An Optimized Circulating Vector Field Obstacle Avoidance Guidance for Unmanned Aerial Vehicles

A thesis presented to

the faculty of

the Russ College of Engineering and Technology of Ohio University

In partial fulfillment

of the requirements for the degree

Master of Science

Garrett S. Clem

August 2018

This thesis titled

An Optimized Circulating Vector Field Obstacle Avoidance Guidance for Unmanned Aerial Vehicles

by

GARRETT S. CLEM

has been approved for

the Department of Mechanical Engineering

and the Russ College of Engineering and Technology by

Jay P. Wilhelm

Assistant Professor of Mechanical Engineering

Dennis Irwin

Dean, Russ College of Engineering and Technology

Abstract

CLEM, GARRETT S., M.S., August 2018, Mechanical Engineering

An Optimized Circulating Vector Field Obstacle Avoidance Guidance for Unmanned Aerial Vehicles

Director of Thesis: Jay P. Wilhelm

Unmanned Aerial Vehicles (UAVs) conventionally navigate by following a series of pre-planned waypoints. When encountering an obstacle during flight, such as no-fly zone or other aircraft, the vehicle's path or waypoints may need to be re-planned. Waypoint guidance can be used to avoid obstacles which is typically generated off-line and relayed to the UAV requiring active communications. Vector Fields (VFs) that are generated based on a desired path can provide guidance around a newly discovered obstacle without the need for a re-plan. Convergence and circulation terms that make up the VF can be controlled by modifying static multiplicative scalars. Additionally, a decay radius can be applied to limit the effects of a repulsive VF, representing an obstacle. VF convergence and circulation components were optimized to minimize deviation from a desired path when guiding around a circular obstacle. Results indicated that the developed VF obstacle avoidance optimizer guidance directs the UAV along a similar route as waypoint guidance without the need for re-planning.

In memory of Jack Clem

Acknowledgments

Throughout my graduate studies I have had the privilege of working with many unique and sophisticated people. First and foremost, I would like to thank my advisor Dr. Jay Wilhelm for his seemingly limitless academic support. Without his leadership this work would not have been completed. I would like to also thank my committee members, Dr. Casbeer, Dr. Williams, and Dr. Uijt de Haag for their time, suggestions, and stimulating discussions in regards to the contents of this research.

To my first lab colleague, Gina, your mentoring and friendship were invaluable and I cannot see myself completing this program without your support and humor. Hunter and Travis, your assistance with debugging the control software was vital to the success of this project. My roommates, Sean, Justin, and Jacob, you all provided welcome distractions. To my mother, who I grew closer with over the past few years, this is for you. Lastly, I would like to thank my best friend Emily whose love and support constantly reminded me why I continued my education to better myself.

Table of Contents

List of Tables

List of Figures

List of Symbols

$\dot{\theta}$	UAV maximum turn rate [deg/s]
$\overrightarrow{X}$	UAV position [m]
t	time [s]
t	time [s]
$\overrightarrow{U}$	UAV velocity [m/s]
θ	UAV heading [deg]
dt	Discrete time step [s]
$\overrightarrow{R}$	Potential Field heading guidance vector [deg]
$\overrightarrow{F}_t$	Potential Field artificial attractive force [-]
(x_t, y_t)	Potential Field goal position [m]
d_t	Potential Field range to goal from robot [m]
$\overrightarrow{F}_{i,j}$	Potential Field obstacle artificial force [-]
(x_i, y_j)	Potential Field obstacle cell position [m]
i, j	Potential Field obstacle cell index [-]
F_{cr}	Potential Field constant repulsive artificial force [-]
W	Potential Field vehicle width [m]
$C_{i,j}$	Potential Field cell certainty [-]
$d_{i,j}$	Potential Field range to cell center [m]
χ^d	Lyapunov guidance vector [deg]
$\chi^{\inf}$	Lyapunov course approach angle [deg]
y	UAV's lateral distance to path [m]
$\bar{k}$	Lyapunov transition constant [-]
ψ	UAV angular position around loiter circle [deg]
r	Radius of circular path [m]
d	Range to path or obstacle center [m]
n	Number of spatial dimensions [-]
q	Spatial dimension set [-]
$\alpha_i(x_1, x_2, ...x_n, t)$	Implicit surface function [-]
$\overrightarrow{V}$	Guidance vector [deg]
G	Convergence weight [-]
H	Circulation weight [-]
L	Time-varying weight [-]
$\overrightarrow{V}_{conv}$	Convergence vector [deg]
$\overrightarrow{V}_{circ}$	Circulation vector [deg]
$\overrightarrow{V}_{tv}$	Time varying vector [deg]
M	Gradient matrix [-]
a	Velocity column vector [-]
$\overrightarrow{V}_p$	Path following guidance [deg]

$\overrightarrow{V}_o$	Obstacle avoidance guidance [deg]
P	Obstacle field decay weight [-]
R	Obstacle field decay edge [-]
δ	Straight target path angle [deg]
$\|\| \bullet \|\|$	Vector magnitude []
x_c	Obstacle x position [m]
y_c	Obstacle y position [m]
$\bar{x}$	x range to obstacle [m]
$\bar{y}$	y range to obstacle [m]
r_o	Obstacle radius [m]
m	Obstacle radius multiplier [-]
k	Decay radius multiplier [-]
θ_r	UAV turning radius [m]
γ	Path deviation cost function [-]
tf	Time final [-]
$\bar{\gamma}$	Path deviation cost function with obstacle penalization [-]
$j(x, y)$	Obstacle penalization function [-]

List of Acronyms

UAV	Unmanned Aerial Vehicles
VF	Vector Field
UAS	Unmanned Aerial System
VFF	Virtual Force Field
TPLVF	Tangent Plus Lyapunov Vector Field
RRT*	Optimal Rapid Radom Trees
DT	Delauny Triangulation
GVF	Gradient Vector Field

1 Introduction

1.1 Motivation and Problem Statement

Unmanned aerial vehicles (UAV) are pilotless aircraft used by military, police, and civilian communities for tasks such as damage assessment [1], surveying [2], and target tracking [3–8]. Many of these tasks depend on the UAVs ability to autonomously follow a path while potentially avoiding obstacles and no-fly zones. UAV paths are typically followed by implementing heading guidance systems such as waypoint [9], Proportional-Integral-Derivative (PID) [10], non-linear guidance laws [11], or Linear Quadratic Regulator (LQR) [12]. Conventional path following guidance systems are typically not capable of avoiding obstacles without partially or completely re-planning the mission path. Vehicle paths are typically generated on a remote ground station and relayed to the UAV's autopilot which may be impossible under certain conditions, such as flying beyond line-of-sight. Heading guidance that accomplishes path following while avoiding obstacles without the need for re-planning waypoints may be beneficial. Methods of minimally deviating from a planned mission path when new obstacles are encountered are desired. Avoiding obstacles without path re-planning has been achieved by vector field guidance [5, 13–16] guidance.

Vector Field (VF) guidance is a method that is mainly used for path following [13–19] and can be useful for obstacle avoidance [20, 21]. VFs can produce continuous heading vectors that can be used to guide a UAV to coverage and follow a path. Vectors are calculated by summing together convergence and circulation terms that are weighted by static scalars. Obstacles can be represented as repulsive VFs that direct the UAV away from the no-fly zone. Strictly repulsive VFs do not always route the UAV around an obstacle and can cause singularities, small regions where path following and obstacle vectors cancel. Modifying repulsive VFs to include circulation and an appropriate decay radius may be

used to produce an optimal guidance. **Problem Statement: Utilize vector field guidance to enable an optimized obstacle avoidance without re-planning waypoints.**

1.2 Methods Overview

This research was conducted in three phases consisting of numerical simulations and indoor flight experiments. Phase I investigated the construction of path following and repulsive Gradient Vector Fields (GVFs) followed by the characterization of regions of null guidance in summed fields, called singularities. Phase II investigated a numerical method for optimizing obstacle field decay radius and circulation weight which minimizes a path deviation cost function. Lastly, the optimized GVF was implemented on an indoor flying quadcopter with Dubin's constraints in order to emulate a fixed wing aircraft. Flight tests and simulation path deviation costs were compared. A summary of each phase's objective is provided below.

1.3 Phase I

Characterize and present a method of locating singularities in a summed GVF. Equations for constructing path following and repulsive GVFs are presented. Target path following and obstacle fields are summed together and the presence of singularities is discussed. A method of locating singularities numerically in a summed field is provided.

1.4 Phase II

Determine a combination of circulation and decay radius for a circular obstacle GVF that produces an optimized obstacle avoidance. A UAV modeled as a Dubin's vehicle was used to demonstrate GVF guidance for converging and following a straight path. A worst case collision scenario was presented where an obstacle centered on a planned path was to be avoided. Obstacle GVF decay radius and circulation were optimized to minimize a path deviation cost function. The optimized GVF was compared against both waypoint and potential field methods for the UAV's path deviation.

