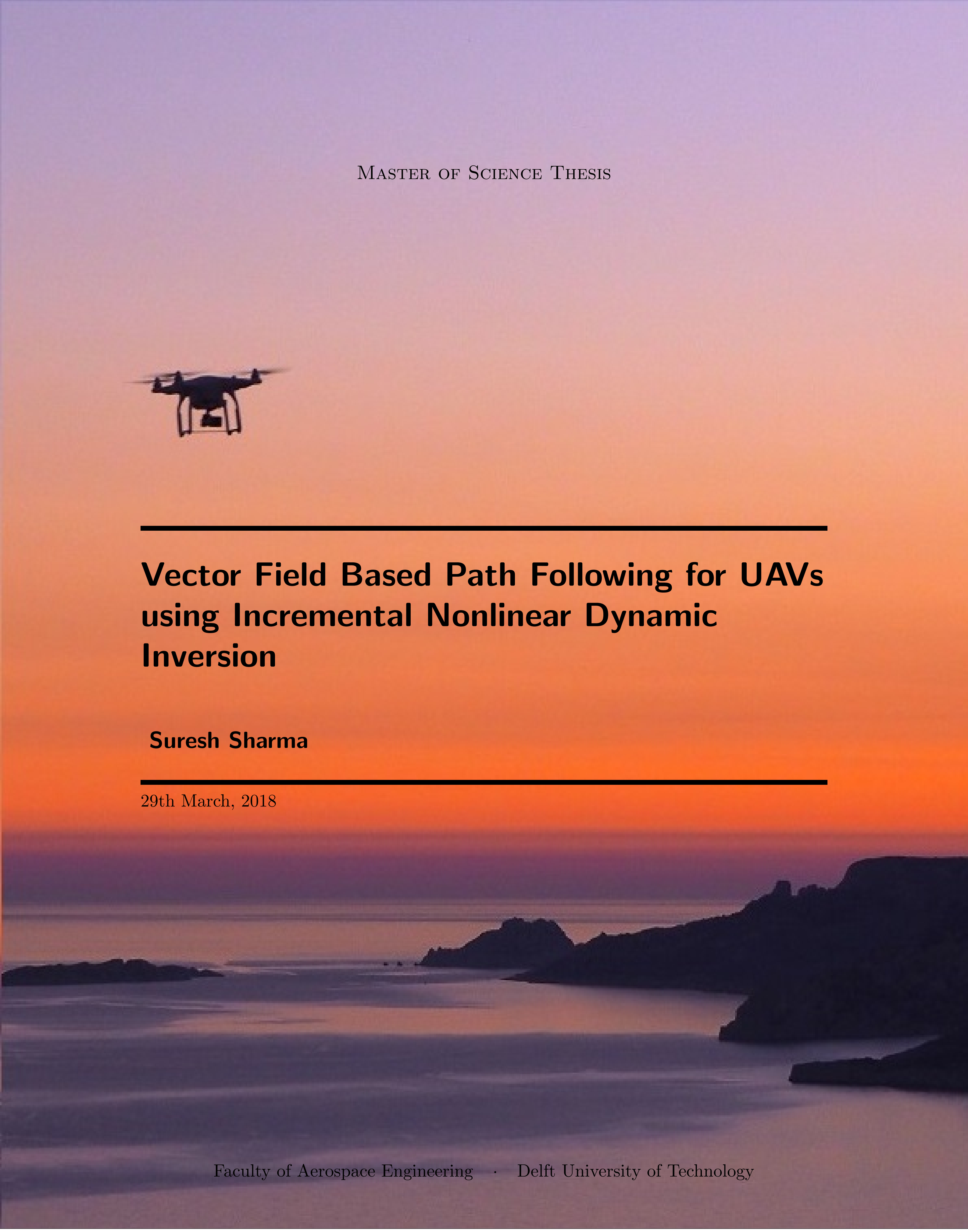
Master of Science Thesis



Vector Field Based Path Following for UAVs using Incremental Nonlinear Dynamic Inversion

Suresh Sharma

29th March, 2018

Faculty of Aerospace Engineering · Delft University of Technology

Vector Field Based Path Following for UAVs using Incremental Nonlinear Dynamic Inversion

Master of Science Thesis

to obtain the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

Suresh Sharma

29th March, 2018

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Delft University Of Technology
Department Of
Control and Operations

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Vector Field Based Path Following for UAVs using Incremental Nonlinear Dynamic Inversion"** by **Suresh Sharma** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 29th March, 2018

Readers:

Dr. ir. Q.P.Chu

Ir. R. Noomen

Dr. ir. E. van Kampen

Ir. E. J. J. Smeur

Summary

As Unmanned Air Vehicles (UAVs) are getting smaller and cheaper, their applications in urban environments are increasing. Multirotor UAVs are especially suitable for use in complex environments due to their ability to hover and fly in any direction.An important requirement of UAVs performing missions in areas such as cities is the ability to accurately follow a specified path, even in the presence of external disturbances such as wind.

Current common methods of path following present certain flaws inherent to the methods, such as being able to follow only straight lines or circles, and following a "Virtual Target Point" instead of actually following the path, which cannot guarantee the accuracy of path following. A more robust technique of path following involves the use of vector fields, which are spatial functions that assign a vector to each point in a subset of space.

In this work, a vector field is used to calculate the acceleration commands which are provided to a multirotor UAV equipped with an acceleration and attitude controller. The vector field based controller is computed in two steps. The first step involves the generation of a vector field around the desired path, which can be a smooth planar shape described in its implicit form. The vector field is constructed such that its integral curves when followed, lead to the desired path. The second step is to compute the desired accelerations that drive the UAV along the integral curves of the vector field and therefore to the desired path.

The result is a vector field based controller that is capable of controlling the position of the UAV, and also independently the velocity. This enables the velocity of the UAV to either be kept constant or to follow a desired speed profile. The algorithm is analysed by means of real-world flight tests, and its performance is compared with the traditional carrot-chasing path following controller. It is seen that the vector field based controller performs significantly better at maintaining a lower tracking error, particular as the desired velocity of the UAV is increased.

Some initial work is also presented in following more complex paths using the vector field based controller, by means of discretizing the path into simpler path segments and combining the vector fields for the individual path segments. Once again it is seen that the vector field method has a lower tracking error, preventing overshoot at points where the different path segments meet.

Finally, some discussion on the results obtained, and recommendations for increasing performance in future work are provided.

Acknowledgements

I would like to begin by taking the time to thank the people who have made my time at TU Delft possible and enjoyable.

Firstly, I would like to express my gratitude to my family for their unconditional love and support, and for making numerous sacrifices along the way to ensure I receive the best education possible. With the completion of my Masters at TU Delft, I hope to make them proud of what I have accomplished so far and I hope the knowledge and experience I have gained at this institution will enable me to continue doing so in the future.

I would like to thank Ewoud Smeur for his guidance and tremendous patience throughout the duration of my graduation. He always made himself available for meetings when I was stuck or confused, and his expertise and insights have been extremely helpful in the completion of this research work. I would also like to thank Dr. Q. P. Chu for his support and feedback, and Hector Garcia de Marina for laying the foundations on which my research work was built.

Lastly, but equally importantly, I would like to thank my friends who have been an integral part of the past few years of my life- particularly Federico Paredes, Clara Soriano, Wouter Plaetinck, and Marta Camarero.

Delft, The Netherlands
29th March, 2018

Suresh Sharma

Contents

Summary v

Acknowledgements vii

List of Figures xi

List of Tables xiii

I Scientific Paper 1

II Thesis 13

1 Introduction 15

2 Literature Survey 17

2.1 Virtual Target Point based Methods . . . 17
2.2 LQR based Path-Following . . . 18
2.3 Model Predictive Control based Path-Following . . . 18
2.4 Vector Field based Methods . . . 19

3 Guidance Vector Field Path-Following for Fixed Wing UAVs 21

3.1 Problem Definition . . . 22
3.1.1 UAV Kinematic Model . . . 22
3.1.2 Description of the path . . . 22
3.1.3 Technical Assumptions . . . 23
3.2 Properties of the Guidance Vector Field . . . 23
3.3 Path Following Controller . . . 25

3.4 Extensions of GVF 25
3.5 Practical Implementation 26
3.5.1 Incorporation of Wind Disturbance 26
3.5.2 Implementation on a UAV 27

4 Incremental Nonlinear Dynamic Inversion Acceleration Controller 29
4.1 Filtering 30

5 Experiments and Results 33
5.1 Experimental Setup 33
5.2 Circular Path Following 34
5.2.1 Results 34
5.2.2 Discussion 35

6 Conclusion and Future Work 39
6.1 Future Work 40

List of Figures

3.1 Representation of the UAV and the level sets. 24

5.1 Average Distance Error for all experiments 35
5.2 Position of the UAV for 3 radii and 3 velocities 36
5.3 Distance of the UAV from the centre for 3 radii and 3 velocities 37

List of Tables

5.1 Avg. Distance Errors (in m) when using Carrot-chasing 34
5.2 Avg. Distance Errors (in m) when using Vector Field Method 35

Part I

Scientific Paper

Vector Field Based Path Following for UAVs using Incremental Nonlinear Dynamic Inversion

Suresh Sharma, Ewoud Smeur, Qiping Chu

Abstract—**This paper presents a vector field based path-following method to be used by multirotor Unmanned Aerial Vehicles (UAVs). The desired path to be followed is a smooth planar path defined in its implicit form. The vector field around the desired path is then constructed using the implicit function, such that the integral curves of the vector field converge to the path. The algorithm takes into account the future change in the trajectory as well as the current state of the UAV in order to calculate the desired linear acceleration, which is then tracked using the Incremental Nonlinear Dynamic Inversion (INDI) controller in the autopilot. The implementation also allows for the velocity of the UAV to be controlled independently. Efficiency of the algorithm is demonstrated using real world flight tests, and the performance is shown to be better than the traditional carrot-chasing controller.**

Keywords*—*vector field, path following, multirotor, incremental nonlinear dynamic inversion.

I. Introduction

Unmanned Air Vehicles (UAVs), both fixed-wing and multi-rotors, have experienced a great increase in their technological development in the past few years, which has led to a surge in their popularity for missions such as surveillance, mapping and search and rescue [1], [2]. Due to miniaturization and the reduction in their cost, they are also increasingly used in urban environments for tasks such as autonomous delivery and traffic monitoring.

Almost all the missions and tasks that the UAVs are used for, require that the UAV is capable of following a predefined path with high accuracy. There exist numerous solutions that deal with the problem of path-following [3]. For real world applications, it is crucial that the path following algorithm used is robust to external disturbances such as wind, and is not too computationally complex, so that it can be run in real time on-board the UAV. Such a robust and error-free path following solution will be beneficial not just to UAVs, but also to ground-based robots [4], self-driving cars [5] and Unmanned Underwater Vehicles (UUVs) [6].

When attempting to follow a predefined path, an important consideration is whether the path also imposes temporal constraints. If it does, the problem is that of *trajectory tracking*, which is defined as the tracking of a time-parameterized reference path [7], [8]. This implies that the UAV is commanded to be at a specific point on the path, at a specific point in time. Therefore, trajectory tracking is the technique used in the case of missile guidance control [9], [10].

In contrast, when the path does not impose temporal constraints, the problem is that of *path following*. In this case, the goal is to drive the "cross-track error" to zero, which is the difference between the UAV's position and the desired flight path. Therefore, the path is not a function of time, which makes implementation easier, and smooth convergence is achieved. For these reasons, path following is the approach adopted in this research. Therefore, the speed profile of the UAV can be controlled as an additional control variable. The speed can either be held constant, or adjusted online, therefore providing the capability of meeting some temporal specifications, that is typical of trajectory tracking tasks.

A technique that is currently widely used for path following involves the use of a "virtual target point" (VTP) which lies on the path, at a short distance ahead of the vehicle, known as the "look-ahead distance". Examples for path-following techniques that employ the use of a VTP are the carrot-chasing algorithm [3], which is the algorithm used in the Paparazzi UAV autopilot [11], and the Nonlinear Guidance Law [12]. While straightforward to understand and easy to implement, such methods are limited in their nature, and are typically only suitable for straight line or circular paths. A summary of the working of the carrot-chasing algorithm is provided in the next section.

Path following techniques can be applied to more general shapes using Vector Fields (VF). Vector fields are spatial functions that assign a vector to each point in space. In the context of UAV flight control, each point in either 3D or 2D space would be associated with a particular velocity vector, which is commanded to the UAV. A vector field based path following technique for nonholonomic mobile robots in developed in [13]. The path is described in its implicit form, a suitable vector field is computed and a controller is developed which steers the robot to the desired trajectory. The work from [13] is extended to be used on fixed-wing UAVs in the presence of wind in [14].

The goal of this research is to develop a path-following controller based on vector-fields for use on a multirotor UAV, that provides better path following performance than the carrot-chasing controller that is currently used. The work in this research is an extension of the technique developed in [13] and [14]. In particular, the vector field based technique is formulated to be used as a position and velocity controller on a multirotor UAV, that is equipped with acceleration and attitude controllers. The technique is also verified by means of flight tests on an actual multirotor UAV.

This paper is organized as follows. In Section II, a basic overview of the various control loops in the autopilot is provided, and the three controllers belonging in the "outer loop" are examined in some detail, as the vector field based controller developed in this research belongs in the outer loop.

