land** state.

When the state **Visual Servoing** is triggered the angle of the UAV is -0.28 rad, this angle is caused by the pose of the UAV at the end of the **Scanning** state. The reason for this pose is that the object found event is activated by quadrant 1, which is located on the left side of the image and thus the pose of the UAV will tend to move slightly away from the center to detect the wind turbine. The image from quadrant 1 collected during this experiment can be seen in figure 1 as the top left segment. This initial angle is corrected quickly by the **Visual Servoing** and the UAV reaches an equilibrium point of -0.125 rad, where the frame of the optical tracking system is offset by -0.085 rad from the lab frame.

The ψ angle oscillates slightly through the **Visual Servoing** state, this is caused by the variations in P_3 and P_4 which are used for the visual servoing as can be seen in the bottom plot in figure 10.

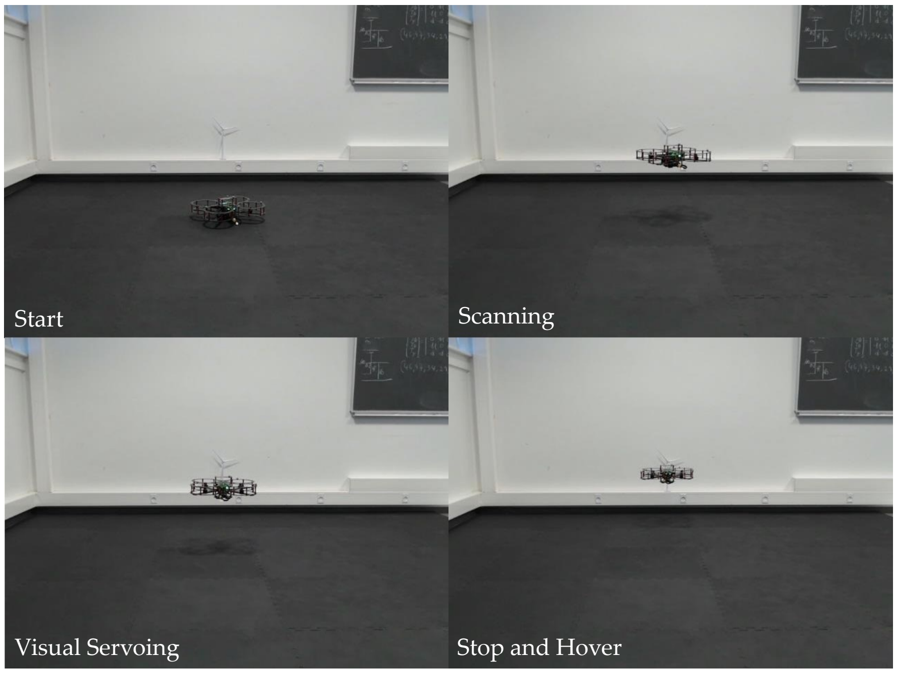

Fig. 7. Experimental results: UAV at four states, **Start**, **Scanning**, **Visual Servoing** and **Stop and Hover**.

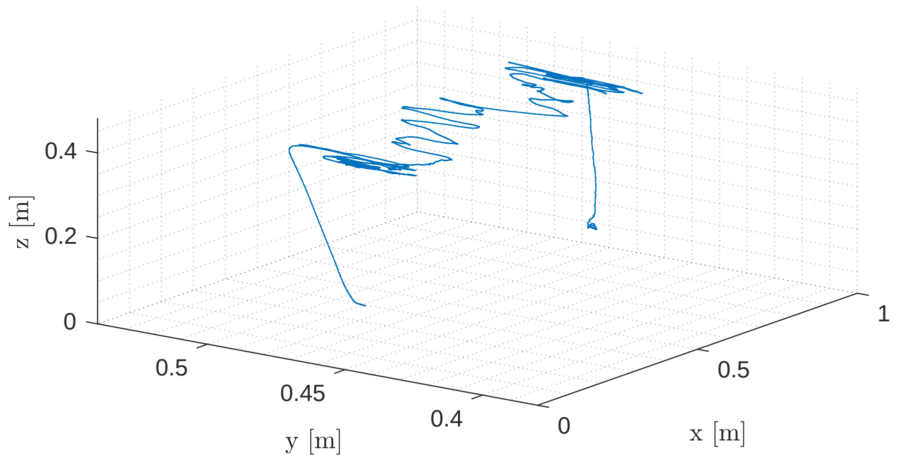

Fig. 8. Path of the UAV in 3D, from takeoff to landing.