1.5 Phase III

Demonstrate optimized GVF guidance on multirotor UAV flying with fixed wing turn-rate constraints. The modified GVF developed in Phase II was implemented on a Crazyflie 2.0 quadcopter flying in an indoor environment. Dubin's turn rate constraints were applied in order to emulate the behavior of a fixed wing UAV. Deviation from a planned path while avoiding a circular obstacle was compared to simulations for several avoidance scenarios.

1.6 Summary of Objectives

Each phase consisted of an **objective** that was accomplished by executing several *tasks*. Completion of all objectives and phases resulted in the final deliverable.

Phase I Objective: Demonstrate and locate singularities in a summed gradient vector field
Tasks:

1. *Construct GVF equations from literature*
2. *Evaluate scenarios where singularities are expected*
3. *Characterize location of singularities*

Phase II Objective: Determine combination of repulsive gradient vector field circulation and decay radius that minimizes a path deviation cost function
Tasks:

1. *Define obstacles in terms of UAV turning radius*
2. *Determine combination of GVF decay radius and circulation weight that minimizes path deviation cost function*

Phase III Objective: Validate modified gradient vector field guidance with indoor quadrotor experiments emulating fixed wing turn rate constraints

Tasks:

1. *Build quadcopter and modify for roll, pitch, thrust, and yaw rate control*
2. *Program optimized GVF guidance and PID control systems into Python ground station*
3. *Repeat simulations performed in Phase II on quadcopter and compare results*

Deliverable: An optimized GVF for avoiding circular obstacles with minimal deviation from a planned path

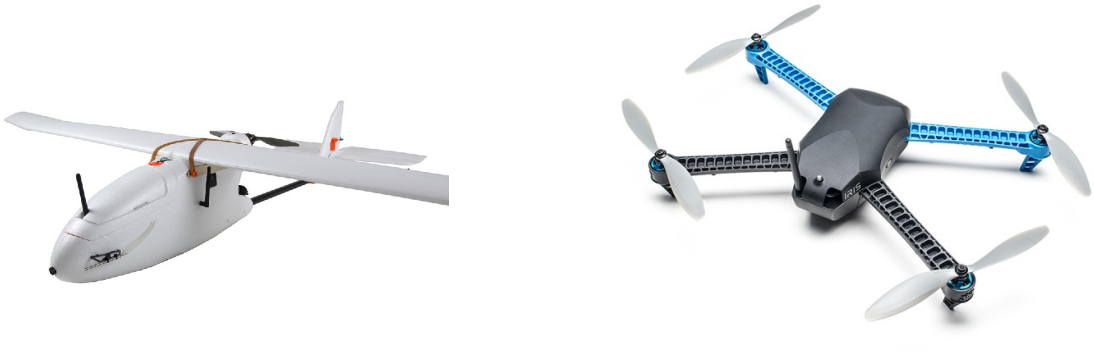
2 Literature Review

2.1 Literature Review Introduction

Unmanned aerial vehicles (UAVs) were determined from literature to a wide range applications. Tasks are typically accomplished by following pre-planned paths that are generally calculated on an off-board ground station. Guidance systems for following pre-planned paths in literature are discussed. In general, guidance systems were found to not be equipped to avoid obstacles without the need to re-plan mission paths. Methods for avoiding obstacles without the need to re-plan an obstacle free path are presented and compared.

2.2 Unmanned Aerial Vehicles

UAVs have gained popularity in both civil and private sectors due to their ability to perform tasks that may put pilots of conventional manned aircraft in harms way [22]. Tasks can be carried out by a single UAV or in cooperation with other air [6, 7, 23] or ground [24] vehicles. UAVs have several advantages over manned aircraft consisting of low operating cost, reduced risk to human operators, and the ability to perform mundane and repetitive tasks autonomously without heavy human interaction [25]. In general, UAVs are categorized into fixed wing and rotor craft varieties that range in size, payload, and flight time capabilities [26]. Fixed wing UAVs, Figure 2.1a, are typically used for tasks that require longer flight times and larger payloads, such as cameras and cargo. Rotor craft, Figure 2.1, have a lower payload capacity compared to fixed wings, have the ability to hover, and are highly maneuverable.



(a) (b)

Figure 2.1: Fixed Wing UAV (a) and Multirotor UAV (b)

UAVs can be programmed to fly autonomously without the direct input of a pilot. When flying autonomously, the vehicle typically accomplishes tasks by following a pre-planned flight path. Flight paths are often generated off-line at dedicated ground station and relayed to the UAV over radio.

2.2.1 Ground Stations

Ground stations are the software framework used to configure UAV settings, plan missions, and collect mission data. Commercial open source ground stations such as QGround Control [27] depend on a human operator's knowledge of the environment to plan an obstacle free path. Takeoff, landing, and emergency return-to-home locations are designated in safe clearings capable of accommodating the vehicle. Other tasks such as area surveying and loitering can be assigned at certain points along the mission path [28]. An example of a mission path consisting of taking off, surveying, and landing is shown in Figure 2.2

Figure 2.2: Ground Station Software Planning a Waypoint Based Mission

2.2.2 Autopilot

Paths are commonly represented as a series of finite waypoints in commercial UAV autopilots such as the Piccolo [29], Kestral [30], and Pixhawk [31]. The UAV autopilot is responsible for controlling a pre-planned path and maintaining vehicle stability while under the influence of environmental disturbances. Stable flight while path following is accomplished by implementing feed-back control, navigation, and guidance systems [26]. Feed-back refers to the closure of an open-loop control system which allows a reference error to be calculated between the desired state of the UAV, the reference, and the current state of the UAV. Reference error is used to calculate the actuator output required to modify the vehicle's attitude and position while preventing unbounded oscillation. Attitude and position feed-back is provided by the navigation system by sampling on-board sensors such as global position system (GPS) and inertial measurement units (IMUs). Filtering

and fusing noisy data from multiple sources is often accomplished through estimation techniques such as the Kalman filter [32]. The guidance system directs the UAV with a commanded heading towards a desired goal, such as a waypoint or path. A high level overview of the autopilot's systems can be seen in Figure 2.3. When simulating new guidance or navigation systems it is common to model the plant as a Dubin's vehicle with turn rate constraints for simplicity. UAVs are high order and non-linear systems that can be represented as a Dubin's vehicle [26].

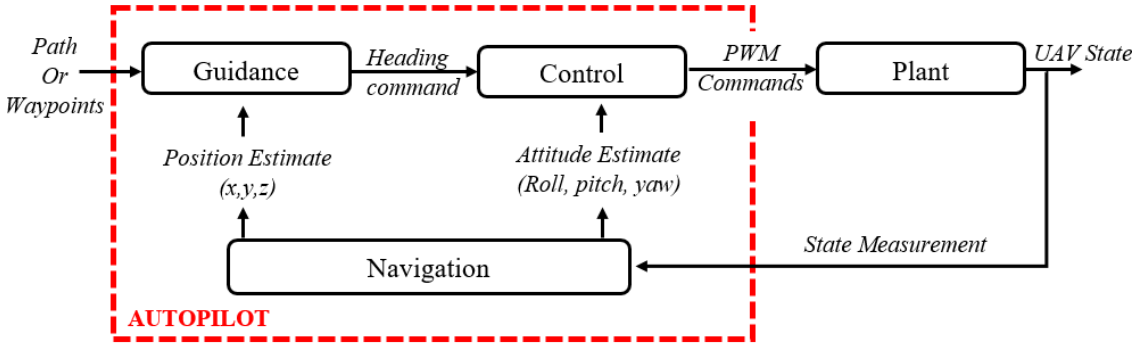


Figure 2.3: Autopilot's Navigation, Guidance, and Control Architecture

2.3 Dubins Vehicle Model

The dynamics of a UAV are often simplified by modeling the UAV as a Dubin's vehicle when simulating guidance systems [5, 13, 17–19]. It is assumed that the autopilot's control system is capable of maintaining stability, speed u, and altitude, and can turn the vehicle at a maximum fixed turn rate $\dot{\theta}$. The position of the UAV $\overrightarrow{X}$ at time t is calculated from the integral of the velocity vector $\overrightarrow{U}$, Equation 2.2. Heading θ is an input from a guidance system. Here the turn rate of the UAV is restricted to 20 degrees per second which is typical for small unmanned aircraft.

$$\overrightarrow{U}(t) = u \begin{bmatrix} cos(\theta(t)) \\ sin(\theta(t)) \end{bmatrix} \tag{2.1}$$

$$\overrightarrow{X}(t) = \overrightarrow{U}dt + \overrightarrow{X}(t-1) \tag{2.2}$$

$$\dot{\theta} \leq 20deg/s \tag{2.3}$$

2.4 UAV Guidance

Guidance methods for following a pre-planned path include geometric methods such as waypont [9] or carrot chasing [33] and control techniques such as proportional-integral-derivative (PID) [10], non-linear guidance laws [11], and linear quadratic regulator (LQR) [12]. These methods may not have the ability to avoid new or previously unknown obstacles without the need to generate a new path. Re-planning and relaying a new obstacle free path may be impossible under certain conditions, such as flying beyond radio communication range. It would be beneficial to remove the guidance system's dependence on the path planner to produce a new obstacle free path when new obstacles are encountered. Avoiding obstacles without the need to re-plan has been achieved with potential field and vector field, which represents obstacles as artificial repulsive forces that locally push away from obstacles.