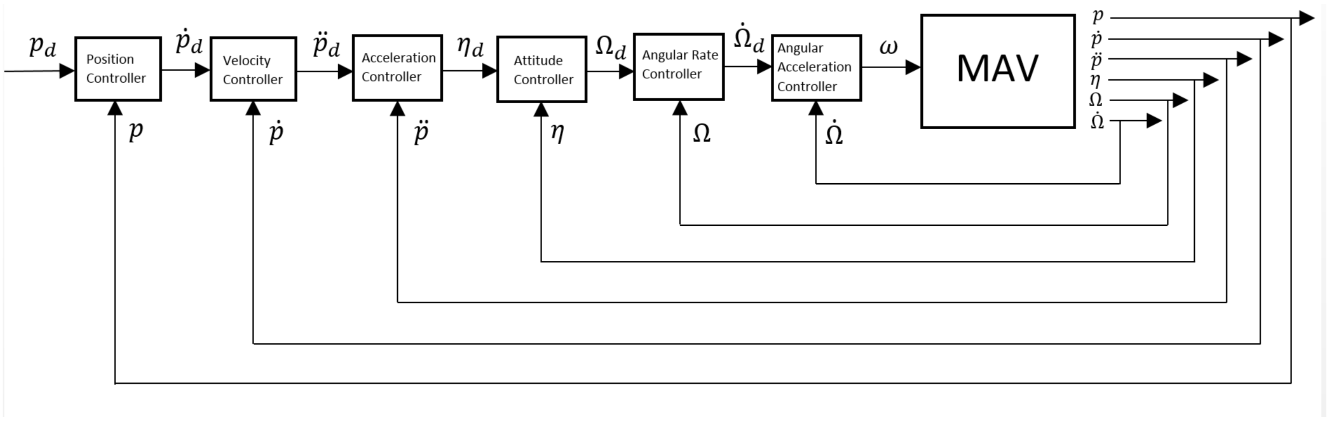


Fig. 1: Control Loop Structure

This is followed by a description of the representation of the path in Section III. The vector-field based control law is derived in Section IV, and the vector field algorithm is tested experimentally and compared against the carrot chasing algorithm in Section V. Finally, the conclusions and future work are discussed in Section VI.

II. Control Loops

A common strategy followed in the control of UAVs is to have multiple loops in a cascaded fashion with each loop controlling a particular state of the UAV. It is also typical to group the control loops into 2 parts: an "Inner Loop" that controls the attitude of the UAV, and an "Outer Loop" which controls the position of the UAV. In Figure 1, starting from the inside, the angular acceleration, angular rate and attitude control loops constitute the "Inner Loop", and the acceleration, velocity and position control loops constitute the "Outer Loop". The functioning of these loops is explained in this section.

A. Inner Loop

The innermost loop in Figure 1 is an angular acceleration control loop, which, as the name suggests, accurately controls the angular acceleration $\dot{\Omega}$ of the UAV, by generating the desired motor commands u. The angular velocity vector ω of the rotors of the UAV is computed from these desired motor commands and sent directly to the UAV.

Incremental Nonlinear Dynamic Inversion (INDI), a sensor-based angular acceleration controller for UAVs is developed in [15]. The controller does not require a model of the UAV, and instead relies only on a model of the actuator effectiveness and an estimation of the angular accelerations of the UAV. In this paper, this INDI attitude controller is used as the innermost angular acceleration loop.

The angular acceleration loop requires a stabilizing reference command, which it receives from the angular rate control loop around it. Here, the angular rate of the UAV is represented as Ω, and the attitude of the UAV is represented as η. The angular rate control loop is surrounded by an attitude control loop. The angular rate control loop and the attitude control loop are simple P controllers, by using a gain on the error in the angular rate, K_Ω and another gain on the error in attitude angles, K_η.

B. Outer Loop

The outer loop is responsible for higher level position control of the UAV. It consists of an acceleration control loop, which receives the desired acceleration command as well as the acceleration of the UAV, and calculates the desired attitude commands which it sends to the attitude controller in the inner loop. Around it is the velocity control loop, which is surrounded by the position control loop. These loops are P controllers. Each of the control loops within the outer loop are discussed in detail below.

1) INDI Acceleration Controller: The INDI angular acceleration controller developed in [15] was extended to also be used as an acceleration controller in [16], and this is the acceleration controller used in this paper. It will be referred to as "Outer INDI controller" here onwards. A short overview of it's functioning is provided here, and a more detailed explanation of the working of this controller can be found in Chapter 4 of the thesis.

Newton's second law of motion can be used to obtain the positional dynamics of a UAV in the NED frame,

$$\ddot{p} = g + \frac{1}{m}F(\dot{p}, w) + \frac{1}{m}T_N(\eta, T) \tag{1}$$

where, p is the position, $\dot{p}$ the velocity and $\ddot{p}$ the acceleration in the NED frame. The acceleration due to gravity is denoted as g, m is the mass, F is the total aerodynamic force acting on the UAV, as a function of the velocity $\dot{p}$ of the UAV and the wind vector w, and T_N is the thrust vector in the NED frame, as a function of the attitude of the UAV, $\eta = [\phi \;\; \theta \;\; \psi]^T$ and the total thrust, T. The total thrust T is the sum of the thrust produced by the individual rotors in the body frame.

This thrust in the body frame is then multiplied by a rotation matrix to obtain the thrust in the NED frame.

A first order Taylor expansion is applied to Equation (1), the result of which is further simplified by means of a few assumptions. For the sake of simplicity, an aerodynamic drag model of the airframe is not used. Therefore, the partial derivative terms of the aerodynamic force F with respect to the velocity of the UAV, and with respect to the wind are neglected. However, not all aerodynamic forces are neglected, as explained later.

After incorporating the above mentioned assumptions, Equation (1) can be re-written as

$$\ddot{p} = \ddot{p}_0 + \frac{1}{m} G(\eta_0, T_0)(u - u_0) \tag{2}$$

where $\ddot{p}_0$ is the acceleration at the previous time step, obtained from an accelerometer measurement, which therefore also takes into account the aerodynamic force acting on the UAV at the previous timestep. The vector $u = [\phi \quad \theta \quad T]^T$, and G is the matrix given below, where sin and cos are abbreviated as 's' and 'c' respectively.

$$G(\eta, T) = \begin{bmatrix} (c\phi s\psi - s\phi c\psi s\theta)T & (c\phi c\psi c\theta)T & s\phi s\psi + c\phi c\psi s\theta \\ (-s\phi s\psi s\theta - c\psi c\phi)T & (c\phi s\psi c\theta)T & c\phi s\psi s\theta - c\psi s\phi \\ -c\theta s\phi T & -s\theta c\phi T & c\phi c\theta \end{bmatrix} \tag{3}$$

The accelerometer measurement used to obtain $\ddot{p}_0$ is noisy, and thus needs to be filtered. Hence, all terms with subscript 0 are filtered, and given the subscript $*$. In Equation (2), $\ddot{p}$ represents the desired acceleration, and u represents the command that will be sent to the inner loop controller. Therefore these terms are given the subscripts d and c respectively. The INDI acceleration control law is then obtained by rearranging Equation (2)

$$u_c = u_* + mG^{-1}(\eta_0, T_0)(\ddot{p}_d - \ddot{p}_*) \tag{4}$$

The INDI acceleration controller receives the desired acceleration command from the velocity controller, which is explained next.

2) Velocity Controller: The velocity controller is also a simple P controller. It receives the desired velocity $\dot{p}_d$ from the position controller, as well the actual velocity of the UAV, from a GPS measurement for example. It then applies a speed gain K_s to the velocity error to obtain the desired acceleration command.

3) Carrot-Chasing Position Controller: Finally, the outermost loop is the position control loop. This is a P controller as well, as it generates a desired velocity command by applying a position gain K_p to the position error $(p_d - p)$, and sends this reference command to the velocity controller.

The position controller obtains the desired position p_d using a technique known as carrot-chasing, a brief description of which is provided here, based on the work in [3]. Carrot-chasing is a simple algorithm that generates the desired position of the UAV. It is currently used in the Paparazzi UAV autopilot system [11], and it is the base algorithm that the vector-field algorithm developed in this paper will be compared against.

Carrot-chasing is a Virtual Target Point (VTP) based method. The approach is to place a VTP on the desired path, which can be either a circle or a straight line. The VTP is then used as the desired position p_d.

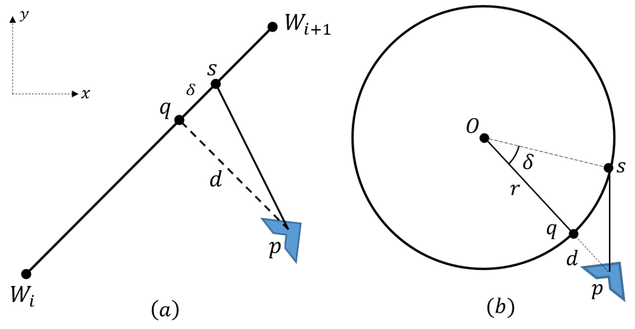


Fig. 2: (a) Straight Line Path (b) Circular Path

Straight line path segments: A straight line segment is represented using 2 waypoints, shown in Figure 2(a). Assume the UAV is at the point p. The first step is to determine the cross-track error, d. The projection of p onto the desired path is the point q, and the distance is d. The VTP, s is then placed a fixed distance ahead of the point q, called the "look-ahead distance", represented by δ. The position of the VTP s is then used as the desired position of the UAV p_d. Thus at each time step, p moves closer to the path, and s moves along the path, thereby making the UAV approach the desired path. The performance of the algorithm can be tuned by changing the value of δ.

Circular path segments: For a circular path segment, the UAV is commanded to loiter around a point O, at a fixed distance r, as shown in Figure 2(b). Assume the UAV is at the point p. The cross track error is then $d = ||O - p|| - r$. The point q is the projection of p onto the circumference of the desired circle. The VTP, s is placed at a fixed angle ahead of the point q, called the "look-ahead angle", represented by δ. Once again, the performance of the algorithm can be tuned by changing the value of δ.

C. Replacing the Position and Velocity Control Loops

The vector field based controller that is developed in this paper is capable of controlling both the position and velocity of the UAV, and therefore replaces the two outermost position and velocity control loops in Figure 1. The vector field controller must also generate the desired acceleration $\ddot{p}_d$, which the outer INDI controller requires as an input. The way in which this acceleration command is computed is explained in the following sections.

III. Path Description

The desired path of the UAV is represented as $\mathcal{P} \in \mathbb{R}^2$ and it is described as

$$\mathcal{P} := \{p : \varphi(p) = 0\} \tag{5}$$

where $p = (x, y)$ is a point in the plane $\mathbb{R}^2$, which is covered by disjointed sets of the function $\varphi(p)$, as shown in Figure 3. These sets are known as *level sets*, and for a particular value of p, the level set $\varphi(p)$ has a constant value, that is, $\varphi(p) = c \in \mathbb{R}$. This function, $\varphi : \mathbb{R}^2 \to \mathbb{R}$, must satisfy the following two criteria:

1) The function φ must belong to the C^2 class, that is, the first and second derivative must both exist and be continuous.
2) The function must be *regular* in the neighborhood of $\mathcal{P}$, that is, the gradient of the function must exist.

$$\nabla\varphi(p) \neq 0, \qquad p \in \mathcal{N}_{\mathcal{P}} \tag{6}$$

where, the neighborhood $\mathcal{N}_{\mathcal{P}} := \{p : |\varphi(p)| \leq c^*\}$ for a constant $c^* \in \mathbb{R}^+$.

Equation (5) states that the *zero level set*, that is, the level set for which $\varphi(p) = 0$, corresponds to the desired path. Thus, the value of the level set $\varphi(p)$ can be interpreted as the signed distance error e of the UAV at the point p.

$$e(p) := \varphi(p) \in \mathbb{R} \tag{7}$$

Therefore, the goal of the path-following controller is to calculate the necessary acceleration commands that would drive the error $e(t) \to 0$ as $t \to \infty$, which, due to (5) would imply that the position of the UAV remains on the desired path $\mathcal{P}$.

IV. Vector Field Control Law

In this section, the vector field based control law is derived. At each point $p = (x, y)$ in the vector field, the normal vector $n(p)$ and the tangent vector $\tau(p)$ to the corresponding level set $\varphi(p)$ are defined, as shown in Figure 3. The normal vector is given by the gradient at the point, $n(p) = \nabla\varphi(p)$, and the tangent vector is given by the rotation of the normal vector

$$\tau(p) = En(p), \qquad E = \begin{bmatrix} 0 & r \\ -r & 0 \end{bmatrix}$$

The direction of rotation of the normal vector, and therefore the direction of the tangent vector and subsequently the direction that the path is followed in, is given by the matrix E, in which r can take the value of either -1 or $+1$.

The control law that drives the UAV to the desired path $\mathcal{P}$ is obtained in two steps. The first step is the design of the vector field, whose integral curves lead to the path. The second step is the calculation of the desired accelerations in order for the UAV to follow the integral curves of the vector field at the desired velocity.

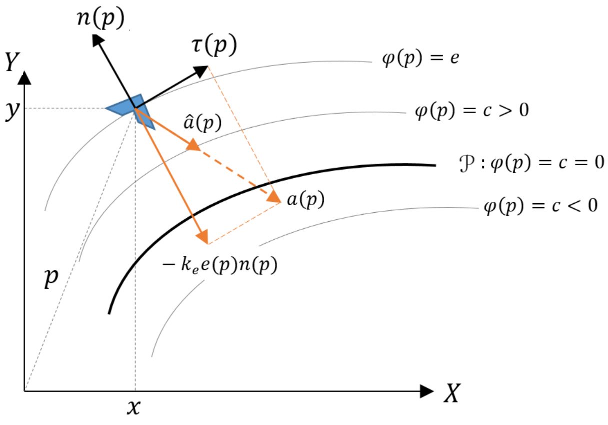


Fig. 3: Representation of the UAV and level sets

A. Vector Field Design

The goal with this step is to obtain a vector field, such that the absolute value of the tracking error $|e|$ decreases to zero along its integral curves. For this purpose, consider the following equation

$$a(p) = \tau(p) - k_e e(p) n(p) \tag{8}$$

where $a(p)$ is a "pseudo-velocity" vector that points in the direction of the desired velocity. The magnitude of this "pseudo-velocity" vector is irrelevant, as it is normalized next, therefore preserving only the information about the direction of the desired velocity. On normalizing $a(p)$, we obtain

$$\hat{a}(p) = \frac{a(p)}{||a(p)||} = \frac{\tau(p) - k_e e(p) n(p)}{||a(p)||} \tag{9}$$

where $\hat{a}(p)$ is a unit vector that points in the direction of the desired velocity. Both $a(p)$ and $\hat{a}(p)$ can be seen in Figure 3. The direction of this unit vector depends on the value of $k_e \in \mathbb{R}^2$, which is a gain that controls the attractive force of the vector field. The effect of k_e on the vector field can be seen in Figure 4.