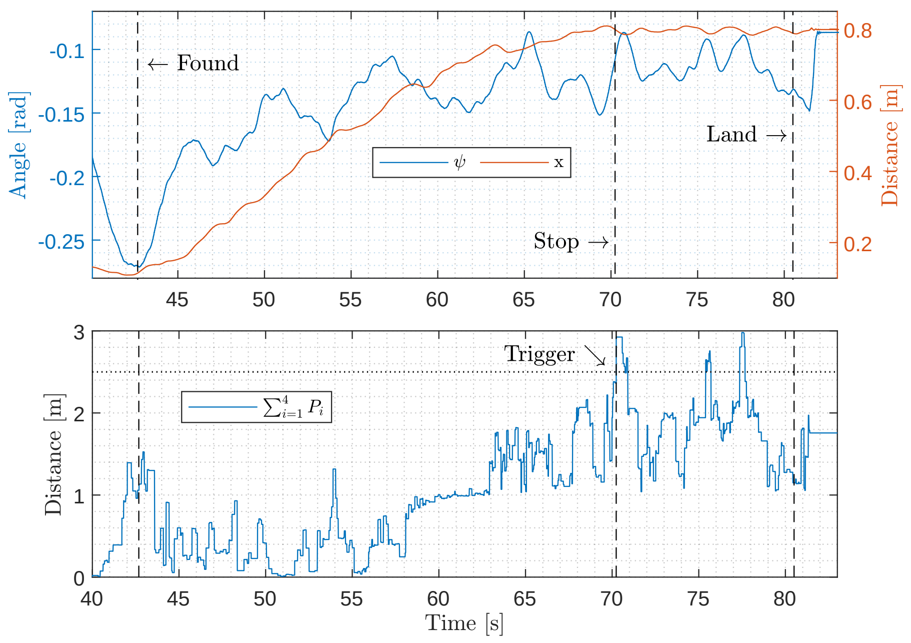

Fig. 9. Test results: *(Top plot)* close-up from the **Visual Servoing** state until the **Land** state. *(Bottom plot)* summed probability.

During the **Visual Servoing** state the v decreases as the UAV approaches the wind turbine, which is consistent with the increase in the summed probability shown in the bottom plot of figure 9. The UAV eventually stops as intended due to the summed probability reaching the threshold value, i.e. $P_t > 2.5$, which is marked by the horizontal dotted line in the bottom plot of figure 9.

During the **Stop and Hover** state, the visual servoing is still active and it continues to keep the UAV's ψ angle straight towards the wind turbine, here the aforementioned ψ offset of -0.085 rad is evident.

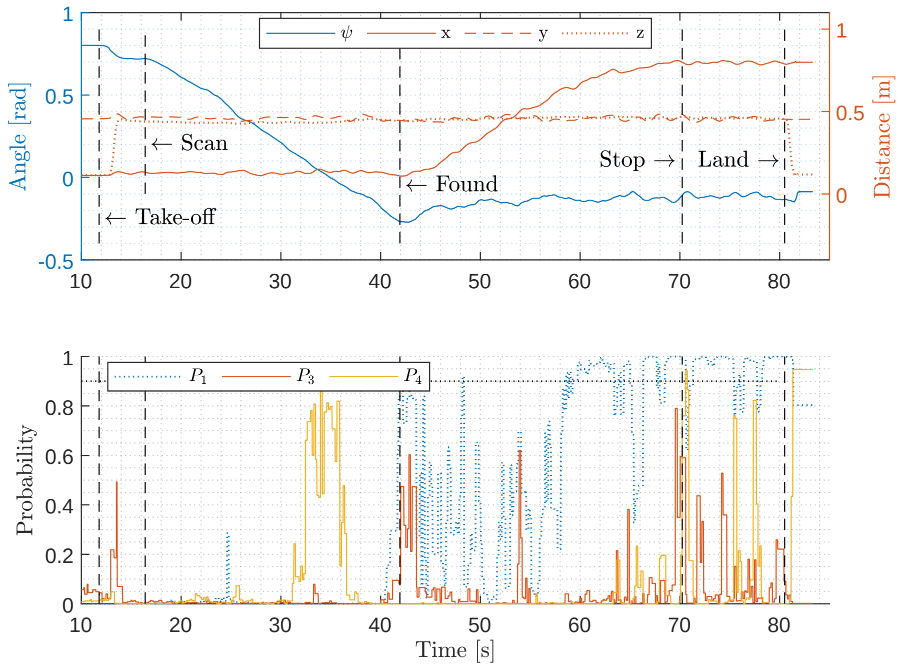

Fig. 10. Test results, *(Top plot)* shows the positions, *(Bottom plot)* shows the probability of quadrant 1 used for detection of target, and the probability of quadrant 3 and 4, used for visual servoing.