2.4.1 Potential Field

Potential field is based on the principle of driving a point mass from a high potential state to a globally low potential state using artificial attractive and repulsive forces [34]. An attractive force directs the point mass in the direction of a goal located at a globally minimum potential and obstacles can be represented as repulsive forces that act locally to push the point mass away. An example of a point mass directed by potential field from a high potential to a globally minimum potential while avoiding an obstacle is shown

in Figure 2.4. Potential field is also capable of acting as a path and trajectory planning algorithm [35], possibly eliminating the off-board path planner.

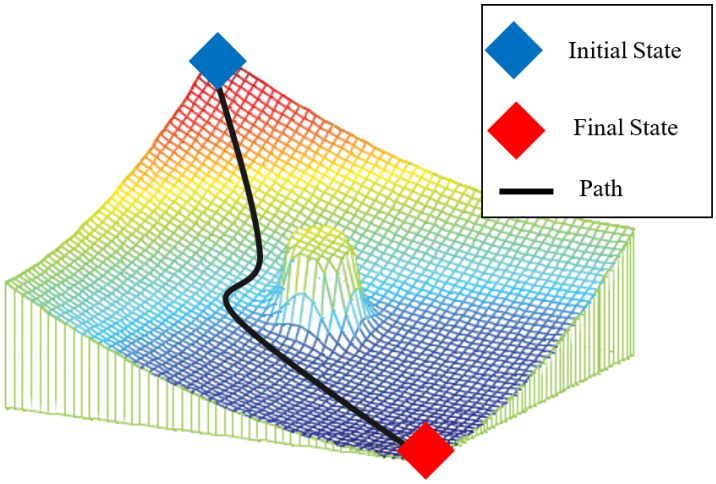


Figure 2.4: Single Obstacle Potential Field Gradient [36]

An implementation of potential field on a mobile ground robot equipped with ultrasonic sensors for real-time obstacle detection can be found in [37–39]. A differential drive robot was guided with the guidance, $\overrightarrow{R}$, which directed the robot towards a goal and away from nearby obstacles. The robot was attracted towards a goal with constant magnitude force, $\overrightarrow{F_t}$, located at (x_t, y_t) and a distance d_t from the robot. In the immediate area of the robot, an active window recorded integer certainty values inside discrete cells. Cells containing an obstacle provide a repulsive force, $\overrightarrow{F_{i,j}}$, opposite in direction to the line-of-sight from vehicle to cell location (x_i, y_j), where (i, j) represents the cell index, F_{cr} is a constant repulsive force, W the vehicle's width, $C_{i,j}$ a cell's certainty, and $d_{i,j}$ the distance to the center of the cell with respect to robots center.

$$\overrightarrow{R} = \overrightarrow{F_r} + \overrightarrow{F_t} \tag{2.4}$$

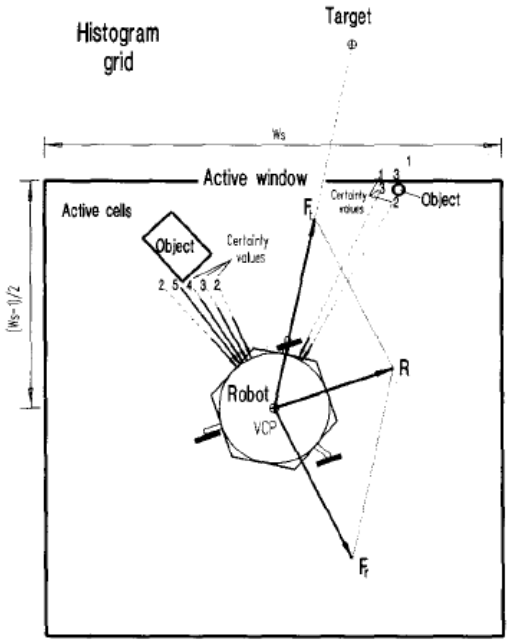


Figure 2.5: Virtual Force Field Histogram Guiding a Mobile Robot [38]

$$\overrightarrow{F_{i,j}} = \frac{F_{cr} W^n C_{i,j}}{d_{i,j}^n}\left(\frac{x_i - x_0}{d_{i,j}}\hat{x} + \frac{y_i - y_0}{d_{i,j}}\hat{y}\right) \tag{2.5}$$

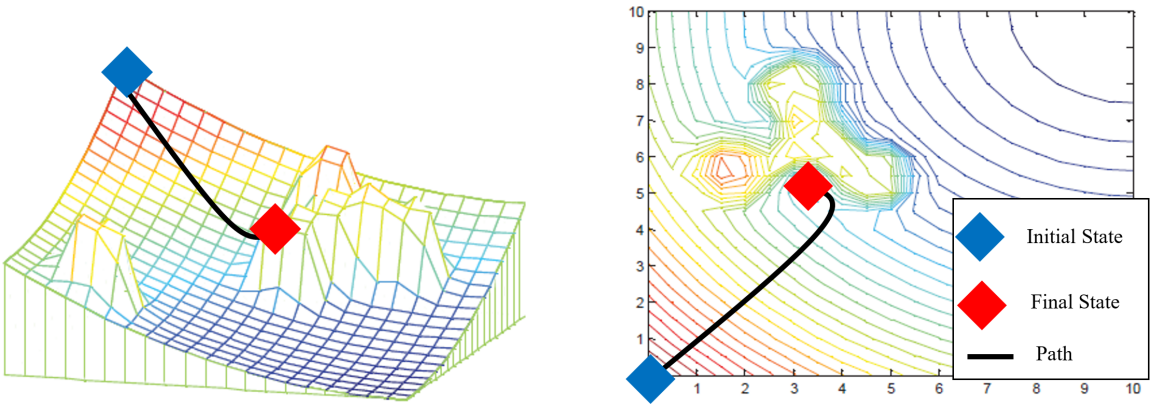
The total repulsive force exerted on the robot is determined by summing the active cells, shown in Equation 2.6

$$\overrightarrow{F_r} = \sum_{i,j} \overrightarrow{F_{i,j}} \tag{2.6}$$

$$\overrightarrow{F_t} = F_{ct}\left(\frac{x_t - x_0}{d_t}\hat{x} + \frac{y_t - y_0}{d_t}\hat{y}\right) \tag{2.7}$$

Major drawbacks to potential field were identified in [39] consisting of local minimum and oscillations in corridors. Local minimum are solutions in the potential field that are caused by the close spacing of obstacles which may result in a well that can trap the system into a local stable point prior to reaching the global minimum. A scenario similar to that shown in Figure 2.4 with two more obstacles results in a trap situation where the

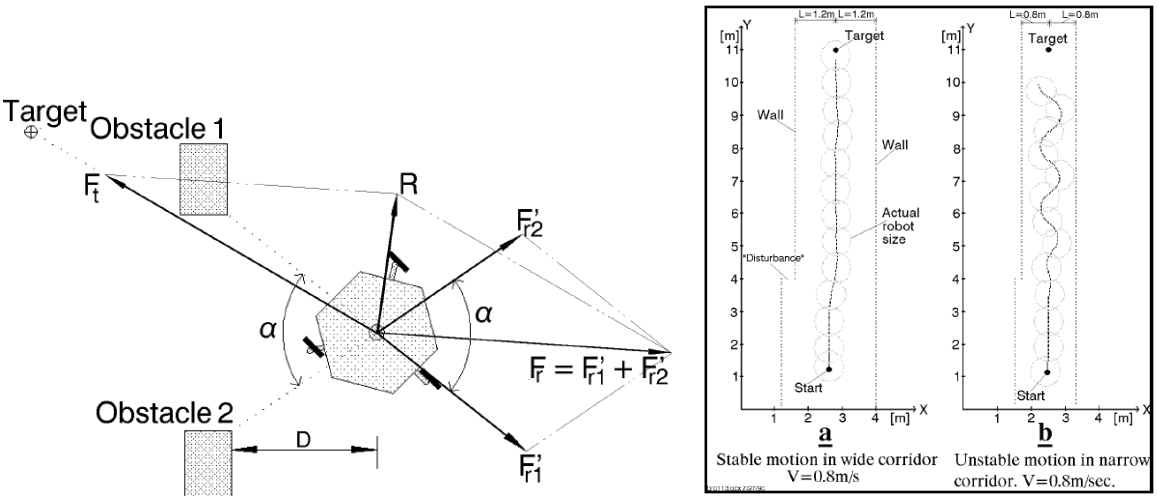
state settles into a well on the gradient, shown in Figure 2.6. Additionally, closely spaced obstacles may also be difficult to pass between, shown in Figure 2.7a. Oscillations can also be experienced near obstacles or in narrow passages at high speeds, shown in Figure 2.7b.