Apart from k_e, the direction of vector $\hat{a}(p)$ is also influenced by the tracking error $e(p)$. When the UAV is on the path $\mathcal{P}$, the tracking error $e(p) = 0$, therefore the unit vector points solely in the direction of the tangent to the path. For $e(p) \neq 0$, the vector points in a direction towards the path.

Thus, (9) defines the desired vector field whose integral curves lead to the desired path. Some examples of various paths and their corresponding vector fields can be seen in Figures 4 and 5.

B. Calculation of Desired Accelerations

Once the vector field is obtained, the right acceleration that enables the UAV to follow the integral curves of the vector field must be calculated.

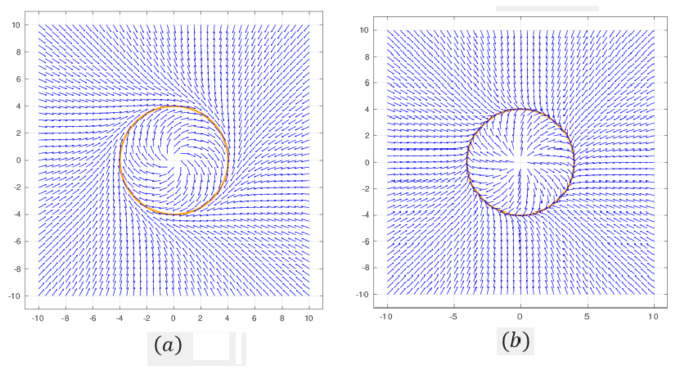


Fig. 4: (a) Vector field for a circle when $k_e = 1$ (b) Vector field for a circle when $k_e = 5$

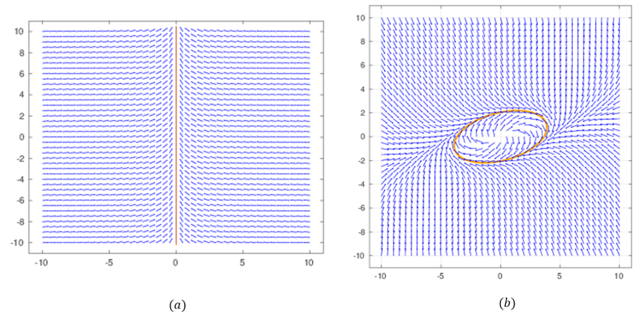


Fig. 5: (a) Vector field for a straight line (b) Vector field for an ellipse

As mentioned earlier, (9) describes the unit vector that points in the direction of the desired velocity. Thus, the desired velocity is given by

$$\dot{p}_d = s\hat{a}(p) \tag{10}$$

where s is a scalar value representing the desired airspeed of the UAV. This desired airspeed can then be used to calculate the desired acceleration, which consists of a feedback term and a feedforward term.

1) Feedback: The actual velocity of the UAV, represented as $\dot{p}$, can be obtained using a GPS module, or an indoor positioning system, while the desired velocity is given by (10). The expression for the feedback acceleration can then be obtained as follows

$$\ddot{p}_{fb} = k_s(\dot{p}_d - \dot{p}) \tag{11}$$

where k_s is the speed gain used by the outer INDI controller.

Consider Figure 6. The difference between the desired velocity from (10) and the velocity of the UAV is represented as the arrow on the right, which is multiplied by the INDI speed gain, which in this case is $k_s = 2$, and this is the feedback acceleration applied to the UAV.

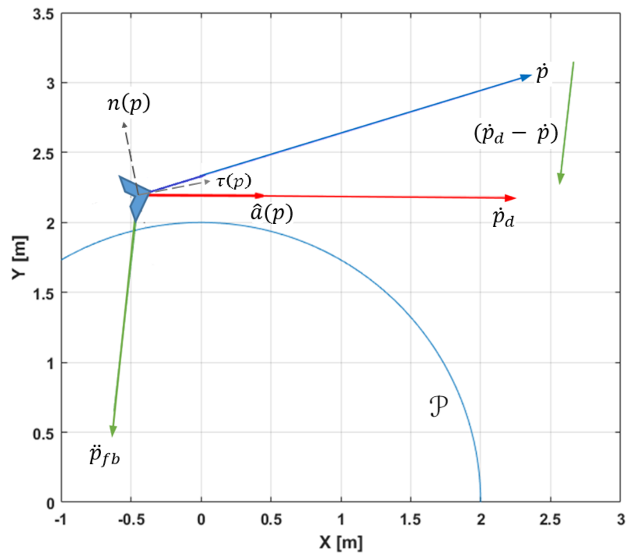


Fig. 6: Representation of the feedback acceleration

2) Feedforward: The desired velocity in (10) can also be simply differentiated, in order to obtain the feedforward acceleration

$$\ddot{p}_{ff} = \frac{d}{dt}(\dot{p}_d) = s\frac{d}{dt}(\hat{a}(p)) \tag{12}$$

The time derivative of the unit vector $\hat{a}(p)$ is derived from (9) to be

$$\begin{aligned}\frac{d}{dt}(\hat{a}(p)) &= (I - \hat{a}\hat{a}^T)\frac{\dot{a}(p)}{||a(p)||} \\ &= -E\hat{a}\hat{a}^T E\frac{\dot{a}(p)}{||a(p)||}\end{aligned} \tag{13}$$

where I is the identity matrix and $\dot{a}(p)$ is the time derivative of (8), obtained as follows

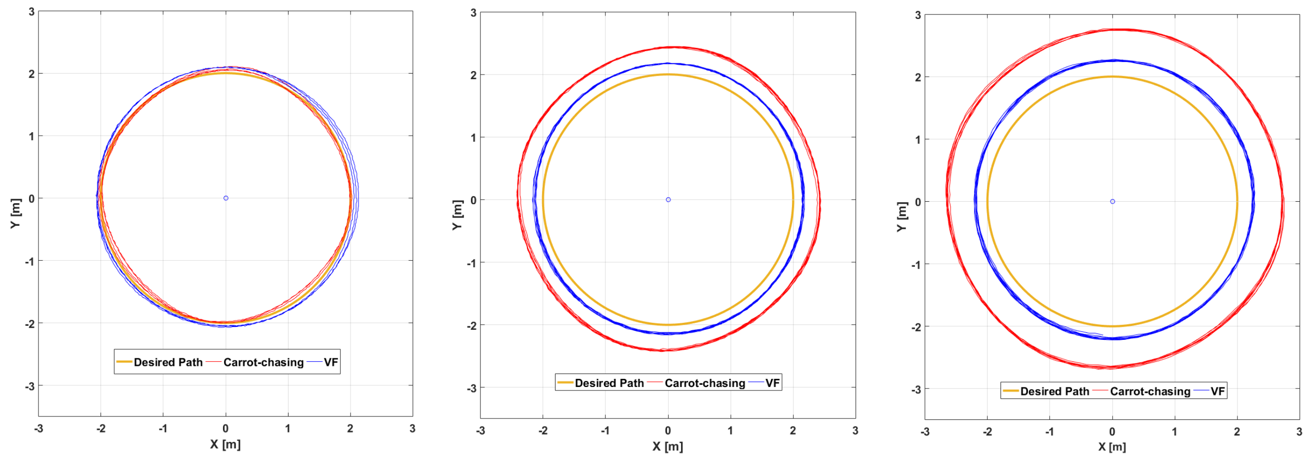
$$\begin{aligned}\frac{d}{dt}(a(p)) &= \frac{d}{dt}\Big(\tau(p) - k_e e(p)n(p)\Big) \\ &= \frac{d}{dt}\Big(En(p) - k_e e(p)n(p)\Big)\end{aligned}$$

$$\dot{a}(p) = (E - k_e e(p)I)H(\varphi(p))\dot{p} - k_e n^T\dot{p}n \tag{14}$$

where $H(\varphi(p))$ is the Hessian matrix of the path function, given by

$$H(\varphi(p)) = \begin{bmatrix} \frac{\partial^2\varphi(p)}{\partial x^2} & \frac{\partial^2\varphi(p)}{\partial x\partial y} \\ \frac{\partial^2\varphi(p)}{\partial x\partial y} & \frac{\partial^2\varphi(p)}{\partial y^2}\end{bmatrix}$$

The Hessian matrix is the rate of change of the gradient, and provides information about how the path changes over time.



(a) Velocity 1 m/s (b) Velocity 2 m/s (c) Velocity 3 m/s

Fig. 7: Position of the UAV for radius r = 2m

The final expression for the feedforward acceleration can be obtained by substituting (14) and (13) into (12). The total acceleration that is required to be tracked by the INDI controller is then

$$\ddot{p}_d = \ddot{p}_{fb} + \ddot{p}_{ff} \tag{15}$$

V. Implementation and Flight Tests

In this section, the performance of the Vector Field based path following controller is reported. The controller was implemented in the Paparazzi open-source autopilot system [11] and the performance of the controller was analyzed by the means of flight test experiments. The UAV platform used for the experiments was the Parrot Bebop quadrotor. The position of the drone was tracked with high accuracy using Optitrack [17], an indoor infrared position tracking system.

The goal of the experiments was to compare the performance of the vector field based path following algorithm with the standard carrot chasing algorithm that is currently used in the Paparazzi autopilot system. For this purpose, the UAV was commanded to track various paths, the tuning parameters for each experiment being the properties of the path, such as the shape and size, and the desired airspeed of the UAV.

A. Circular Path

The first set of experiments required the UAV to follow a circular path. The desired path $\mathcal{P}$ is described using the general equation of an ellipse

$$\varphi(p) = \left(\frac{(p_x - c_x)cos\alpha - (p_y - c_y)sin\alpha}{a}\right)^2 + \left(\frac{(p_x - c_x)sin\alpha + (p_y - c_y)cos\alpha}{b}\right)^2 - 1 \tag{16}$$

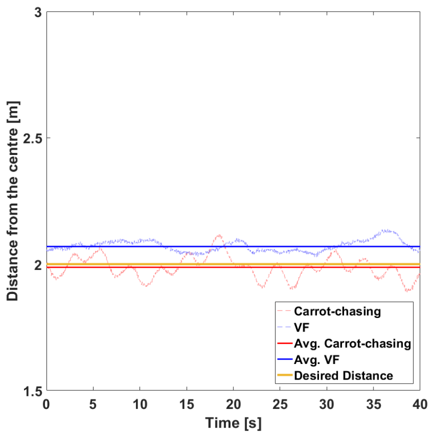
where $c = [c_x \ c_y]^T$ is the centre of the ellipse with respect to the inertial navigation frame, a and b are the lengths of the major and minor semi-axes of the ellipse respectively, and α is the rotation angle of the ellipse with respect to the horizontal axis of the frame. For the experiments, the values of a and b were always kept equal, thereby making the desired path a circle of radius $r = a = b$, and the value of was set to zero.

For the experiments, the value of the gain was set to $k_e = 1$, and the radii that were considered are $r = 1.5m$, $r = 2m$ and $r = 2.5m$. The dimensions were selected according to the space limitations of the indoor experimental setup. For each of the 3 radii, the UAV was commanded to fly at airspeeds of $1m/s$, $2m/s$ and $3m/s$, therefore making a total of 9 experiments per algorithm. In this paper, the tests and results shown are for the experiments corresponding to $r = 2m$, the results of which are shown in Figures 7-10. The results of all the other experiments can be found in Chapter 5 of the thesis.

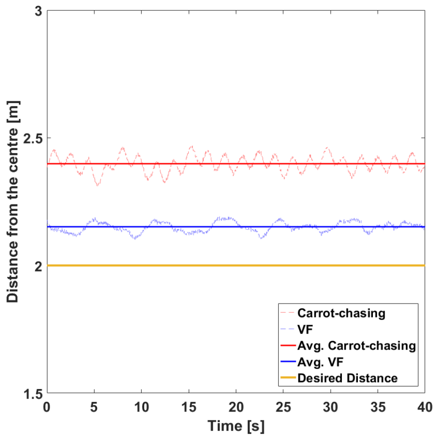
Figure 7 shows the logged position data of the UAV for each of the test conditions. The position when using the carrot chasing controller is represented in red, and the position when using the vector-field controller is represented in blue. The desired path of the UAV is represented in orange.

Figure 8 shows the distance of the UAV from the centre of the desired circle, where the cross track error can be better perceived. The orange line represents the desired distance from the centre of the circle. The blue line represents the average distance from the centre when using the vector field controller, over the course of the whole experiment, while the red represents the same when using the carrot chasing controller.

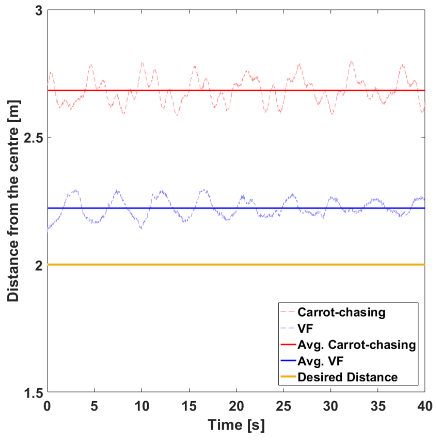
As can be seen from the figures, the performance of both the path following algorithms is very similar when at a low velocity of $1m/s$, and as the velocity increases, so does the steady state position tracking error, for both the path following algorithms. However, the rate of increase of the



(a) Velocity 1 m/s



(b) Velocity 2 m/s



(c) Velocity 3 m/s

Fig. 8: Distance from the centre for radius r = 2m

error is much higher for the carrot-chasing algorithm than for the VF algorithm, as is evident in Figure 9. The average position error when using the VF controller is considerably lesser than the traditional carrot-chasing controller, particularly as the velocity increases. The rate of increase of position error per unit increase in velocity is approximately $0.35m$ when using carrot chasing, and approximately $0.05m$ when using the vector field controller.

Also, by extrapolating the carrot-chasing data in Figure 9, it can be seen that on further decreasing the velocity, the position error will become negative, meaning the UAV will always fly inside the desired circle, and the position error will not be zero. This is one of the inherent issue with virtual target point based path following methods.