5 Discussion

From the current results the algorithm is capable of finding the wind turbine, navigate towards it and stop when the wind turbine has been reached. From the data, it is evident that the system performs well and that every state is performed with success.

During the **Visual Servoing** state the v is relatively low, as it takes the UAV 28s to move roughly 0.7m. This is due to a low $\dot{x}_{max}$, which was chosen to keep the UAV safe during these experiments. During the experiments some issues did arise when the UAV started far away from the wind turbine, as it appeared too small in the image to be detected and we aim at improving this by introducing a higher resolution image with more segments.

In this work the triggering of the *Object Found* event was based on the probability of quadrant 1 P_1, in future work the trigger should instead be based any quadrant which produces a probability larger than the threshold P_t^{scan}. In addition, the choice of P_3 and P_4 for the calculation of the ψ_{ref}, should in future work be done dynamically, depending on the location of the wind turbine relative to the quadrants.

The threshold parameters P_t^{scan} and P_t^{serv} are in the current paper subjected to ad-hoc tuning, future work will involve a more systematic way of tuning these parameters.

6 Conclusions

In this work we have successfully used a DNN to navigate a UAV autonomously, trained from captured images to recognize a wind turbine using transfer learning techniques. The images acquired from an RGB camera on the UAV were segmented to create what we call the compounded eye with four quadrants, where each quadrant's image is processed by a DNN. The probability produced by the DNN in quadrant one is used in searching for the wind turbine, whilst the probabilities of quadrant three and four are used for visual servoing to perform perception-based navigation. The algorithm for visual servoing, uses the probabilities of quadrant three and four to obtain a reference for the UAV's yaw position, which is then fed to the UAV's outer loop controller. The sum of the probabilities obtained in all quadrants is used in a velocity and collision avoidance controller. The controller computes a desired x reference and feeds it to the UAV's outer loop controller, enabling the UAV to safely reach the wind turbine and stop in front of it for inspection purposes. The algorithm was initially tested within a simulation environment as a proof of concept. Our preliminary experiments, show that the UAV can autonomously find and navigate towards the wind turbine, control its speed towards it and stop at a safe distance. In our future work we will use a larger amount of quadrants and incorporate in the algorithm the calculation of pitch and roll angles additionally to implement a complete visual servoing navigation system.

Bibliography

Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, Jackel LD, Monfort M, Muller U, Zhang J, et al (2016) End to end learning for self-driving cars. arXiv preprint arXiv:160407316

Bolandi H, Rezaei M, Mohsenipour R, Nemati H, Smailzadeh SM (2013) Attitude control of a quadrotor with optimized pid controller. Intelligent Control and Automation 4(03):335

Choi YC, Ahn HS (2015) Nonlinear control of quadrotor for point tracking: Actual implementation and experimental tests. IEEE/ASME transactions on mechatronics 20(3):1179–1192

Day D (2017) Drones for transmission infrastructure inspection and mapping improve efficiency. Natural Gas & Electricity 33(12):7–11

Dikmen C I, Arisoy A, Temeltas H (2009) Attitude control of a quadrotor. In: Recent Advances in Space Technologies, 2009. RAST'09. 4th International Conference on, IEEE, pp 722–727

Drews P, Williams G, Goldfain B, Theodorou EA, Rehg JM (2018) Vision-based high speed driving with a deep dynamic observer. arXiv preprint arXiv:181202071

Drews P, Williams G, Goldfain B, Theodorou EA, Rehg JM (2019) Vision-based high-speed driving with a deep dynamic observer. IEEE Robotics and Automation Letters 4(2):1564–1571

Durdevic P, Ortiz-Arroyo D, Li S, Yang Z (2019) Vision aided navigation of a quad-rotor for autonomous wind-farm inspection. In: I F A C Workshop Series, IFAC-PapersOnLine - (In Press)

Espiau B, Chaumette F, Rives P (1992) A new approach to visual servoing in robotics. IEEE Transactions on Robotics and Automation 8(3):313–326, DOI 10.1109/70.143350

Flener C, Vaaja M, Jaakkola A, Krooks A, Kaartinen H, Kukko A, Kasvi E, Hyyppä H, Hyyppä J, Alho P (2013) Seamless mapping of river channels at high resolution using mobile lidar and uav-photography. Remote Sensing 5(12):6382–6407