(a) (b)

Figure 2.6: Potential Field Local Minimum [36]

Proposed solutions to local minimum include object clustering and virtual waypoint method [36], virtual escaping route [40], and use of navigation functions [41]. Oscillations in potential field were addressed in [42] and [43].



(a) (b)

Figure 2.7: Potential Field Local Minimum [36]

Navigation functions [41] and obstacle clustering [36] have been used to prevent local minimums in potential field. Navigation functions relate kinematic constraints to the gradient potential to produce a bounded and local minimum free solution [35]. Clustering closely spaced obstacles into a single and equally repulsive obstacle prevents local minimum from forming, shown in Figure 2.8.

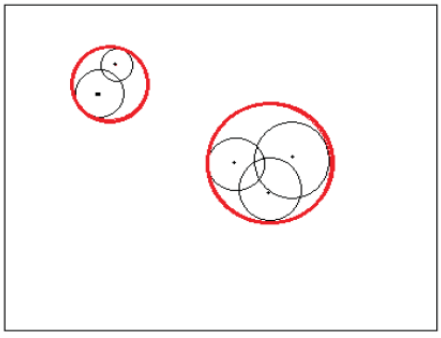


Figure 2.8: Obstacle Clustering to Prevent Local Minimum [36]

Potential Field's ability to avoid obstacles and combine path planning, trajectory planning, and control into a single system makes it an attractive guidance method for robots seeking a singular point, even with the limitations discussed in [39]. Unlike the mobile ground robots in [37], fixed wing UAVs must maintain a minimum forward velocity, have limited turning radius, and cannot converge to a single point. Vehicles with velocity and turn rate constraints may not return to a pre-planned path once the obstacle has been avoided, shown in Figure 2.9. Methods that direct a UAV to a path instead of a discrete point has been achieved with Lyapunov and gradient vector fields.

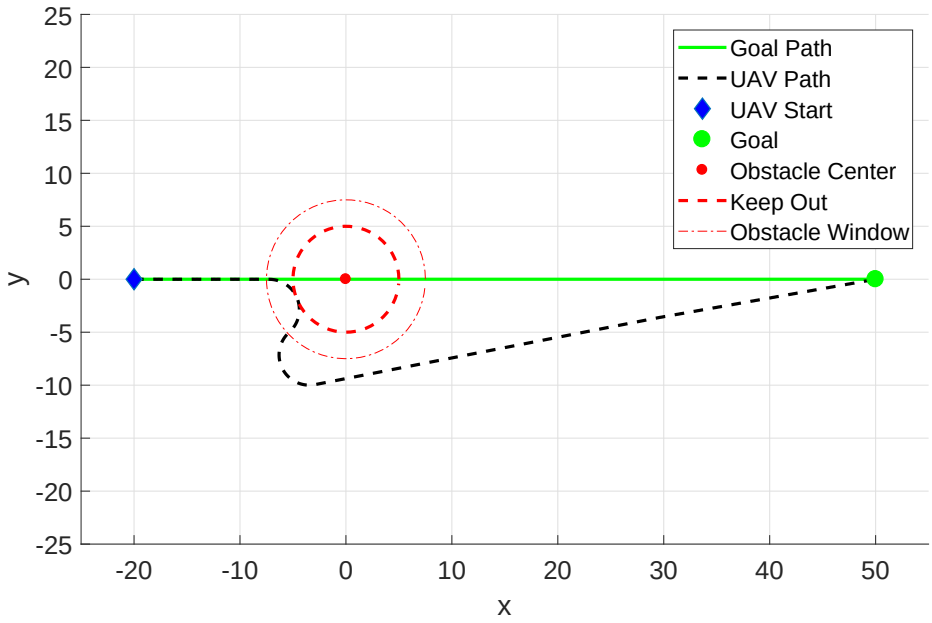


Figure 2.9: UAV Avoiding Obstacle with VFF Guidance

2.4.2 Lyapunov Vector Fields

Lyapunov vector fields for converging and following straight and circular paths were described in [17]. For converging and following a straight path, a guidance vector χ^d is determined in Equation 2.8, where χ^∞ is the course approach angle, y is the lateral

distance to the path, and $\bar{k}$ is a positive constant that determines the rate of transition between convergence and following. An example of a Lyapunov vector field converging and following a straight line is shown in Figure 2.10a.

$$\chi^d(y) = -\chi^\infty \frac{2}{\pi} \tan^{-1}(\bar{k}y) \tag{2.8}$$

For converging and following a circular path, a guidance vector χ^d is determined in Equation 2.9, where ψ is the UAVs angular position with respect to the circle, r is the paths radius, d is the distance from the circles center, and $\bar{k}$ is a positive constant that determines the transition behavior. An example of a Lyapunov vector field for converging and following a circular path is shown in Figure 2.10b.

$$\chi^d(d) = \psi - \frac{\pi}{2} - \tan^{-1}\left(\bar{k}\frac{d-r}{r}\right) \tag{2.9}$$

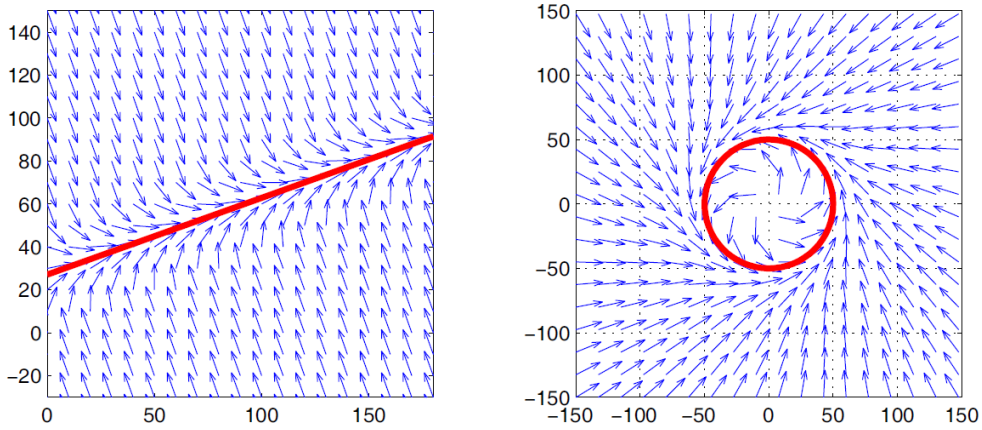


Figure 2.10: Lyapunov Vector Field for Straight Line and Circular Primitives [17]

Straight and circular path vector fields can be selectively activated throughout flight to form more complex paths, shown in [17–19, 44] and Figure 2.11. Selectively activating, or stitching together, vector fields to produce more complex paths was the preferred method

in [17] since summing together multiple fields has the possibility to produce dead zones, sinks, and singularities.

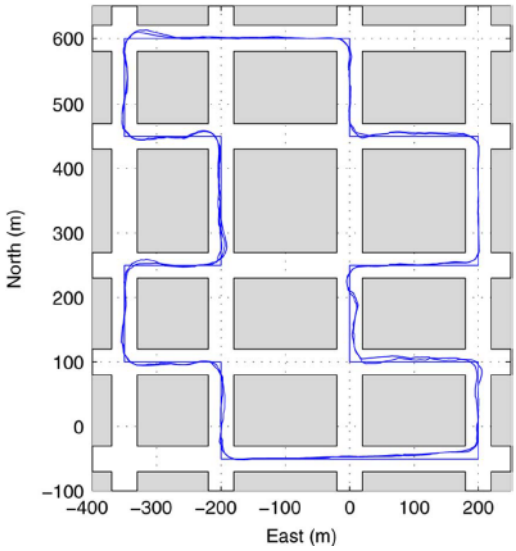


Figure 2.11: Straight Path Following in Urban Environment [17] using Lyapunov Vector Field

Lyapunov Vector field construction for curved paths was presented in [13] and is shown in Figure 2.12. Constructing a Vector Field for an arbitrary curve may allow for more complex paths and could eliminate the need for switching between primitives.