A possible explanation for the existence of a position error is due to the fact that instantaneous acceleration commands are calculated by the vector field controller, which are passed through the various control loops and finally the UAV receives instantaneous rotational velocity commands for the rotors. However, the rotors have some dynamics and take a finite amount of time before they achieve the desired rotational velocities. During the time that it takes for the rotors to reach the desired rotational velocity, the UAV travels forward, based on the command it last received, which causes a small steady state error. As the velocity of the UAV increases, it travels further in the time between two successive commands, and hence the error increases with an increase in velocity. This can be observed when comparing the x components of the commanded acceleration and the measured acceleration of the UAV in Figure 10 for the case of $v = 3m/s$. As can be seen, the actual acceleration of the UAV lags behind slightly from the commanded acceleration. This is because the algorithm calculates the instantaneous acceleration and commands it to the UAV, but the rotors on the UAV take a small amount of time to reach the desired rotational velocity.

The average delay of the actual acceleration, compared to

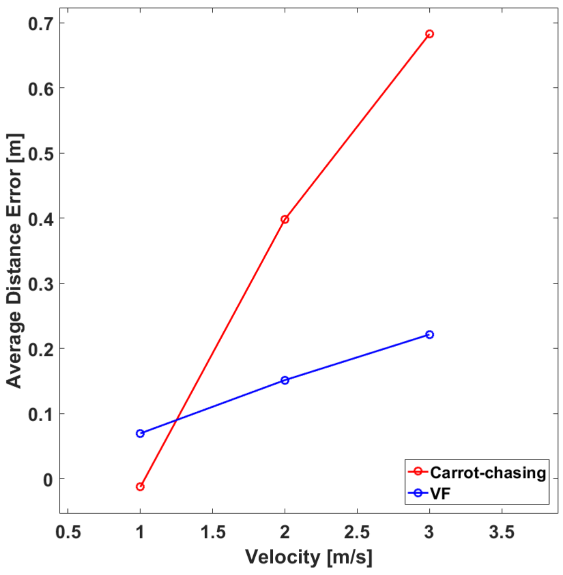


Fig. 9: Average Distance Errors

the commanded acceleration, is 32 samples. The sampling frequency is 512 Hz, and therefore, the average time delay is 0.0625 seconds. When the UAV is flying at a velocity of 3 m/s, the average position error can be calculated to be 0.1875 m. This matches the average position error as seen in Figure 9, which is about 0.2 m.

B. *Combination of Multiple Vector Fields for Complex Paths*

Some initial work has also been done in the direction of using vector field path following for complex paths, where it

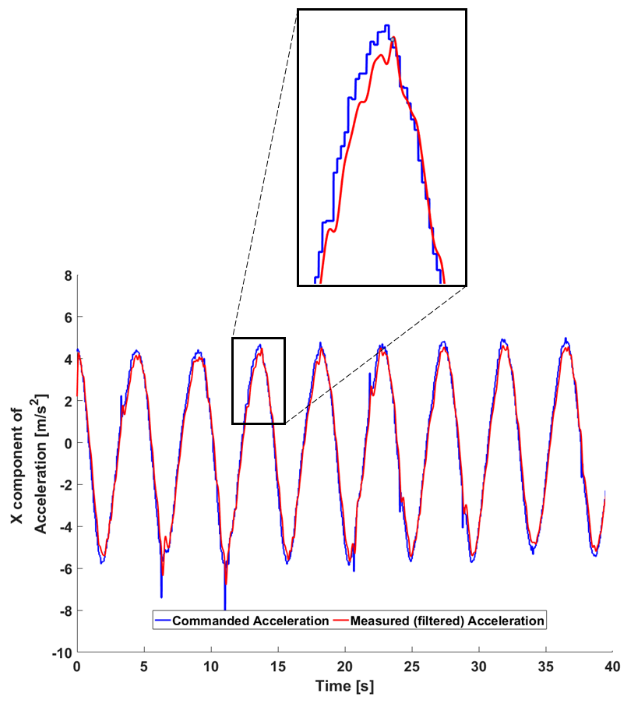


Fig. 10: Commanded and Measured Accelerations at $3m/s$

is difficult to generate a single vector field for the entire path. In such a case, the path can be split up into multiple simpler segments, and the vector field for each individual path segment can be generated.

For this, a second set of experiments was performed in which a figure eight shaped path was to be followed. The path was discretized using two partial circles, and two lines connecting them, as can be seen in Figure 11. The UAV first follows the straight line from $p1$ to $p2$, then the circular path from $p2$ to $p3$ around $c1$, then the straight line from $p3$ to $p4$ and finally the circular path from $p4$ to $p1$ around $c2$, after which the path starts again from the first segment. The radius of the two circular paths is $1m$, and the distance between their centres is $4m$.

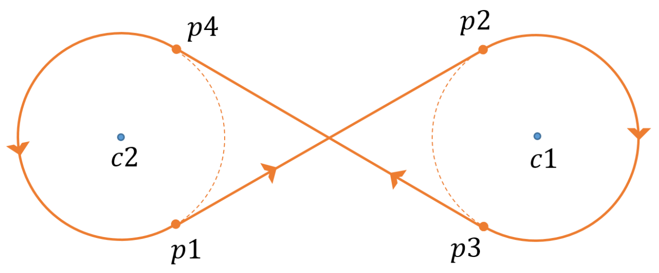


Fig. 11: A figure eight path as the combination of two circles and two straight lines

The four different vector fields, corresponding to the four path segments were combined using a basic switching algorithm. The UAV only receives commands from the vector field of the segment it is tracking. Once the UAV reaches the end of a path segment, the vector field switches to that of the next segment. This prevents any discontinuities or sinks that may arise from the combination of multiple vector fields.

The experiments were performed for velocities of $1m/s$ and $2m/s$, and the logged position data for both algorithms is shown in Figures 12 and 13 respectively. For both experiments, the value of the gain was set to $k_e = 1$ when tracking the circular segment, and $k_e = 5$ when tracking the line segment. As can be seen from these images, once again the path following performance when using the vector field based controller is better than when using the carrot chasing controller. In particular, when using the carrot chasing controller, there is a large overshoot when the UAV switches from the line path segment to the circle path segment, visible at points $p2$ and $p4$. The overshoot when using the vector field based controller is comparatively lesser. Also similar to the previous experiment, the position error increases with an increase in velocity, and this can be attributed primarily to the same reasons as discussed earlier.

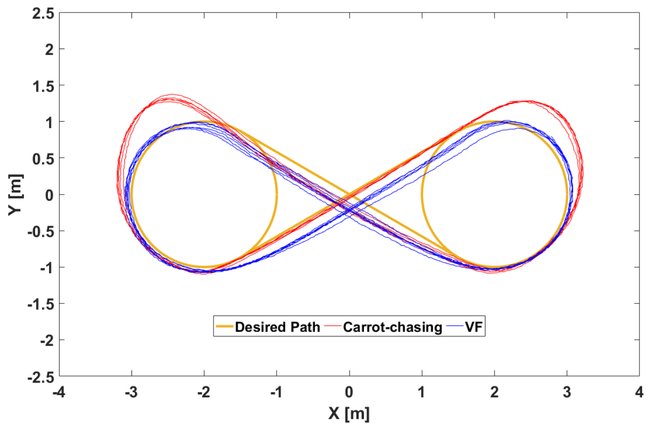


Fig. 12: Position of the UAV following a figure eight path at $1m/s$

VI. Conclusion

In this paper, a vector-field based path following approach for UAVs was successfully developed, which makes use of the existing Incremental Nonlinear Dynamic Inversion acceleration controller. The algorithm is implemented in the Paparazzi autopilot system, and experimental tests have shown an increase in path following performance when compared with the traditional carrot-chasing controller, as the UAV is able to follow the desired path more closely.

It was shown that complex paths can also be followed using this approach, by discretizing the complex path into multiple simple path segments. Experiments are performed where the UAV was required to follow a complex figure eight path, and

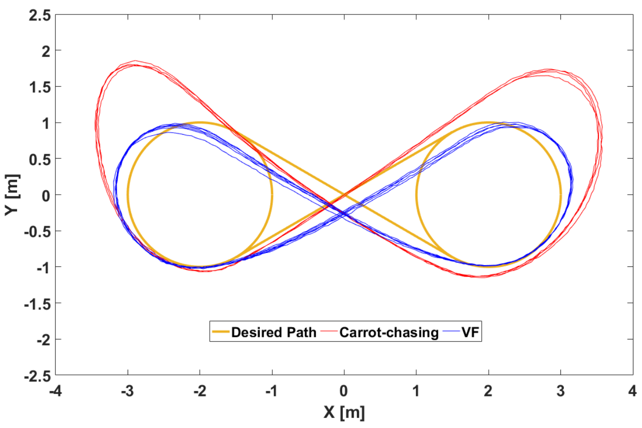


Fig. 13: Position of the UAV following a figure eight path at $2m/s$

the results are analyzed. Once again it was seen that the vector field based controller performs better with a lower tracking error.

This new path following method increases the ability of the UAV to accurately follow predefined paths, thereby making it more reliable for use in critical missions in urban environments or when flying near obstacles.

Future Work

To further improve the performance of the vector field based path following controller, the response time of the rotors can be taken into account when calculating the desired accelerations. By measuring the response curve of the rotors, the time required to achieve a desired increment in the rotational velocity can be calculated, say t_r. Since the velocity and position of the UAV can be measured, the position of the UAV after t_r seconds can be predicted. If this predicted position is used to calculate the position error, and therefore the desired acceleration, the performance of the controller could see an improvement.

In the current research, the complex figure eight path is split in such a way that each individual segment is either a straight line or a circle, since the functions describing such shapes are well known. However, this method of splitting the paths may cause some overshoot, particularly if there is an abrupt change in path angle at the point of connection of two path segments. A better approach might be to analyze the path and split it at points where no change in path angle occurs, such as at the middle of a straight line. Then, for each path segment a function that best approximates its shape can be generated, and the vector fields can be computed for each path segment using this function. This could possibly prevent overshoot, and enable the UAV to maintain a low tracking error all along the path.

References

[1] Michael A Goodrich, Bryan S Morse, Damon Gerhardt, Joseph L Cooper, Morgan Quigley, Julie A Adams, and Curtis Humphrey. Supporting wilderness search and rescue using a camera-equipped mini uav. *Journal of Field Robotics*, 25(1-2):89–110, 2008.

[2] Randal W Beard, Timothy W McLain, Derek B Nelson, Derek Kingston, and David Johanson. Decentralized cooperative aerial surveillance using fixed-wing miniature uavs. *Proceedings of the IEEE*, 94(7):1306–1324, 2006.

[3] PB Sujit, Srikanth Saripalli, and Joao Borges Sousa. Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicles. *IEEE Control Systems*, 34(1):42–59, 2014.

[4] Claude Samson. Path following and time-varying feedback stabilization of a wheeled mobile robot. In *Proceedings of the international conference on advanced robotics and computer vision*, volume 13, pages 1–1, 1992.

[5] Unghui Lee, Sangyol Yoon, HyunChul Shim, Pascal Vasseur, and Cedric Demonceaux. Local path planning in a complex environment for self-driving car. In *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2014 IEEE 4th Annual International Conference on*, pages 445–450. IEEE, 2014.

[6] Ehsan Peymani and Thor I Fossen. Path following of underwater robots using lagrange multipliers. *Robotics and Autonomous Systems*, 67:44–52, 2015.

[7] Isaac Kaminer, Antonio Pascoal, Eric Hallberg, and Carlos Silvestre. Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control. *Journal of Guidance, Control, and Dynamics*, 21(1):29–38, 1998.

[8] Wei Ren and Randy W Beard. Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints. *IEEE Transactions on Control Systems Technology*, 12(5):706–716, 2004.

[9] Sanghyuk Park, John Deyst, and Jonathan P How. A new nonlinear guidance logic for trajectory tracking. In *AIAA guidance, navigation, and control conference and exhibit*, pages 16–19, 2004.

[10] Paul Zarchan. *Tactical and strategic missile guidance*. American Institute of Aeronautics and Astronautics, 2012.

[11] Paparazzi uav. http://wiki.paparazziuav.org/wiki/Main$_Page. [Online]$.

[12] Sanghyuk Park, John Deyst, and Jonathan P How. Performance and lyapunov stability of a nonlinear path-following guidance method. *Journal of Guidance Control and Dynamics*, 30(6):1718–1728, 2007.

[13] Yuri A Kapitanyuk, Anton V Proskurnikov, and Ming Cao. A guiding vector-field algorithm for path-following control of nonholonomic mobile robots. *IEEE Transactions on Control Systems Technology*, 2017.

[14] Hector Garcia de Marina, Yuri A Kapitanyuk, Murat Bronz, Gautier Hattenberger, and Ming Cao. Guidance algorithm for smooth trajectory tracking of a fixed wing uav flying in wind flows. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5740–5745. IEEE, 2017.

[15] Ewoud JJ Smeur, Qiping Chu, and Guido CHE de Croon. Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *Journal of Guidance, Control, and Dynamics*, 38(12):450–461, 2015.

[16] Ewoud JJ Smeur, Guido CHE de Croon, and Qiping Chu. Cascaded incremental nonlinear dynamic inversion control for mav disturbance rejection. *arXiv preprint arXiv:1701.07254*, 2017.

[17] Optitrack - motion capture systems. http://optitrack.com/. [Online].

Part II

Thesis

Chapter 1

Introduction

Unmanned Aerial Vehicles (UAVs) have proven to be useful in many military and commercial applications, such as mapping, surveillance, precision agriculture, disaster relief and search and rescue [1, 2]. In recent years, their use in cities and complex urban environments has also seen an increase due to applications such as item delivery and traffic monitoring. The ability of the UAV to accurately follow a predefined path is of utmost importance to ensure safe operation in cities, and most applications depend on this ability [3, 4]. The path following algorithm used must be robust to external disturbances such as wind, and must not be computationally complex, so as to run in real time on-board the UAV. An accurate path-following algorithm is of importance not just to UAVs, but also to ground-based robots, self-driving cars [5, 6], and Unmanned Underwater vehicles (UUVs) [7].

Most complex missions can be approximated as a combination of two simple paths: straight line segments connected by two waypoints (a starting waypoint and an ending waypoint), and circular segments knows as loiters [8]. A number of algorithms exist that ensure the UAV can follow a 3 dimensional path, or a 2 dimensional path at a fixed altitude, and they can be broadly divided into 2 categories: trajectory-tracking and path-following.