Giusti A, Guzzi J, Ciresan DC, He FL, Rodríguez JP, Fontana F, Faessler M, Forster C, Schmidhuber J, Di Caro G, et al (2016) A machine learning approach to visual perception of forest trails for mobile robots. IEEE Robotics and Automation Letters 1(2):661–667

Goldfain B, Drews P, You C, Barulic M, Velev O, Tsiotras P, Rehg JM (2019) Autorally: An open platform for aggressive autonomous driving. IEEE Control Systems Magazine 39(1):26–55

Ho H, Chu Q (2013) Automatic landing system of a quadrotor uav using visual servoing. Proceedings of the EuroGNC 2013, 2nd CEAS Specialist Conference on Guidance, Navigation and Control pp 1264–1283

Jamieson P, Hassan G (2011) Innovation in wind turbine design, vol 2. Wiley Online Library

Kanellakis C, Nikolakopoulos G (2017) Survey on computer vision for uavs: Current developments and trends. Journal of Intelligent & Robotic Systems 87(1):141–168, DOI 10.1007/s10846-017-0483-z, URL https://doi.org/10.1007/s10846-017-0483-z

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105

Lattanzi D, Miller G (2017) Review of robotic infrastructure inspection systems. Journal of Infrastructure Systems 23(3):04017,004

Loquercio A, Maqueda AI, del Blanco CR, Scaramuzza D (2018) Dronet: Learning to fly by driving. IEEE Robotics and Automation Letters 3(2):1088–1095

Metni N, Hamel T (2007) A uav for bridge inspection: Visual servoing control law with orientation limits. Automation in Construction 17(1):3 – 10, DOI https://doi.org/10.1016/j.autcon.2006.12.010, URL http://www.sciencedirect.com/science/article/pii/S0926580507000052

Mondragón IF, Olivares-Méndez MA, Campoy P, Martínez C, Mejias L (2010) Unmanned aerial vehicles uavs attitude, height, motion estimation and control using visual systems. Autonomous Robots 29(1):17–34, DOI 10.1007/s10514-010-9183-2, URL https://doi.org/10.1007/s10514-010-9183-2

Moolan-Feroze O, Karachalios K, Nikolaidis DN, Calway A (2019) Improving drone localisation around wind turbines using monocular model-based tracking. arXiv preprint arXiv:190210474

Morgenthal G, Hallermann N (2014) Quality assessment of unmanned aerial vehicle (uav) based visual inspection of structures. Advances in Structural Engineering 17(3):289–302

Nicolas Guenard RM Tarek Hamel (2008) A practical visual servo control for an unmanned aerial vehicle. IEEE Transactions on Robotics 24(2):331–340

Nikolov I, Madsen CB (2017) Lidar-based 2d localization and mapping system using elliptical distance correction models for uav wind turbine blade inspection. In: VISIGRAPP (6: VISAPP), pp 418–425

Pomerleau DA (1989) Alvinn: An autonomous land vehicle in a neural network. In: Advances in neural information processing systems, pp 305–313

Pomerleau DA (1991) Efficient training of artificial neural networks for autonomous navigation. Neural Computation 3(1):88–97

Quanser (2018) Autonomous Vehicles Research Studio. URL https://www.quanser.com/products/autonomous-vehicles-research-studio

Rakha T, Gorodetsky A (2018) Review of unmanned aerial system (uas) applications in the built environment: Towards automated building inspection procedures using drones. Automation in Construction 93:252–264

Rusnák M, Sládek J, Kidová A, Lehotskỳ M (2018) Template for high-resolution river landscape mapping using uav technology. Measurement 115:139–151

Sa I, Hrabar S, Corke P (2014) Inspection of pole-like structures using a vision-controlled vtol uav and shared autonomy. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 4819–4826, DOI 10.1109/IROS.2014.6943247

Shakmak B, Al-Habaibeh A (2015) Detection of water leakage in buried pipes using infrared technology; a comparative study of using high and low resolution infrared cameras for evaluating distant remote detection. In: 2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), IEEE, pp 1–7

Smolyanskiy N, Kamenev A, Smith J, Birchfield S (2017) Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness. arXiv preprint arXiv:170502550

Stokkeland M, Klausen K, Johansen TA (2015) Autonomous visual navigation of unmanned aerial vehicle for wind turbine inspection. In: 2015 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, pp 998–1007

Wang L, Zhang Z (2017) Automatic detection of wind turbine blade surface cracks based on uav-taken images. IEEE Transactions on Industrial Electronics 64(9):7293–7303