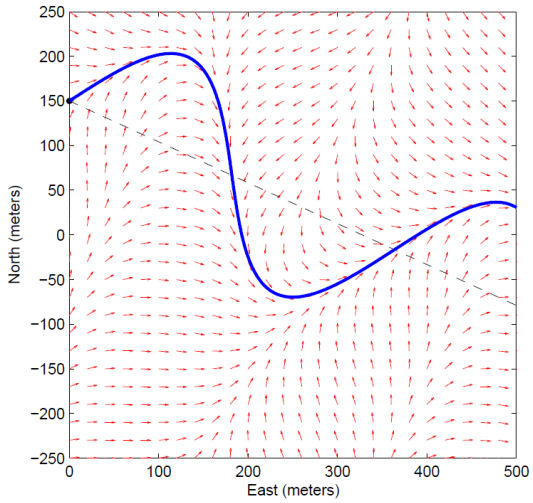
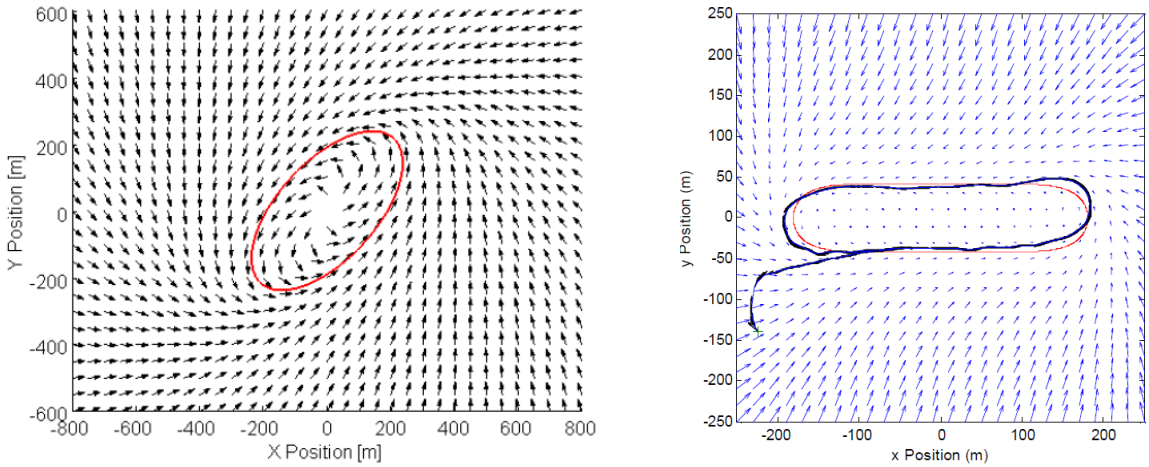


Figure 2.12: Lyapunov Vector Field Approach Curved Path [13]

Primitive circular vector fields were modified in [5, 45] via non-linear coordinate transformations to produce elliptical 2.13a, or racetrack 2.13b, fields. Transforming the circular field as a function of a Kalman filter's covariance matrix when sensing an uncertain target was investigated in [5].



(a) (b)

Figure 2.13: Elliptical VF Produced by Non-linear Coordinate Transformations a) [5] and b) [45]

Target tracking Tangent Plus Lyapunov Vector Field (TPLVF) was introduced in [46] that produced shorter paths compared to Lyapunov alone. Outside of the standoff circle, tangent vectors provided the shortest distance to a standoff circle. Inside the standoff circle, no tangent lines exist and Lyapunov was used in its place. The two paths taken for Lyapunov and tangent vector fields outside the standoff circle are shown in Figure 2.14. TPLVF was later used for path planning to avoid obstacles in [47] while [48] constructed a tangent vector field for curved paths.

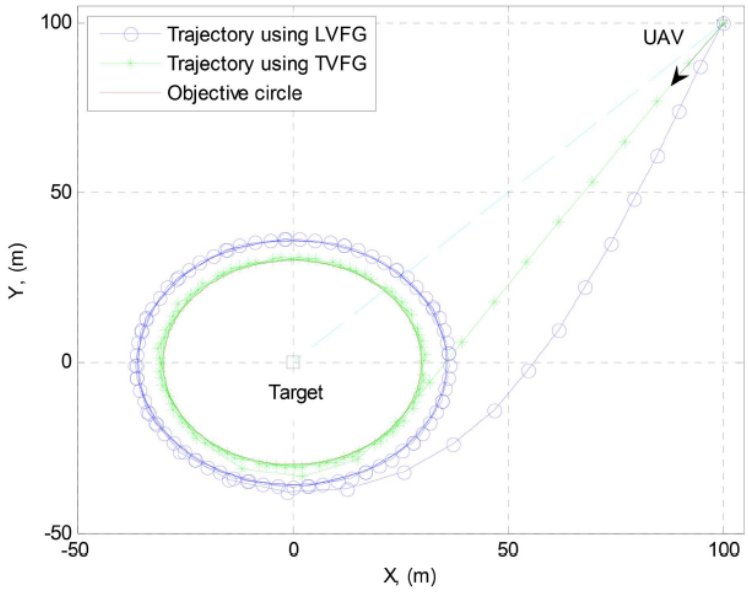


Figure 2.14: Tangent Plus Lyapunov Vector Fields for Shortest Path Target Tracking [47]

2.4.3 Path Planning Vector Fields

Another use of vector fields is a high level specification for heuristic path planning algorithms [49]. An optimal Rapid Random Trees (RRT*) algorithm used a vector field as a guide to explore the configuration space of the UAV for an obstacle free path. Branches extend from the root, or initial location of the UAV, randomly throughout the map with a finite deviation from the initial vector field. When a branch encounters an obstacle it is

trimmed and no longer explored. The path of minimum cost, or least distance, is selected for the UAV to use as a reference path. An example of the algorithm is shown in Figure 2.15.

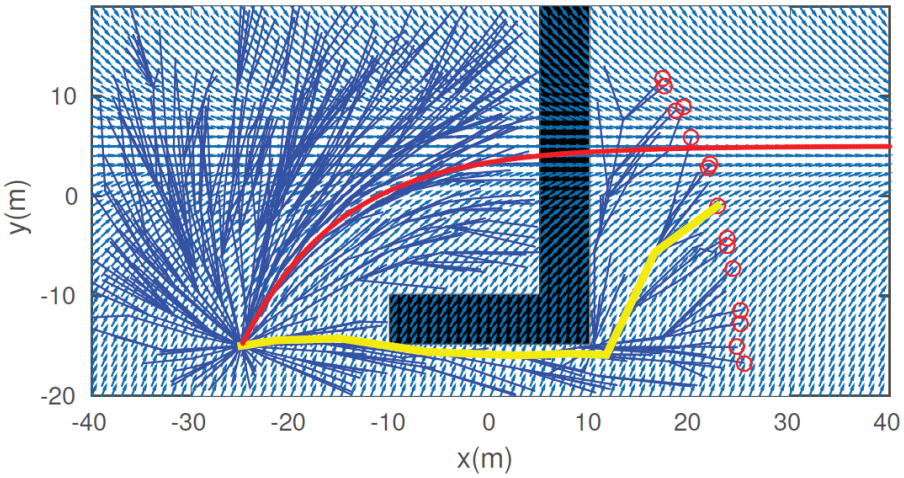


Figure 2.15: RRT* Path Planner with a VF Used as a Task Specification [49]

A VF was constructed inside a configuration space with edges defined by Delauny triangulation (DT) in [50]. A simulation of a robot traversing a vector field inside a set of DTs can be seen in Figure 2.16. Vector fields designed to stay inside a region of DTs may be used with optimal path planning algorithms for navigating urban environments [51].

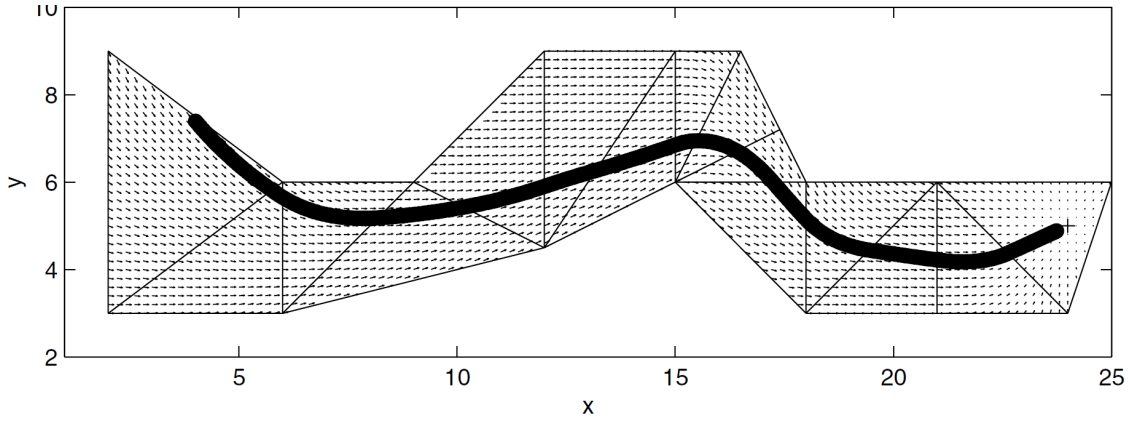


Figure 2.16: Vector Field Within a Set of Delaunay Triangles [50]

So far all of the vector field methods discussed have avoided obstacles by planning paths around them. Paths are typically calculated at the ground station and if communication is lost a new path may not be relayed to a UAV encountering a new obstacle. A possible solution is using vector fields to provide a repulsive force such as that seen in [20, 21, 52].