In trajectory-tracking, the UAV is instructed to be at a specific waypoint at the specified time, that is, the trajectory is time parameterised. This is therefore the technique used in the case of missile guidance control [9, 10, 11]. In the case of path-following, the goal is to drive the "cross-track error" to zero, where the cross-track error is the difference between the UAV's position and the desired flight path. Hence, the path is not a function of time, and this is the reason it has been chosen as the approach during this research. Moreover, analysis and comparisons of path following and trajectory tracking is provided in [12] and [13], which shows that trajectory tracking has a performance limitation due to unstable zero-dynamics, and such issues can be avoided by using path following.

An increasingly common technique for path following is the use of vector fields. Vector fields are spatial functions that assign a vector to each point in space. In the context of UAV flight control, each point in either 3D or 2D space would be associated with a particular velocity vector, which is commanded to the UAV.

The goal of this literature study is to understand the limitations that exist with the currently widely used methods of path-following, and explore the possibilities in the use of vector fields for improved path following.

This report is structured as follows. In Chapter 2, a concise study of important literature is provided, which forms the theoretical background of this research. A number of different approaches to the path-following problem are reviewed. Following this, in Chapter 3, a brief summary is provided of the main reference literature adopted for this research, which is the Guidance Vector Field based method as applied to fixed wing UAVs. The INDI based acceleration controller that the vector field based controller sends commands to is reviewed in Chapter 4. All the experiments performed and their results are presented in Chapter 5 along with a short discussion of the results. Finally, the conclusions and some recommendations for possible future research are provided in Chapter 6.

Chapter 2

Literature Survey

Extensive research has already been done in the field of mobile-robot path following. This section provides an overview of the scientific literature relevant to this field of research. The various approaches to path-following and the use of vector fields are examined in detail in each of the following sub-sections. This overview will help in mapping the current state-of-the-art and analysing what improvements are yet to be made, thereby helping in formulating the research questions to answer.

2.1 Virtual Target Point based Methods

A technique that is currently widely used for path following involves the use of a ”virtual target point” (VTP) which lies on the path, at a short distance ahead of the vehicle, known as the ”look-ahead distance”. Examples for path-following techniques that employ the use of a VTP are the carrot-chasing algorithm, which is the algorithm used in the Paparazzi UAV autopilot [14], and the Nonlinear Guidance Law [15]. While straightforward to understand and easy to implement, such methods are limited in their nature. This is mainly due to the fact that following the target point does not guarantee accurate path following, and the performance of the path following method depends on the way in which the VTP is calculated. Other issues may exist with VTP based path following such as difficulty in calculating the distance to the path when an optimization problem has to be solved, inability to obtain a unique target point on the path when the path consists of self-intersections, or the inability to find a target point at all, as is the case with the Nonlinear Guidance Law method [15] when the UAV is further than the fixed radius used to obtain the VTP. Due to these disadvantages, more robust path following algorithms are preferred, as discussed in the following sections.

2.2 LQR based Path-Following

Another method to solve the path following problem is to consider it as an infinite horizon regulator problem, which allows using the Linear Quadratic Regulator (LQR) technique to derive an optimal guidance law [17]. This optimal guidance law uses an adaptive state weighting matrix, which is a function of the current and maximum allowable position errors of the UAV. This adaptive property enables precise control of the UAV in the presence of wind disturbances, as is shown in their work using numerical simulations for both straight lines and circular paths under multiple wind conditions.

An integrated approach to path generation and following, implemented with linear-quadratic regulator control laws is presented in [18]. The method proposed is capable of representing complex paths between given waypoints in an efficient manner by using just a few parameters, and low computational complexity, thereby making the technique suitable for real-time on-board use. The weighting matrices are tuned using a series of simulations, and are non-adaptive in nature, adopted from [19]. It is shown that when using an appropriate gain matrix, and when the heading and cross-track errors are not too large, good performance results are obtained. However, in the case of large errors, inputs generated are large, which causes instability. In order to avoid instability, heading and cross-track errors are saturated. Experimental flight tests were performed using a fixed wing UAV, and results show cross-track errors to be under 10m when subjected to moderate wind conditions. On long straight line paths, where the heading of the UAV is not required to change often, the errors converge to under 2m, which is less than 1 wingspan.

2.3 Model Predictive Control based Path-Following

Model Predictive Control (MPC) is an advanced control method that originated in the Process Control Industry. MPC uses the model of the plant to be controlled, to predict the output of the plant at future time instants, known as the horizon. At each time step, it performs an optimization procedure to obtain a suitable control sequence. The main advantages of MPC are that it is suitable for Multi-Input Multi-Output systems, and it inherently handles constraints in the inputs, states and outputs, thereby guaranteeing stability of the controlled system. In the past, MPC was only used in the process industry, as the processes are slow and do not require very fast control inputs. This was suitable for MPC, as when constraints are considered, the computational time is very high. However, in recent years, due to increased computational power, and the development of faster optimization techniques, MPC is increasingly used in the robotics industry.

Nonlinear Model Predictive Controller (NMPC), a variant of MPC which uses nonlinear system models, has been used to solve the path-following problem for nonlinear systems, while maintaining stability [20]. By using the path parameter as a part of the state variable, a well performing controller is obtained. Both input and state constraints are inherently considered and state-feedback is taken into account, in contrast to some previous work [13, 21] where input constraints were not considered and only output-feedback was used. Results were demonstrated using simulations of an autonomous vehicle.

A concept within MPC, known as Receding Horizon Control is used to develop a tracking controller for wheeled mobile robots in [22]. Although stability was achieved and accurate tracking was performed, the method developed required an initial feasible solution. Also, the results were only demonstrated through simulations due to high computational load. This computational problem was overcome in [23] by making use of an MPC optimization law based on a linearized error dynamics model. This enables use of the controller on real time systems, as is shown by implementation on an actual omnidirectional robot. However, a disadvantage of using a linearized model is that stability is only guaranteed around the trim points. Another proposed method to prevent excessive computational times is the use fast feedback algorithm in MPC [24].

The NMPC concept has also been tested to control a UAV, albeit in simulations [25]. The problem of selection of the length of the control horizon was previously dealt with by simulating multiple values, and selecting the best fit. It was treated as a tuning parameter, which once set was used throughout the flight regime. However, in this work, an adaptive NMPC solution is proposed, where the control horizon varies based on the curvature of the path. This provides more accurate tracking of paths, as shown by simulation results, when compared with fixed-horizon NMPC techniques.

2.4 Vector Field based Methods

A Vector Field function is computed in a way that its integral curves converge to the desired path asymptotically. The robot or UAV that is to follow the vector field, is steered along the integral curves until it eventually follows the path. The efficiency of VF techniques have been proven in [26] and [27].

The use of vector fields for path-following has seen an increase in popularity due to many inherent advantages. For instance, for a known path, the vector field can be calculated completely offline, prior to the flight, which reduces the amount of online calculations required. A few prominent vector field strategies are discussed below.

An accurate path following method for miniature UAVs based on vector fields is developed in [27]. The method first calculates a vector field surrounding the desired path that is to be followed by the UAV. Each vector in the field is directed towards the path, and the magnitude of the vector depends on its distance from the path. The vector values are provided as course command inputs to the inner loop attitude controllers in the UAV. By representing the equations of motion in terms of groundspeed and course angle, instead of airspeed and heading angle, the authors show that disturbances due to wind are significantly reduced. Lyapuov stability criteria were used to show that the path errors decay asymptotically, even in the presence of wind. Experimental flight tests were performed using a fixed-wing UAV and it was shown that the average errors during path following were less than one wingspan.

In the work in [27], when straight line segments and circular arcs are combined to form a complex path, the UAV is only under the influence of the one vector field at a time, which is the current path segment that it is following. When the UAV reaches the end of a segment, the vector field changes to the one corresponding to the next segment. This is done in order to prevent issues that may arise when combining various vector fields, such as multiple sinks, dead-zones and singularities. In contrast, [28] developed a technique that uses Lyapunov stability properties to generate globally stable vector fields to track circular arcs in combination with waypoints. This is achieved by the use of switching algorithms, and by warping the circular arc to produce other shapes.

The vector field methods described are based on a previous method of path following for mobile robots using potential fields. In [29], a real time obstacle avoidance and path following scheme is introduced, that utilises the concept of artificial potential fields. The mobile robots moves in a field of forces that can influence the position of the robot. The desired goal positions act as attractive forces, and the obstacles act as repulsive forces. Although the method developed in [29] is implemented on a robot-arm system, potential fields have also been used for UAV navigation. The gradient of potential fields has been used in [30] to enable a swarm of UAVs to navigate in the vicinity of each other. In this approach, each UAV generates a magnetic field around itself, and is able to estimate the gradient of the total field generated by the entire network, which it uses to move in the opposite direction to avoid collision.

Chapter 3

Guidance Vector Field Path-Following for Fixed Wing UAVs

In this chapter, a brief summary of the main reference literature is provided. The primary work is [31], a recent publication that provided a method of generating a vector-field based path following control law that enables a UAV, or generally any non-holonomic robot, to follow a general smooth planar path of any shape.

A vehicle or robot that is able to instantaneously move in any direction, irrespective of its angular orientation is a holonomic robot. Examples of such holonomic vehicles are helicopters and robots that use omni-directional wheels.

In practical applications however, non-holonomic vehicles are more commonly encountered. A non-holonomic robot is one that can only move is some directions. A unicycle is an example of a simple non-holonomic vehicle, as it cannot move in the lateral direction but only in the longitudinal direction. UAVs and fixed-wing aircraft can be simplified to be represented as unicycles under certain conditions. Moreover, fixed-wing UAVs also have an additional restriction in that their velocity cannot be reduced below a certain minimum, as it would cause the aircraft to stall.

The work in [31] presents a method of generating vector-field algorithms for a non-holonomic robot along a general smooth planar path. Convergence to the path can be proved for any trajectory that is free of critical points, that is, points where the vector field is degenerate.

3.1 Problem Definition

3.1.1 UAV Kinematic Model

The kinematic model of the UAV is defined as follows:

$$\dot{\bar{r}} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = u_r \bar{m}(\alpha) \tag{3.1}$$

$$\dot{\alpha} = \omega \tag{3.2}$$

where, $\bar{r} \in \mathbb{R}^2$ is the position of the center of gravity of the UAV in an inertial Cartesian reference frame, u_r is a positive constant representing the longitudinal velocity, α is the orientation of the UAV in the reference frame, and $\bar{m}(\alpha)$ is the unit orientation vector given be $\bar{m}(\alpha) = [cos(\alpha) \;\; sin(\alpha)]^T$ (See Figure 3.1). ω is the angular velocity, and is the only control input available, thereby making the UAV an underactuated system.

3.1.2 Description of the path

Consider the plane $\mathbb{R}^2$ covered by disjointed level sets of the function $\varphi(p) \in C^2(\mathbb{R}^2 \to \mathbb{R})$, where $\varphi(p) = c = const$. Then, the desired curvilinear path to be followed, $\mathcal{P}$ is defined as the zero set of the function φ.

The path is described by the implicit equation:

$$\mathcal{P} \triangleq \{(x, y) : \varphi(x, y) = 0\} \in \mathbb{R}^2 \tag{3.3}$$

.

The restriction imposed on the path $\mathcal{P}$ is that of *regularity*, which dictates that in the region around $\mathcal{P}$, the gradient of $\varphi(p)$ must exist.

$$\bar{n}(x, y) \triangleq \nabla\varphi(x, y) = \left[\frac{\partial\varphi(x, y)}{\partial x}; \frac{\partial\varphi(x, y)}{\partial y}\right]^T \neq 0. \tag{3.4}$$

If the above stated condition for regularity holds, then the vector $\bar{n}(x, y)$ is the normal vector at the point (x, y).

The value of the function $\varphi(x, y)$ serves as the signed distance error of the UAV from the desired path, known as the *tracking error.* As mentioned earlier, the path $\mathcal{P}$ itself is also one of the level sets corresponding to $\varphi(x, y) = c = 0$. This signifies that the tracking error when the UAV is on the path is zero, which makes sense intuitively.

Formally, the tracking error is defined as follows:

$$e(x, y) \triangleq \psi[\varphi(x, y)] \in \mathbb{R} \tag{3.5}$$

where, ψ is a chosen as an arbitrary strictly increasing function $\psi \in C^1(\mathbb{R} \to \mathbb{R})$ with $\psi(0) = 0$. Therefore, when $(x, y) \in \mathcal{P}$, the value of the function $\varphi(x, y) = 0$, and so the tracking error $e = \psi(0) = 0$.

The function definition of the path-defining function $\varphi(x, y)$ depends on the path to be followed. For example, if a circular path is to be followed, then $\varphi(x, y) = (x - x_0)^2 + (y - y_0)^2$. Some examples of the different possible function definitions of the error-defining function ψ can be found in [31].

3.1.3 Technical Assumptions

The following technical assumptions are adopted:

- The tracking error $e(x, y)$ can be used as the signed distance to the path $\mathcal{P}$, and the asymptotic vanishing of the error $e(\bar{r}(t)) \xrightarrow[t\to\infty]{} 0$ implies convergence to the path $dist(\bar{r}(t), \mathcal{P}) \to 0$.
- Let C_0 be the set of critical points, that is, points where the gradient $\nabla\varphi(x, y)$ doesn't exist, therefore $\bar{n}$ vanishes.

$$C_0 \triangleq \{(x, y) \in \mathbb{R}^2 : \nabla\varphi(x, y) = 0\} \tag{3.6}$$

  The second assumption is that this set of critical points is separated from the path $\mathcal{P}$ by a positive distance $dist(C_0, \mathcal{P}) > 0$. This assumption can also be stated as, the desired path $\mathcal{P}$ does not contain any critical points, that is, $C_0 \cap \mathcal{P} = \emptyset$.
- The asymptotic vanishing of the normal vector $\bar{n}(\bar{r}(t)) \xrightarrow[t\to\infty]{} 0$ is only possible along a trajectory that converges to C_0, that is $dist(\bar{r}(t), C_0) \to 0$.

3.2 Properties of the Guidance Vector Field

- At each point on the plane $\mathbb{R}^2$, the normal vector and the tangent vector to the level set are considered.
- The normal vector is given by the gradient, $\bar{n}(x, y) = \nabla\varphi(x, y)$.
- The tangent vector is given by

$$\bar{\tau}(x, y) = E\bar{n}(x, y), \qquad E = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{3.7}$$

- The goal is to compute a vector field $\bar{v}(x, y)$ such that the tracking error e decreases along its integral curves (unless e = 0), and the curves that originate on $\mathcal{P}$ remain on the path.

  The following vector field is defined:

$$\bar{v}(x, y) \triangleq \bar{\tau}(x, y) - k_n e(x, y)\bar{n}(x, y). \tag{3.8}$$

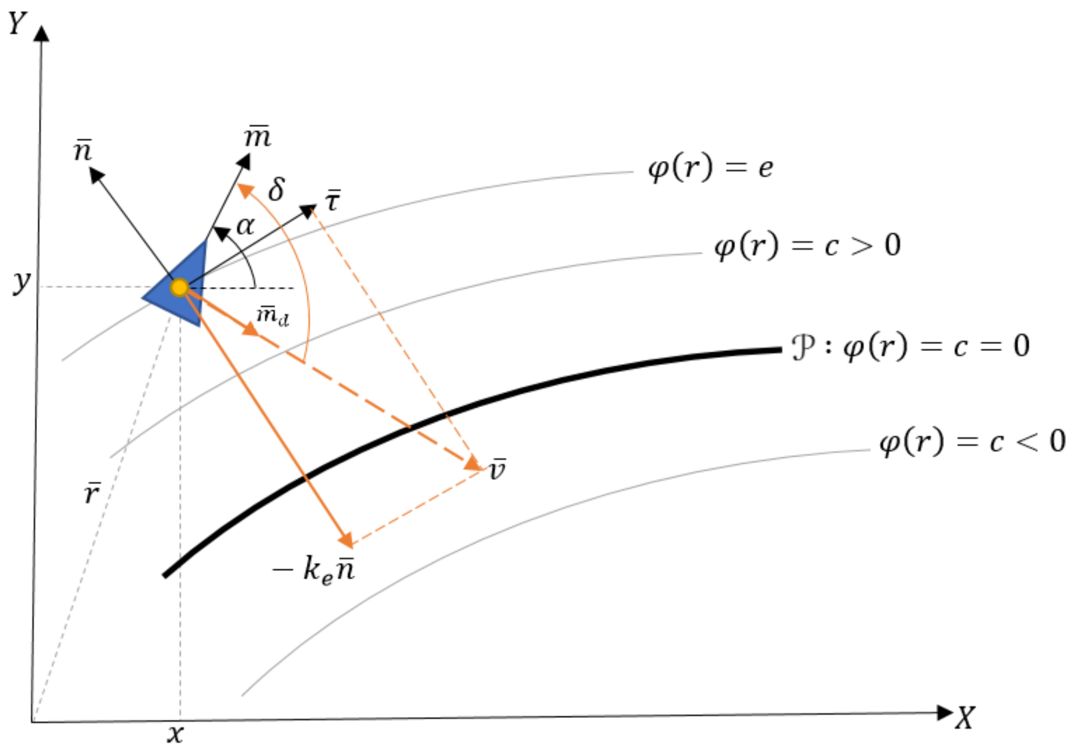


Figure 3.1: Representation of the UAV and the level sets.

The visual representation of $\bar{v}$ for a particular point $\bar{r} = (x, y)$ can be seen in Figure 3.1.

In [31], the property of "dichotomy" of the integral curves of the proposed vector field is discussed. It is shown that the integral curves of the vector field either lead to the desired path $\mathcal{P}$ or to the critical set C_0. Furthermore, certain criteria are proposed which ensures that the integral curves that lead to C_0 are either absent, or cover a set of zero measure on $\mathbb{R}^2$.

When the UAV is at a point $\bar{r} = (x, y)$ in the vector field, its desired orientation can be obtained by:

$$\bar{m}_d(x, y) = \frac{1}{|\bar{v}(x, y)|}\bar{v}(x, y). \tag{3.9}$$

where $\bar{m}_d(x, y)$ is the unit vector in the desired orientation. The field of unit vectors is defined as the GVF, and it is defined at all points where $\bar{n} \neq 0$, that is, all regular points.

Consider the UAV ideally oriented along the integral curves of the vector field at any point. Its position vector $\bar{r}(t)$ then obeys the following differential equation:

$$\dot{\bar{r}} = u_r \bar{m}_d(\bar{r}(t)), \qquad t \geq 0 \tag{3.10}$$

3.3 Path Following Controller

A path following controller is designed that steers the UAV to the GVF. The GVF at the point (x, y) is defined as $\bar{m}_d(t) = \bar{m}_d(x(t), y(t))$. Then, its derivative is given by:

$$\dot{\bar{m}}_d(t) = -\omega_d(x(t), y(t), \alpha(t))E\bar{m}_d(t) \tag{3.11}$$

Here, $\omega_d(x, y, \alpha)$ is a function that measures the GVF's "rotation rate" along the trajectory of the UAV. In general, it can be considered as the "desired" curvature of the path of the UAV, therefore when the UAV is moving strictly along the integral curves of the GVF, then ω_d is the signed curvature of the UAV's trajectory at it's current position.

The angle between the UAV's current orientation $\bar{m}$ and the desired orientation $\bar{m}_d$ is called the directed angle, represented by $\delta = \delta(x, y, \alpha) \in (-\pi; \pi]$. Then, the relationship between δ, ω and ω_d is:

$$\dot{\delta} = \omega - \omega_d \tag{3.12}$$

Using the GVF rotation rate and the directed angle, the path following algorithm can be described as follows:

$$\boxed{\omega(t) = \omega_d(x(t), y(t), \alpha(t)) - k_\delta \delta(x(t), y(t), \alpha(t))} \tag{3.13}$$

where, k_δ is a positive constant that determines the rate of convergence. Following from 3.12 and 3.13, we have,

$$\boxed{\dot{\delta} = \omega - \omega_d = k_\delta \delta} \tag{3.14}$$

$\dot{\delta}$ is considered as the new control input, and acts as a simple proportional controller, provided $\delta(x(t), y(t), \alpha(t)) < \pi \forall t > 0$.

The full derivation of the control law and the proof of existence of solutions can be found in [31].

3.4 Extensions of GVF

The work presented in [31] can be extended in a number of direcitons.

- The GVF algorithm can be extended for use in 3D case. Some earlier 3D vector field based methods are presented in [28] and [32]. In the 3D case, the desired path $\mathcal{P}$ can be described as the intersection of two surfaces, and is defined as follows:

$$\mathcal{P} \triangleq \{(x, y, z) : \varphi_1(x, y, z) = 0 \wedge \varphi_2(x, y, z) = 0\} \subset \mathbb{R}^3 \tag{3.15}$$

where, $\varphi_i \in C^2(\mathbb{R}^3 \to \mathbb{R}), i = (1,2)$. The GVF is then defined as:

$$\bar{v} = \bar{\tau} - k_{n1}e_1\bar{n}_1 - k_{n2}e_2\bar{n}_2, \quad \bar{\tau} = \bar{n_1}x\bar{n_2} \tag{3.16}$$

where $\bar{n_i} = \nabla\varphi_i$ and $e_i = \varphi(x,y,z), i = (1,2)$.

- The results can also be extended to follow time-varying paths, where the predefined trajectory may be translated, scaled or rotated. Similarly, it can therefore also be used for standoff tracking of slow moving targets, as demonstrated in [33].
- When considering implementation of the GVF algorithm on an actual fixed-wing UAV, some practical issues such as the presence of wind disturbances must be considered as well. Work in this direction has been shown in [34], and will be summarized in the next section.

3.5 Practical Implementation

Following closely the work of [31], Hector et al. present an algorithm for path following of smooth curves by a fixed wing UAV while travelling with a constant speed and under the influence of a constant wind disturbance [34]. The work is summarized in this section, as it provides practically important improvements to the initial work such as considering the presence of wind disturbances, and also demonstrates the use of the path following algorithm on an actual fixed wing UAV.

3.5.1 Incorporation of Wind Disturbance

Since this approach also considers the presence of wind disturbance, the model of the UAV looks slightly different than the model discussed earlier. The non-holonomic 2D model including wind is as follows:

$$\dot{p} = sm(\psi) + w \tag{3.17}$$

$$\dot{\psi} = u \tag{3.18}$$

where, $p \in \mathbb{R}^2$ is the position of the UAV with respect to the inertial navigation frame, $s \in \mathbb{R}^+$ is a positive constant representing the airspeed, $\psi \in (-\pi, \pi]$ is the yaw angle, and $m = [cos(\psi) \;\; sin(\psi)]^T$. The constant wind is represented by $w \in \mathbb{R}^2$, and u is the control input to change the heading of the UAV.

Due to the presence of wind, apart from the yaw angle ψ, which is the direction that the UAV is pointing towards, there is also a course angle $\chi \in (-\pi, \pi]$ which is the direction that the velocity vector $\dot{p}$ points at.

Similar to [31], the path is represented using level sets, where the level set with a value of zero corresponds to the path itself. The value of the level set at the position of the UAV is considered as the signed distance to the path, or the tracking error, $e(p)$.

The guidance controller is designed in two parts. The first part is is to steer the UAV such that it aligns itself with the GVF. The other part is the construction of a guidance vector field such that once a UAV is tracking it, it will converge to the desired path.

The desired guidance vector field to be followed is given by:

$$\dot{p}_d := \tau(p) - k_e e(p) n(p) \tag{3.19}$$

where $k_e \in \mathbb{R}^+$ is a gain that determines how aggressive the control action by the vector field is.

The goal is to find an expression for the desired course heading rate $\dot{\chi}_d(\dot{p}, p)$, once the UAV is correctly tracking the GVF, that is, once the set $\mathcal{G} := \{\dot{p} : \dot{p} = \dot{p}_d\}$ is invariant. The desired acceleration is obtained as:

$$\begin{aligned}\ddot{p}_d &= \frac{d}{dt}(E - k_e e)n \\ &= (E - k_e e)H(\varphi(p))\dot{p} - k_e n^T \dot{p} n\end{aligned} \tag{3.20}$$

where, $H(.)$ is the Hessian operator, therefore implying that the function $\varphi(p)$ is C^2, meaning that the first and second derivative of the function both exist and are continuous. The use of the Hessian means that the UAV has a knowledge of the evolution of the curvature of the desired path $\mathcal{P}$.

Define $\hat{x} = \frac{x}{||x||}$, where the superscript $\hat{}$ is used to represent a unit vector. It is then shown that the desired course heading rate $\dot{\chi}_d$ can be obtained using:

$$\dot{\chi}(p, \dot{p}) = -\left(E\hat{\dot{p}}_d \hat{\dot{p}}_d^T E\Big((E - k_e e)H\big(\varphi(p)\big)\dot{p} - k_e n^T \dot{p} n\Big)\right)^T E\frac{\dot{p}_d}{||\dot{p}_d||^2} \tag{3.21}$$

Thus, it can be seen that the desired course heading depends only the velocity and position, which can be obtained, for example, from a GPS system on the UAV.

Finally, the control action that ensures the UAV converges to the vector field is:

$$\boxed{u(p, \dot{p}, \psi) = \dot{\psi} = \frac{||\dot{p}||}{s cos\beta}\big(\dot{\chi}_d(p, \dot{p}) + k_d \hat{\dot{p}}^T E \hat{\dot{p}}_d\big)} \tag{3.22}$$

where, $\beta = arccos(\hat{\dot{p}}^T m(\psi))$ is the sideslip angle, and $k_d \in \mathbb{R}^+$ is a gain that determines how quickly the UAV converges to the GVF. The proof of convergence can be found in [34].

3.5.2 Implementation on a UAV

The guidance vector field was implemented on a fixed wing UAV using the open-source autopilot Paparazzi [14]. The physical constraint on the fixed-wing UAV is that of a maximum bank angle, ϕ^*. Therefore, the two gains, namely, k_e in the GVF equation 3.19 and k_d in the control action 3.22 must be tuned such that the UAV satisfies this physical constraint.

Assume the UAV maintains a constant altitude, it's airspeed is greater than the norm of the wind velocity, $s >> ||w||$, then the condition for a coordinated turn is:

$$\dot{\psi} = \frac{gtan\phi cos\theta}{s} \tag{3.23}$$

where g is the acceleration due to gravity, $\dot{\psi}$ is the yaw rate, and ϕ and θ are the roll and pitch angles respectively. Therefore, the constraint to be satisfied is:

$$|\phi^*| \leq arctan\frac{su(p, \dot{p}, \psi)}{gcos\theta} \tag{3.24}$$

Thus, the path $\mathcal{P}$ must be designed such that 3.24 is always satisfied.

Chapter 4

Incremental Nonlinear Dynamic Inversion Acceleration Controller

The vector field based controller developed in this research is capable of controlling both, the position and velocity of the UAV. It does this by calculating appropriate acceleration commands, which it then sends to an acceleration control loop as a reference signal.

The acceleration controller used in this research is an Incremental Nonlinear Dynamic Inversion (INDI) acceleration controller developed in [35]. For the sake of completeness, a brief overview of the functioning of this INDI controller is provided in this chapter, in order to understand how the desired acceleration commands generated by the vector field controller are handled.

For a UAV with position p, velocity $\dot{p}$ and acceleration $\ddot{p}$ in the North East Down (NED) frame, the acceleration of the UAV in the NED frame can be obtained from Newton's second law of motion as

$$\ddot{p} = g + \frac{1}{m}F(\dot{p}, w) + \frac{1}{m}T_N(\eta, T) \tag{4.1}$$

where g is the acceleration due to gravity, m is the mass, F is the total aerodynamic force acting on the UAV, as a function of the velocity $\dot{p}$ of the UAV and the wind vector w, and T_N is the thrust vector in the NED frame, as a function of the attitude of the UAV, $\eta = [\phi \quad \theta \quad \psi]^T$ and the total thrust, T.

The thrust in the NED frame T_N is obtained by the equation

$$\begin{aligned} T_N(\eta) &= R_{NB}(\eta)T_B \\ &= \begin{bmatrix} (sin\phi sin\psi + cos\phi cos\psi sin\theta)T \\ (cos\phi sin\psi sin\theta - cos\psi sin\phi)T \\ (cos\phi cos\theta)T \end{bmatrix} \end{aligned} \tag{4.2}$$

where $R_{NB}(\eta)$ is the rotation matrix, and T_B is the thrust in the body frame, $T_B = [0, 0, T]^T$.