2.4.4 Gradient Vector Field

The Gradient Vector Field (GVF) method produces a similar field to LVF, however has several advantages. GVF produces an n-dimensional vector field that converges and circulates to both static and time varying paths [14]. Additionally, convergence, circulation, and time-varying terms that make up the GVF are decoupled from each other allowing for easy weighting of the total field [15]. GVFs converge and circulate at the intersection, or level set, of $n-1$ dimensional implicit surfaces ($\alpha_i : \mathbb{R}^n \rightarrow \mathbb{R} | i = 1, ..., n-1$). The integral lines of the field are guaranteed to converge and circulate the level set when two conditions are met: 1) the implicit surface functions are positive definite and 2) have bounded derivatives. Consider the space with dimensions in set $\mathbf{q}$:

$$\mathbf{q} = \left[x_1, x_2, ..., x_n \right] \tag{2.10}$$

The total vector field $\overrightarrow{V}$ is calculated by:

$$\overrightarrow{V} = G\nabla V + H \wedge_{i=1}^{n-1} \nabla_q \alpha_i - LM(\alpha)^{-1} a(\alpha) \tag{2.11}$$

or in component form:

$$\overrightarrow{V} = \overrightarrow{V}_{conv} + \overrightarrow{V}_{circ} + \overrightarrow{V}_{tv} \tag{2.12}$$

where $\overrightarrow{V}_{conv}$ produces vectors that converge to the path, $\overrightarrow{V}_{circ}$ produces vectors that circulate the path, and $\overrightarrow{V}_{tv}$ is a feed-forward term that produces vectors accounting for a time

varying path. The scalars G, H, and L weight convergence, circulation, and time varying components respectively.

Convergence is calculated by:

$$\overrightarrow{V}_{conv} = G\nabla V \tag{2.13}$$

where scalar G is multiplied by the gradient of the definite potential function V used in [15]:

$$V = -\sqrt{{\alpha_1}^2 + {\alpha_2}^2} \tag{2.14}$$

Circulation is calculated by taking the wedge product of the gradient:

$$\overrightarrow{V}_{circ} = \wedge_{i=1}^{n-1} \nabla\alpha_i \tag{2.15}$$

In the case of ($n = 3$) dimensions the wedge product simplifies as the cross product:

$$\overrightarrow{V}_{circ} = \nabla\alpha_1 \times \nabla\alpha_2 \tag{2.16}$$

The feed-forward time-varying component is calculated by:

$$\overrightarrow{V}_{tv} = M^{-1}a \tag{2.17}$$

where,

$$M = \begin{bmatrix} \nabla\alpha_1^T \\ \nabla\alpha_2^T \\ (\nabla\alpha_1 \times \nabla\alpha_2)^T \end{bmatrix} \tag{2.18}$$

$$a = \begin{bmatrix} \frac{\partial\alpha_1}{\partial t} & \frac{\partial\alpha_2}{\partial t} & 0 \end{bmatrix}^T \tag{2.19}$$

In [14–16] GVFs were constructed to control the velocity $\dot{q}$ of a holonomic robot by $\dot{q} = h$. Constant speed u was controlled by calculating the weighting scalars G, H, and

L that maintained the condition $||\dot{q}|| = u$. Other studies normalized the vector $\overrightarrow{V}$ and used it as a heading guidance while assuming velocity is held constant by the autopilot [21, 53]. Assuming velocity is controlled by a separate system frees up the vector field scalars to modify field behavior for other applications, such as specifying circulation direction, transition from convergence to circulation, and repulsion.

The standoff tracking and avoidance scenario presented in [21] used GVF as a heading guidance and negative GVF convergence weights to produce repulsive fields. A fixed UAV was tasked with loitering around a slow moving ground target while avoiding obstacles. A circular attractive time-varying vector field $\overrightarrow{V}_p$ was attached to a moving ground target and summed with repulsive obstacle vector fields $\overrightarrow{V}_o$ centered at the obstacles to provide the guidance $\overrightarrow{V}$ in Equation 2.20.

$$\overrightarrow{V} = \overrightarrow{V}_p + P\overrightarrow{V}_o \tag{2.20}$$

The strength of repulsive obstacle fields were weighted by the hyperbolic tangent decay function $P(d)$ bounded between $[0, 1]$ in Equation 2.21, where d is the range to the obstacle's center and R is the radius of at which the function P is near zero.

the decay, shown in Figure 2.17

$$P = -R\frac{tanh(2\pi d - \pi) - 2}{2} \tag{2.21}$$

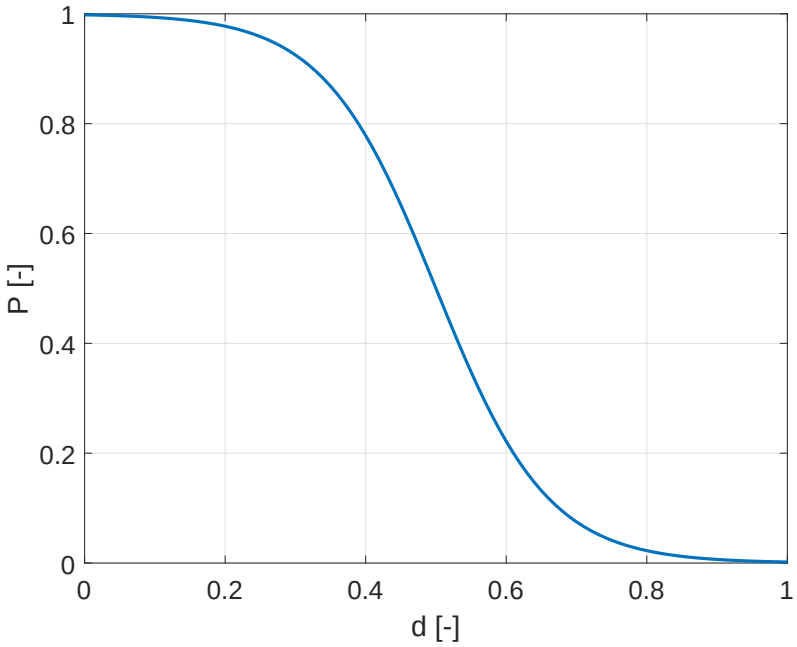


Figure 2.17: Tangent Hyperbolic Function for Obstacle Decay in [21]

The performance of LVF [21] and GVF [14–16] were compared for their cross track error with respect to the loiter circle in [21]. GVF was shown to have less cross track error compared to LVF due to the compensation for a time-varying vector field attached to the moving target. Avoidance was achieved without the need to re-plan mission paths. Additionally, the GVF avoidance guidance is an algebraic guidance that consists of summing an attractive path with obstacle fields that direct a UAV along the desired route while avoiding obstacles.

Summing attractive and repulsive vector fields may result in null guidance where the fields cancel, providing no guidance. The presence of singularities were mentioned briefly in [17] and observed in [20]. For fixed wing UAVs the lack of guidance may prevent the UAV from avoiding an obstacle, while multi-rotor UAVs may end up in a trap situation where they are guided to a singular point. Singularities may be present at any location where a goal field and obstacle field are of equal strength. Further, no optimization to

determine what the decay radius or circulation of an avoidance field should be for avoiding an obstacle with the least distance deviation from a planned path. Detecting singularities and modifying the GVF for an improved obstacle avoidance is the contribution of this research.

2.5 Literature Review Summary

UAVs are versatile tools that can be used for remote data collection without putting a pilot in harms way. Tasks are typically accomplished by following a series of waypoints that lie along an obstacle free and flyable pre-planned path. When unplanned or previously unknown obstacles are discovered along the UAVs planned path, a new obstacle free and flyable path may have to be generated which could be impossible if the ground station is beyond communications range. Methods such as potential field and vector field have been used to avoid obstacles without re-planning mission waypoints through the use of artificial attractive and repulsive forces. Potential field is not ideal for fixed wing UAVs since it converges to a singular point whereas LVFs and GVFs can converge and follow paths. GVFs allows for easy weighting of individual convergence and circulation terms making it convenient to modify field behavior without re-solving the Lyapunov equations. A summary of each guidance method discussed is shown in Table 2.1 below.

Table 2.1: Comparison of UAV Guidance Methods

Guidance Method	Avoids without re-plan	Returns to path	De-coupled convergence and circulation vectors	Time-varying
Waypoint		✓	N/A	N/A
VFF	✓			
LVF	✓	✓		
GVF	✓	✓	✓	✓

Summing paths and negating GVF convergence weight can be used for obstacle avoidance, however have so far been used only as high level specification of guidance behavior. Determining repulsive field weights and decay radius has not been optimized for obstacle avoidance. Additionally, characterization of vector field singularities in a summed field has yet to be addressed in literature due to most methods avoiding summation entirely. The contribution of this research is to characterize GVF singularities and provide an optimized decay radius and circulation weight for repulsive fields that minimize path deviation.

3 Methodology

3.1 Introduction to Methodology

Circular obstacle avoidance guidance without the need to re-plan a mission path was achieved by summing together a path following and obstacle avoidance vector field with optimized decay and circulation weights. Singularities in summed vector fields were defined and a method for numerically locating their position is presented in Phase I and demonstation of how adding circulation to repulsive GVFs may remove singularities from the UAVs path. Phase II investigated a method for selecting the combination of repulsive GVF decay radius and circulation weights that minimize a path deviation cost function. The optimized GVF guidance was implemented on an indoor multirotor UAV operating under fixed wing constraints in Phase III.

3.2 Phase I: Gradient Vector Field Singularity Detection

The objective of Phase I was to characterize and present a method of locating singularities in a summed gradient vector field. Phase I presents guidance equations for calculating GVFs that converge and follow a line path using the GVF method in literature [14–16]. Repulsive GVF for avoiding circular obstacles along the path is achieved by modifying a circular GVF's decay radius, convergence, and circulation weights. Summing the attractive and repulsive GVFs results in guidance that directs the UAV along the planned path while pushing away from the obstacle. Regions in the summed guidance where the path following and obstacle guidance directly oppose each other can result in vectors of zero length, called singularities. A method for identifying the location of singularities is presented along with a method for mitigating them.

3.2.1 Path Following Vector Field Guidance

Two vector fields were constructed consisting of a path following and repulsive obstacle field. Guidance for converging and following a time-invariant path using GVF guidance is achieved by summing together convergence $\overrightarrow{V}_{conv}$ and circulation $\overrightarrow{V}_{circ}$ terms that are multiplied by scalars G and H respectively, shown in Equation 2.12. The potential function V in Equation 2.14 decreases asymptotically to null when approaching the target path and therefore the convergence vector begins to decrease as well [14]. As the potential function decreases, the circulation term begins to dominate the total guidance, promoting path following. The distance from the target path at which the field begins to transition to circulation depends on the scalar weights G and H respectively. The total vector $\overrightarrow{V}_p$ represents the non-normalized attractive path following vector field comprised of both convergence and circulation terms. The shape of the target path that the vectors converge and follow depend on the specification of the implicit 3-dimensional surface functions α_1 and α_2. Intersecting two planes can be used to generate a GVF that converges and follows a straight path. The vertical plane, described in Equation 3.1, at angle δ with respect to the paths $y - axis$, is intersected with a horizontal plane at constant altitude z, Equation 3.2.

$$\alpha_1 = cos(\delta)x + sin(\delta)y \tag{3.1}$$

$$\alpha_2 = z \tag{3.2}$$

The gradient of the potential function, ∇V, for calculating the path following convergence term is shown in Equation 3.3.

$$\nabla V = \frac{-1}{2(\sqrt{\cos^2(\delta)x^2 + 2\cos(\delta)\sin(\delta)xy + \sin^2(\delta)y^2})} \begin{bmatrix} 2x\cos^2(\delta) + 2\cos(\delta)\sin(\delta)y \\ 2y\sin^2(\delta) + 2\cos(\delta)\sin(\delta)x \\ 2z \end{bmatrix} \tag{3.3}$$

Circulation is calculated by the cross product of the surface function gradients, which evaluates to that shown in Equation 3.4.

$$\overrightarrow{V}_{circ} = \begin{bmatrix} cos(\delta) \\ sin(\delta) \\ 0 \end{bmatrix} \tag{3.4}$$

The vector field $\overrightarrow{V}_p$ described will converge and circulate a straight path at the intersection of surfaces α_1 and α_2, depicted in Figure 3.1.

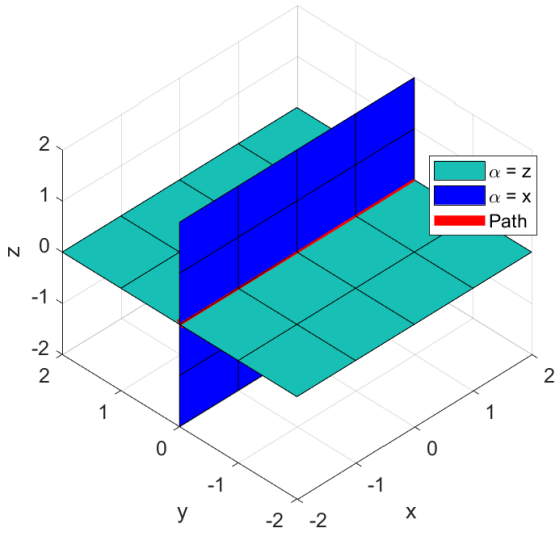


Figure 3.1: Intersection of Planes Defined by Implicit Surface Functions

Prior to using the path following guidance $\overrightarrow{V}_p$, it is normalized, denoted as $\overrightarrow{V}_{\|P\|}$, to have a magnitude $\|\overrightarrow{V}_p\| = 1$. The reason for normalizing the summed path following vector field is threefold. First, the vector is used as a heading controller only, therefore the angle of the vector is the information required by the autopilot. Second, the normalized vectors result in quiver plots with equal density arrows making the field easier to visualize. Lastly, normalizing the path following vector $\overrightarrow{V}_p$ fixes the length of the vector allowing for

prediction of singularity location after summing the field, which will be discussed in the next section. Before discussing singularities, the obstacle field is introduced.

$$\overrightarrow{V}_{\|P\|} = \frac{\overrightarrow{V}_p}{\|\overrightarrow{V}_p\|} \tag{3.5}$$

To produce guidance for following a path and avoiding an obstacle, a repulsive obstacle vector field $\overrightarrow{V}_{\|O\|}$ is constructed and summed with the normalized path following guidance $\overrightarrow{V}_{\|P\|}$. The repulsive vector field is multiplied by a decay function P which limits the influence of the obstacle to a finite range and will be discussed after the avoidance field equations are presented. The sum of the two normalized guidances with obstacle field decay is represented by $\overrightarrow{V}_g$ and is shown in Equation 3.6

$$\overrightarrow{V}_g = \overrightarrow{V}_{\|P\|} + P\overrightarrow{V}_{\|O\|} \tag{3.6}$$

Constructing the obstacle avoidance vector field $\overrightarrow{V}_{\|O\|}$ will now be discussed.

3.2.2 Constructing an Avoidance Vector Field

A circular avoidance vector field can be constructed in a way similar to that of the path following field in the previous section. What differentiates the obstacle field from the path following field is that the individual convergence $\overrightarrow{V}_{conv}$ and circulation $\overrightarrow{V}_{circ}$ components are normalized prior to summing to result in the obstacle field $\overrightarrow{V}_o$. The benefit of normalizing each field component before multiplying by their respective scalars G and H prior to summing is to produce an obstacle field with uniform behavior as distance from the obstacle increases. Normalizing each component allows both convergence and circulation terms to be present in the obstacle guidance at larger distances. Additionally, negative convergence weights will be used to produce vectors that diverge away from the path. The obstacle vector field is constructed using the normalized component Equation 3.7 with obstacle convergence and circulation weights G_o and H_o respectively.

$$\overrightarrow{V}_o = G_o \frac{\overrightarrow{V}_{conv_{obst}}}{\|\overrightarrow{V}_{conv_{obst}}\|} + H_o \frac{\overrightarrow{V}_{circ_{obst}}}{\|\overrightarrow{V}_{circ_{obst}}\|} \tag{3.7}$$

A circular avoidance vector field with radius r centered at (x_c, y_c) is constructed by intersecting a cylinder, Equation 3.8, and a plane Equation 3.2.

$$\alpha_3 = (x - x_c)^2 + (y - y_c)^2 - r^2 \tag{3.8}$$

Convergence is calculated by the gradient of the potential function in Equation 3.3, which when simplified evaluates to

$$\nabla V = A\overrightarrow{B} \tag{3.9}$$

where

$$A = \frac{-1}{\sqrt{\bar{x}^4 + \bar{y}^4 + 2\bar{x}^2\bar{y}^2 - 2r^2\bar{x}^2 - 2r^2\bar{y}^2 + r^2 + z^2}} \tag{3.10}$$

$$\overrightarrow{B} = \begin{bmatrix} 2\bar{x}^3 + 2\bar{x}\bar{y}^2 - 2r^2\bar{x} \\ 2\bar{y}^3 + 2\bar{x}^2\bar{y} - 2r^2\bar{y} \\ z \end{bmatrix} \tag{3.11}$$

and

$$\bar{x} = x - x_c \tag{3.12}$$

$$\bar{y} = y - y_c \tag{3.13}$$

Evaluating equation 3.9 results in a vector field that converges to a circular path. Circulation is calculated from the cross product of each implicit surface function's gradient, which simplifies to

$$\overrightarrow{V}_{circ_{obst}} = \begin{bmatrix} 2(y - y_c) \\ -2(x - x_c) \\ 0 \end{bmatrix} \tag{3.14}$$

The guidance $\overrightarrow{V}_o$ described by intersecting a cylinder and a plane can be shown in Figure 3.2.