The first order Taylor expansion is applied to (4.1), which gives us

$$\begin{aligned}\ddot{p} &= g + \frac{1}{m}F(\dot{p}_0, w_0) + \frac{1}{m}T_N(\eta_0, T_0)\\ &+ \frac{\partial}{\partial \dot{p}}\frac{1}{m}F(\dot{p}, w_0)|_{\dot{p}=\dot{p}_0}(\dot{p} - \dot{p}_0)\\ &+ \frac{\partial}{\partial w}\frac{1}{m}F(\dot{p}_0, w)|_{w=w_0}(w - w_0)\\ &+ \frac{\partial}{\partial \phi}\frac{1}{m}T_N(\eta, T_0)|_{\phi=\phi_0}(\phi - \phi_0)\\ &+ \frac{\partial}{\partial \theta}\frac{1}{m}T_N(\eta, T_0)|_{\theta=\theta_0}(\theta - \theta_0)\\ &+ \frac{\partial}{\partial T}\frac{1}{m}T_N(\eta_0, T)|_{T=T_0}(T - T_0)\end{aligned} \tag{4.3}$$

The first term is simply the acceleration at the previous time-step, $\ddot{p}_0$. This is obtained by using the measurement of the accelerometer, which measures specific force in the body frame, rotating these measurements to the NED frame and adding to them the acceleration due to gravity. Term 2 and 3 are neglected because an aerodynamic drag model of the UAV is not used in order to avoid complexity, and estimation of the change in wind is extremely difficult. Therefore, equation (4.3) can be simplified as

$$\ddot{p} = \ddot{p}_0 + \frac{1}{m}G(\eta_0, T_0)(u - u_0) \tag{4.4}$$

where $u = [\phi \;\; \theta \;\; T]^T$, and G is the matrix given below, where sin and cos are represented as 's' and 'c' respectively.

$$G(\eta, T) = \begin{bmatrix} (c\phi s\psi - s\phi c\psi s\theta)T & (c\phi c\psi c\theta)T & s\phi s\psi + c\phi c\psi s\theta \\ (-s\phi s\psi s\theta - c\psi c\phi)T & (c\phi s\psi c\theta)T & c\phi s\psi s\theta - c\psi s\phi \\ -c\theta s\phi T & -s\theta c\phi T & c\phi c\theta \end{bmatrix} \tag{4.5}$$

4.1 Filtering

As mentioned in the previous section, the acceleration at the previous time step is obtained using the measurements from the accelerometer. However, these measurements are noisy, and so have to be filtered. The filter used is a second order filter, adopted from [36]

$$H(s) = \frac{{\omega_n}^2}{s^2 + 2\zeta\omega_n s + {\omega_n}^2} \tag{4.6}$$

Although the use of filter removes the noise present in the measurements, it introduces a delay which needs to be accounted for. Therefore, all terms with subscript 0 are filtered with the same filter and denoted with the subscript f. Equation (4.4) can then be re-written as

$$\ddot{p}_d = \ddot{p}_f + \frac{1}{m} G(\eta_0, T_0)(u_c - u_f) \tag{4.7}$$

Here, the $\ddot{p}$ on the left side of Equation (4.4) is given the subscript d to indicate that it is the desired acceleration command that the INDI controller receives and the u term is given a subscript c to indicate that it is the command that will be sent to the inner loop attitude controller. From Equation (4.7), we can obtain the INDI control law

$$u_c = u_f + mG^{-1}(\eta_0, T_0)(\ddot{p}_d - \ddot{p}_f) \tag{4.8}$$

The u_c calculated from Equation (4.8) is then sent to the inner loop controller.

Chapter 5

Experiments and Results

In this chapter, the flight experiments that are carried out to validate the vector-field based path following algorithm are detailed. The first set of experiments required the UAV to follow a circular path of varying radii and at varying velocities. The next set of experiments required the UAV to follow a more complex path, which was discretized into multiple simple path segments, consisting of straight lines and circles.

The main test metric to analyze the performance of the path following algorithms is the mean cross track error, which refers to the difference between the UAV's actual position and the desired position. The experiments were first conducted with the UAV using the carrot chasing algorithm to track the desired path. Then, for the same conditions of circle radius and airspeed, the experiment was repeated with the UAV using the vector field algorithm to track the desired path. Important data such the position, velocity and acceleration of the UAV was logged during the experiments, in order to analyze and obtain the average tracking error.

5.1 Experimental Setup

The experiments were conducted at the Cyberzoo, in the faculty of Aerospace Engineering at TU Delft. The dimensions of the Cyberzoo are 10m x 10m x 2m, and it is equipped with an infrared motion capture system, Optitrack [37] , that is used to track the position of the UAV with high accuracy. The UAV platform used for the experiments was the Bebop quadrotor from Parrot. The vector field algorithm was implemented in the Paparazzi UAV open-source autopilot [14], which was loaded onto the Bebop.

5.2 Circular Path Following

The first set of experiments required the UAV to follow a circular path. The desired path $\mathcal{P}$ is described using the general equation of an ellipse

$$\varphi(p) = \left(\frac{(p_x - c_x)cos\alpha - (p_y - c_y)sin\alpha}{a}\right)^2 + \left(\frac{(p_x - c_x)sin\alpha + (p_y - c_y)cos\alpha}{b}\right)^2 - 1 \tag{5.1}$$

where, $c = [c_x\ c_y]^T$ is the centre of the ellipse with respect to the inertial navigation frame, a and b are the lengths of the major and minor semi-axes of the ellipse respectively, and α is the rotation angle of the ellipse with respect to the horizontal axis of the frame. $p = [p_x\ p_y]^T$ is the position of the UAV, and therefore, when the UAV is on the ellipse, the value of the function $\varphi(p) = 0$. For the purpose of the experiments, the values of a and b were always kept equal, thereby making the desired path a circle of radius $r = a = b$, and the value of α was set to zero.

For the experiments, the radii that were considered are $r = 1.5m$, $r = 2m$ and $r = 2.5m$. The dimensions were selected according to the space limitations of the Cyberzoo and the infrared tracking system. For each of the 3 radii, the UAV was commanded to fly at airspeeds of $1m/s$, $2m/s$ and $3m/s$, therefore making a total of 9 experiments. These 9 experiments were performed for both, the carrot-chasing algorithm as well as the vector-field based algorithm, and essential data such as position, velocity and acceleration of the UAV was logged on-board for all 18 experiments. Only the experiments corresponding to the radius of $r = 2m$ are described in the research paper. In the next section, the results of all 18 experiments are presented and discussed.

5.2.1 Results

The main test metric for the experiments was the cross track error. The average cross-track error for both the algorithms, over the course of the flight for each experiment is shown in Tables 5.1-5.2.

Table 5.1: Avg. Distance Errors (in m) when using Carrot-chasing

	v:1m/s	**v:2m/s**	**v:3m/s**
r:1.5m	0.1185	0.4227	0.7036
r:2m	-0.0124	0.3984	0.6829
r:2.5m	-0.1469	0.3344	0.6858

The values of the average cross-track error are compared and analysed visually in Figure 5.1, where the rate of increase of cross-track error with the increase in velocity is easier to observe.

Figure 5.2 shows the position of the UAV for each of the 9 test conditions. The position when using the carrot chasing controller is represented in red, and the position when

Table 5.2: Avg. Distance Errors (in m) when using Vector Field Method

	v:1m/s	**v:2m/s**	**v:3m/s**
r:1.5m	0.0892	0.1563	0.2107
r:2m	0.0698	0.1517	0.2216
r:2.5m	0.0551	0.1533	0.2244

Figure 5.1: Average Distance Error for all experiments

using the vector-field controller is represented in blue. The desired path of the UAV is represented in orange.

Figure 5.3 shows the distance of the UAV from the centre of the desired circle, where the cross track error can be better perceived. The orange line represents the desired distance from the centre of the circle. The blue line represents the average distance from the centre when using the vector field controller, over the course of the whole experiment, while the red represents the same when using the carrot chasing controller.

5.2.2 Discussion

From the figures presented in the previous section, the increase in position error with increase in velocity is evident, however it is seen that the performance of the vector field algorithm is considerably better, particularly at higher velocities. The existence of a position error is due to the fact that instantaneous acceleration commands are calculated by the vector field controller, which are passed through the various control loops and finally the UAV receives instantaneous rotational velocity commands for the rotors. However, the rotors have some dynamics and take a finite amount of time before they achieve the desired rotational velocities. During the time that it takes for the rotors to reach the desired rotational velocity, the UAV travels forward, based on the command it last received, which causes a small steady state error. As the velocity of the UAV increases, it travels further in the time between two successive commands, and hence the error increases with an increase in velocity.

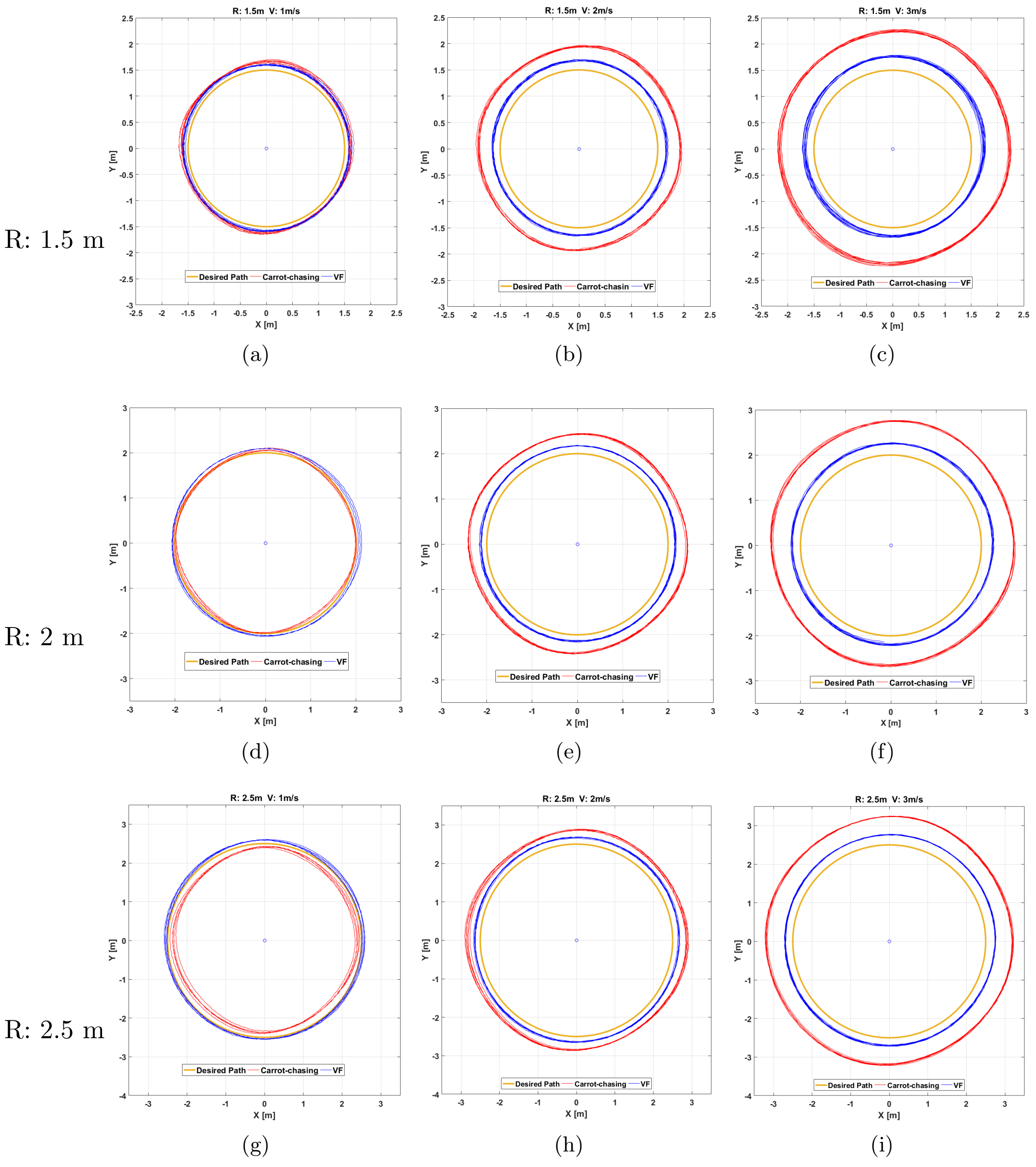


Figure 5.2: Position of the UAV for 3 radii and 3 velocities

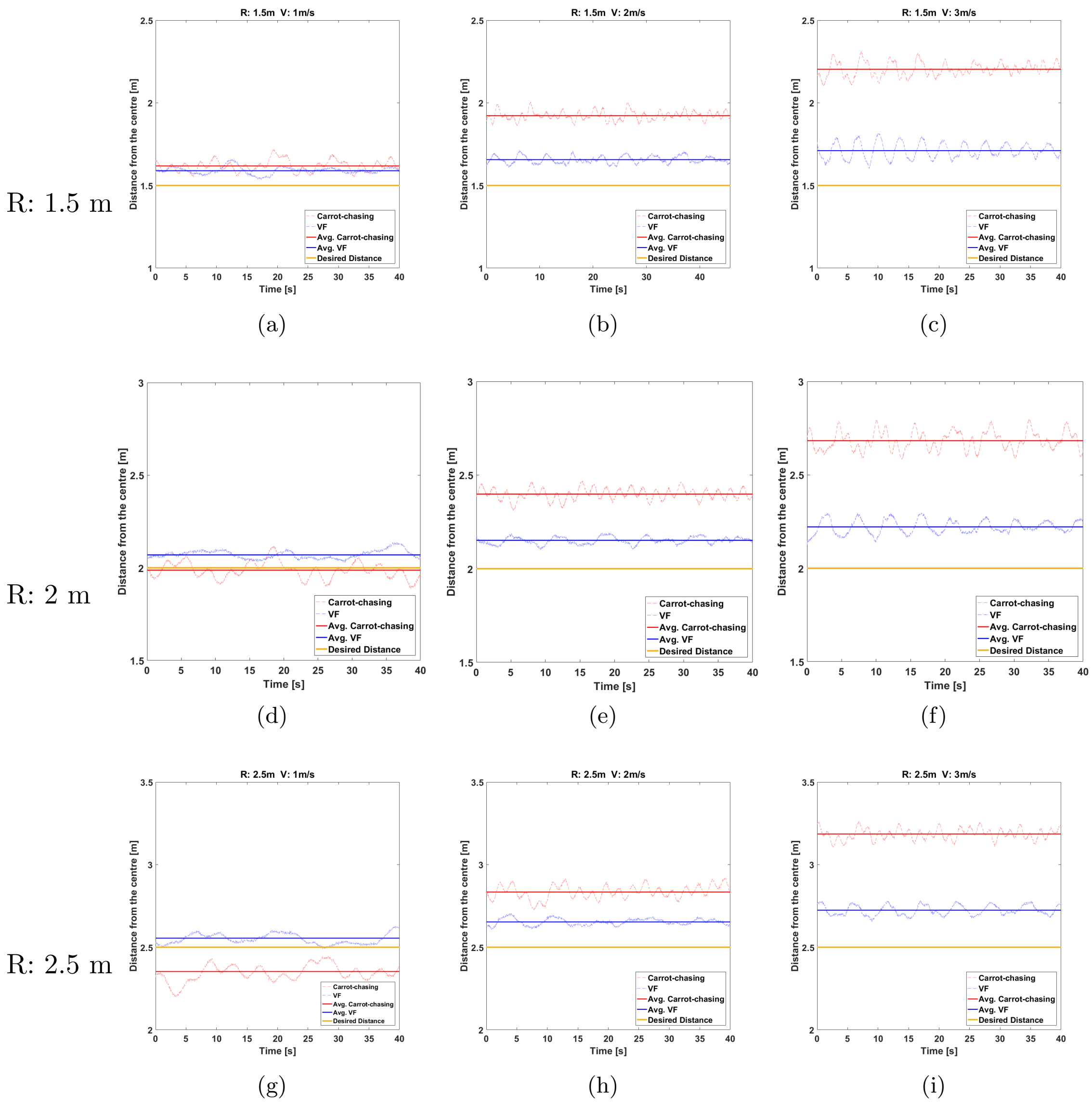


Figure 5.3: Distance of the UAV from the centre for 3 radii and 3 velocities

Chapter 6

Conclusion and Future Work

The goal of this research was to develop a path following method that overcomes the limitations of a common path following technique used currently, known as the carrot-chasing algorithm. The use of vector fields was employed in order to obtain a control algorithm that is capable of controlling both the position of the UAV, as well as maintain a desired speed profile as an independent variable along the desired path.

The work assumes that the autopilot of the UAV is equipped with an attitude controller and an acceleration controller. Thus, the goal was to build a position and velocity controller that generates the correct desired acceleration commands which it sends to the acceleration controller as a reference signal.

The vector field based path following technique in this research was developed in two steps. The first step was to compute a vector field for a given path, such that the integral curves of the vector field lead to the desired path. Once a vector field leading to the desired path was created, the next step was to generate the acceleration commands that drive the UAV along the integral curves of the vector field. First, a feedback acceleration was obtained by comparing the actual and desired velocities of the UAV. A second feedforward acceleration term was then calculated, that uses the information of the change in the desired path over time, by means of the Hessian of the path function. By combining these two acceleration terms, the final desired acceleration is obtained, which is sent to the outer INDI controller, which is an acceleration controller.

The performance of the vector field based path following algorithm was verified by means of indoor flight tests. The first set of tests required the UAV to follow a circle as the desired path. It was seen that for an increase in velocity, a small non-zero steady state error appears, due to the fact that the UAV is receiving instantaneous acceleration commands, but the motors of the UAV take some finite time to reach the desired commands, and in the time between two successive commands, the UAV flies according to the previously received command. However, performance of the UAV when using the vector field algorithm is shown to be significantly better than when using the carrot chasing algorithm. The rate of increase of position error with increase in velocity for the carrot-chasing algorithm was shown to be much higher than the same when using the vector field algorithm.

For a second set of tests, multiple simple path segments were combined to form a complicated path. Two half circle segments were combined with 2 line segments to form a figure eight. The UAV is only under the influence of a single vector field, which corresponds to the path segment that the UAV is tracking. Once the UAV reaches the end of the segment, the vector field switches to that of the next segment. The performance of both path following algorithms was compared and it was shown that at high velocities, the vector field based algorithm has a lower tracking error than the carrot chasing algorithm.

6.1 Future Work

Based on the work presented in this report, a few recommendations for future work are provided.

- As discussed in the Results section, there still exists a small position error, due to the fact that the rotors on the UAV take some finite time to achieve the commanded rotational velocities as dictated by the controller. In order to further improve the performance of the algorithm, the response time of the rotors could be accounted for in the calculation of the desired accelerations. By measuring the response curve of the rotors, the time required to achieve a desired increment in the rotational velocity can be calculated, say t^*. Since the velocity and position of the UAV can be measured, the position of the UAV after t^* seconds can be predicted. If this predicted position is used to calculate the position error, and therefore the desired acceleration, the performance of the controller could see an improvement.
- Since the position error with respect to the desired path, the gradient and Hessian are calculated at every step, it is possible to extend the vector-field based method discussed in this report to time varying paths, in particular paths that vary slowly over time. Work in this direction is done in [38] where a guidance vector field is first constructed for a stationary target, and then modified based on the movement of the target. A slightly different approach to solving the moving path following problem is also presented in [39], where a fixed path is considered to be attached to a moving reference frame.
- The work in this research only considered 2D planar paths, but the method discussed can be extended directly to 3D paths, as the outer loop INDI controller is capable of receiving 3D acceleration set-points. In [32], a 3D path is described as the intersection of 2 surfaces, and the UAV is made to simultaneously approach both surfaces.
- An advantage of the vector field based method is that the path to be followed could be any planar smooth shape, not necessarily just a circle or a straight line, for instance sinusoids [34] and Cassini ovals [31]. However, for more complex paths, particularly paths that intersect themselves, such as the figure eight path used in this research, it is difficult to generate a vector field for the entire path due to the presence of discontinuities at path intersections. In such cases, the path is split into multiple simpler segments, and the vector field for individual segments can be generated. In the current research, the path is split in such a way that each

individual segment is either a straight line or a circle, since the functions describing such shapes are well known. However, this may cause some overshoot, particularly if there is an abrupt change in path angle at the point of connection of two path segments. A better approach might be to analyze the path and split it at points where no change in path angle occurs. Then, for each path segment a function that best approximates its shape can be generated, and the vector fields can be computed for each path segment.

Bibliography

[1] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, “Supporting wilderness search and rescue using a camera-equipped mini uav,” *Journal of Field Robotics*, vol. 25, no. 1-2, pp. 89–110, 2008.

[2] R. W. Beard, T. W. McLain, D. B. Nelson, D. Kingston, and D. Johanson, “Decentralized cooperative aerial surveillance using fixed-wing miniature uavs,” *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1306–1324, 2006.

[3] T. R. Kurfess, *Robotics and automation handbook.* CRC press, 2004.

[4] B. Siciliano and O. Khatib, *Springer handbook of robotics.* Springer, 2016.

[5] U. Lee, S. Yoon, H. Shim, P. Vasseur, and C. Demonceaux, “Local path planning in a complex environment for self-driving car,” in *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2014 IEEE 4th Annual International Conference on.* IEEE, 2014, pp. 445–450.

[6] C. Samson, “Path following and time-varying feedback stabilization of a wheeled mobile robot,” in *Proceedings of the international conference on advanced robotics and computer vision*, vol. 13, 1992, pp. 1–1.

[7] E. Peymani and T. I. Fossen, “Path following of underwater robots using lagrange multipliers,” *Robotics and Autonomous Systems*, vol. 67, pp. 44–52, 2015.

[8] E. P. Anderson, R. W. Beard, and T. W. McLain, “Real-time dynamic trajectory smoothing for unmanned air vehicles,” *IEEE Transactions on Control Systems Technology*, vol. 13, no. 3, pp. 471–477, 2005.

[9] S. Park, J. Deyst, and J. P. How, “A new nonlinear guidance logic for trajectory tracking,” in *AIAA guidance, navigation, and control conference and exhibit*, 2004, pp. 16–19.

[10] I. Kaminer, A. Pascoal, E. Hallberg, and C. Silvestre, “Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control,” *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 1, pp. 29–38, 1998.

[11] W. Ren and R. W. Beard, “Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints,” *IEEE Transactions on Control Systems Technology*, vol. 12, no. 5, pp. 706–716, 2004.

[12] A. P. Aguiar, D. B. Dačić, J. P. Hespanha, and P. Kokotović, “Path-following or reference tracking?: An answer relaxing the limits to performance,” *IFAC Proceedings Volumes*, vol. 37, no. 8, pp. 167–172, 2004.

[13] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotovic, “Path-following for nonminimum phase systems removes performance limitations,” *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 234–239, 2005.

[14] “Paparazzi uav,” http://wiki.paparazziuav.org/wiki/Main_Page, [Online].

[15] S. Park, J. Deyst, and J. P. How, “Performance and lyapunov stability of a nonlinear path-following guidance method,” *Journal of Guidance Control and Dynamics*, vol. 30, no. 6, pp. 1718–1728, 2007.

[16] J. Ackermann, J. Guldner, W. Sienel, R. Steinhauser, and V. I. Utkin, “Linear and nonlinear controller design for robust automatic steering,” *IEEE Transactions on Control Systems Technology*, vol. 3, no. 1, pp. 132–143, 1995.

[17] A. Ratnoo, P. Sujit, and M. Kothari, “Adaptive optimal path following for high wind flights,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 12 985–12 990, 2011.

[18] S. Lee, A. Cho, and C. Kee, “Integrated waypoint path generation and following of an unmanned aerial vehicle,” *Aircraft Engineering and Aerospace Technology*, vol. 82, no. 5, pp. 296–304, 2010.

[19] F. L. Lewis, *Applied optimal control and estimation.* Prentice Hall PTR, 1992.

[20] T. Faulwasser, B. Kern, and R. Findeisen, “Model predictive path-following for constrained nonlinear systems,” in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on.* IEEE, 2009, pp. 8642–8647.

[21] R. Skjetne, T. I. Fossen, and P. V. Kokotović, “Robust output maneuvering for a class of nonlinear systems,” *Automatica*, vol. 40, no. 3, pp. 373–383, 2004.

[22] D. Gu and H. Hu, “Receding horizon tracking control of wheeled mobile robots,” *IEEE Transactions on control systems technology*, vol. 14, no. 4, pp. 743–749, 2006.

[23] K. Kanjanawanishkul and A. Zell, “Path following for an omnidirectional mobile robot based on model predictive control,” in *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on.* IEEE, 2009, pp. 3341–3346.

[24] Y. Kim and M. Yamakita, “Model predictive path-following for bike robot,” in *SICE Annual Conference (SICE), 2013 Proceedings of.* IEEE, 2013, pp. 1592–1597.

[25] K. Yang, Y. Kang, and S. Sukkarieh, “Adaptive nonlinear model predictive path-following control for a fixed-wing unmanned aerial vehicle,” *International Journal of Control, Automation and Systems*, vol. 11, no. 1, pp. 65–74, 2013.

[26] P. Sujit, S. Saripalli, and J. B. Sousa, "Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless," *IEEE Control Systems*, vol. 34, no. 1, pp. 42–59, 2014.

[27] D. R. Nelson, D. B. Barber, T. W. McLain, and R. W. Beard, "Vector field path following for miniature air vehicles," *IEEE Transactions on Robotics*, vol. 23, no. 3, pp. 519–529, 2007.

[28] D. A. Lawrence, E. W. Frew, and W. J. Pisano, "Lyapunov vector fields for autonomous uav flight control," in *AIAA Guidance, Navigation and Control Conference and Exhibit.* Hilton Head, South Carolina, USA, 2007, pp. 2007–6317.

[29] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[30] K. Sigurd and J. How, "Uav trajectory design using total field collision avoidance," *American Institute of Aeronautics and Astronautics*, 2003.

[31] Y. A. Kapitanyuk, A. V. Proskurnikov, and M. Cao, "A guiding vector-field algorithm for path-following control of nonholonomic mobile robots," *IEEE Transactions on Control Systems Technology*, 2017.

[32] J. Wang, I. A. Kapitaniuk, S. A. Chepinskiy, D. Liu, and A. J. Krasnov, "Geometric path following control in a moving frame this work was partially financially supported by government of russian federation, grant 074-u01," *IFAC-PapersOnLine*, vol. 48, no. 11, pp. 150–155, 2015.

[33] H. Oh, S. Kim, H.-s. Shin, and A. Tsourdos, "Coordinated standoff tracking of moving target groups using multiple uavs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 2, pp. 1501–1514, 2015.

[34] H. G. de Marina, Y. A. Kapitanyuk, M. Bronz, G. Hattenberger, and M. Cao, "Guidance algorithm for smooth trajectory tracking of a fixed wing uav flying in wind flows," *arXiv preprint arXiv:1610.02797*, 2016.

[35] E. J. Smeur, G. C. de Croon, and Q. Chu, "Cascaded incremental nonlinear dynamic inversion control for mav disturbance rejection," *arXiv preprint arXiv:1701.07254*, 2017.

[36] B. J. Bacon, A. J. Ostroff, and S. M. Joshi, "Reconfigurable ndi controller using inertial sensor failure detection & isolation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 4, pp. 1373–1383, 2001.

[37] "Optitrack - motion capture systems," http://optitrack.com/, [Online].

[38] E. W. Frew, D. A. Lawrence, and S. Morris, "Coordinated standoff tracking of moving targets using lyapunov guidance vector fields," *Journal of guidance, control, and dynamics*, vol. 31, no. 2, pp. 290–306, 2008.

[39] Y. A. Kapitanyuk, H. G. de Marina, A. V. Proskurnikov, and M. Cao, "Guiding vector field algorithm for a moving path following problem," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 6983–6988, 2017.