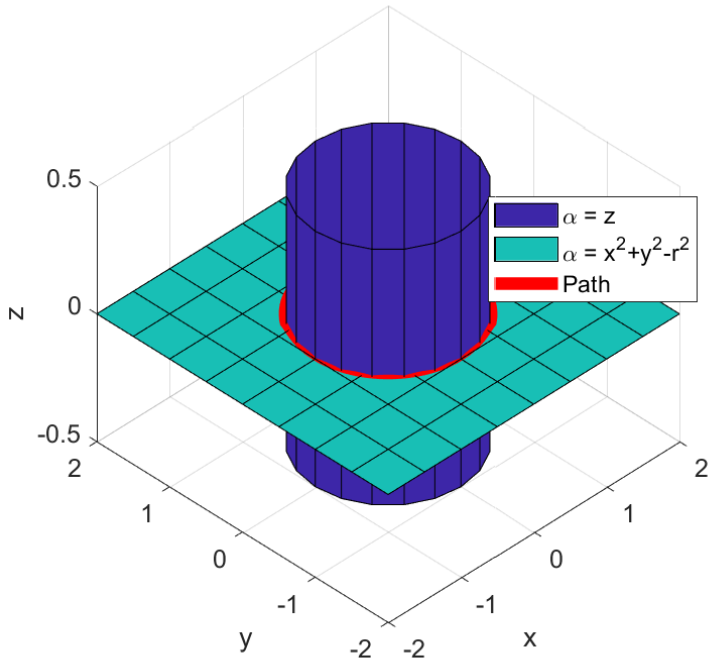


Figure 3.2: Intersection of a Cylinder and Plane Defined by Implicit Surface Functions

Obstacle fields should only act locally on the UAV guidance, which is accomplished by applying a decay function for a field of radius R. The decay strength P is determined in 3.15, where d is the Euclidean distance, or range, between the UAV and the center of the obstacle, shown in Equation 3.16. At a distance $d > R$ the decay strength P is effectively zero, having virtual no influence on the total guidance. At a distance $d \leq R$, the field strength is bounded between $[0, 2]$. The selection of the decay function P to be bounded as such is so that the obstacle field $\overrightarrow{V}_o$ eventually overpowers the path field $\overrightarrow{V}_{\|P\|}$.

$$P = -\tanh\left(\frac{2\pi d}{R} - \pi\right) + 1 \tag{3.15}$$

$$d = \sqrt{\bar{x}^2 + \bar{y}^2} \tag{3.16}$$

Prior to applying the decay function P the obstacle field $\overrightarrow{V}_o$ is normalized. Forcing the obstacle guidance $\overrightarrow{V}_{||o||}$ to have unity magnitude, bounds the decayed guidance magnitude $P\overrightarrow{V}_{||o||}$ to the interval $[0, 2]$ which allows a prediction of singularity location based on the range from obstacle center, d.

$$\overrightarrow{V}_{||o||} = \frac{\overrightarrow{V}_o}{||\overrightarrow{V}_o||} \tag{3.17}$$

3.2.3 Summed Guidance and Singularity Definition

The total summed guidance $\overrightarrow{V}_g$ defined in Equation 3.6 can now be calculated and visualized, Figure 3.3. In general, the GVF guidance is normalized for plotting, however, plotting the non-normalized guidance $\overrightarrow{V}_g$ demonstrates regions where the fields oppose each other and decrease the total fields magnitude. Guidance from summing a strictly repulsive field that is centered on a straight path, regions of both constructive and destructive summation occurs. Note the vectors on the positive x-axis increasing in length as the two fields come together. Conversely, vectors in the negative x-axis show decreasing length and in some areas disappearing entirely.

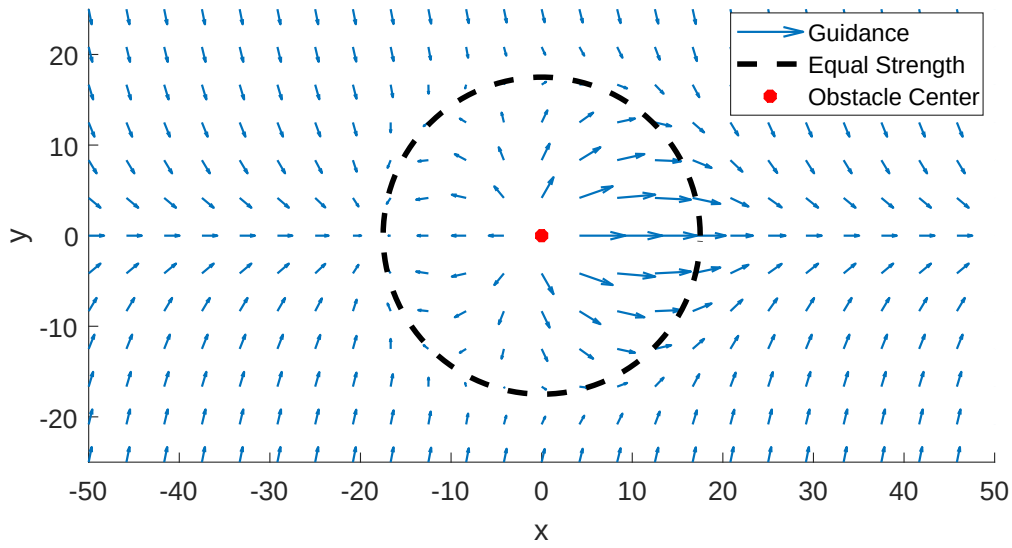


Figure 3.3: Summed Fields Without Total Normalization $\overrightarrow{V}_g$ with Null Circulation

Observing the regions around disappearing vectors it is shown that vectors appear to converge to a singular point from all directions, possibly indicating a trap situation. Visualizing the vector magnitudes of the scenario presented above can be accomplished with a summed field heat map, shown in Figure 3.4, which shows the regions of decreasing magnitude more clearly.

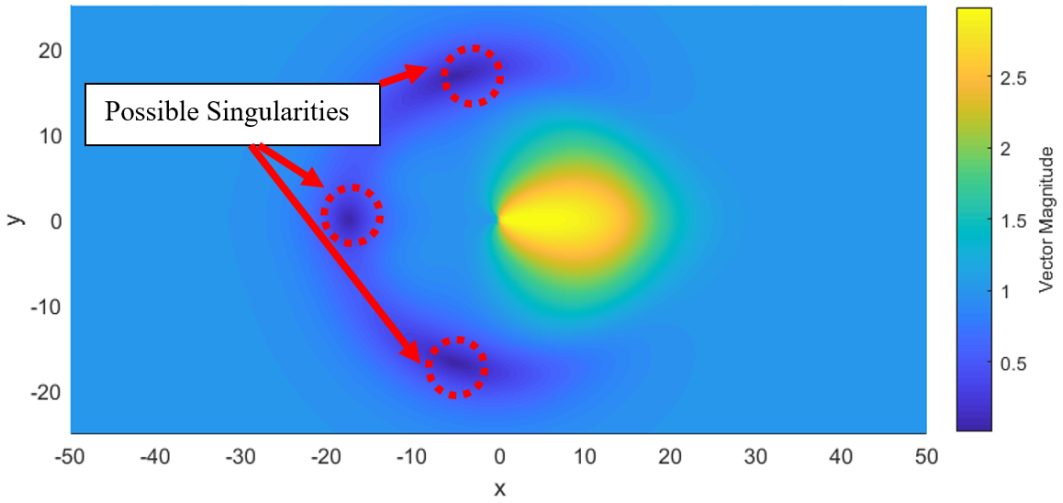


Figure 3.4: Summed Fields Without Total Normalization

If singularities indicate possible trap situations, it is important to detect and avoid them. Singularities in the vector field are defined as a region in the GVF space where the vector has zero magnitude, shown in Equation 3.18.

$$||\vec{V}_g|| = 0 \tag{3.18}$$

By extension, singularities are a result of a zero vector, shown in Equation 3.19 and Equation 3.20.

$$\vec{0} = \vec{V}_{||P||} + P\vec{V}_{||O||} \tag{3.19}$$

$$\vec{V}_{||P||} = -P\vec{V}_{||O||} \tag{3.20}$$

Vectors $\vec{V}_{||P||}$ and $\vec{V}_{||O||}$ are normalized, meaning that their magnitudes are of equal length $||\vec{V}_{||P||}|| = ||\vec{V}_{||O||}||$. For the condition shown in Equation 3.20 to be true for an obstacle field with a negative convergence weight $G = -1$, the decay function P must be unity.

Setting Equation 3.15 equal to 1, the distance at which the fields have equal strength, and therefore possible singularity locations, is determined to be that shown in Equation 3.21

$$d = \frac{R}{2} \tag{3.21}$$

Searching the GVF for locations that satisfy Equation 3.18 can be accomplished numerically, however a good initial condition is necessary. Initial conditions placed at the radius of equal strength, defined by Equation 3.21, on the side of the obstacle where deconstruction summation occurs the singularities can be found. Solving for the singularity locations with initial conditions placed evenly on the left hand side of the obstacle of the above scenario locates three singularities located on the radius of equation strength, shown in Figure 3.5.

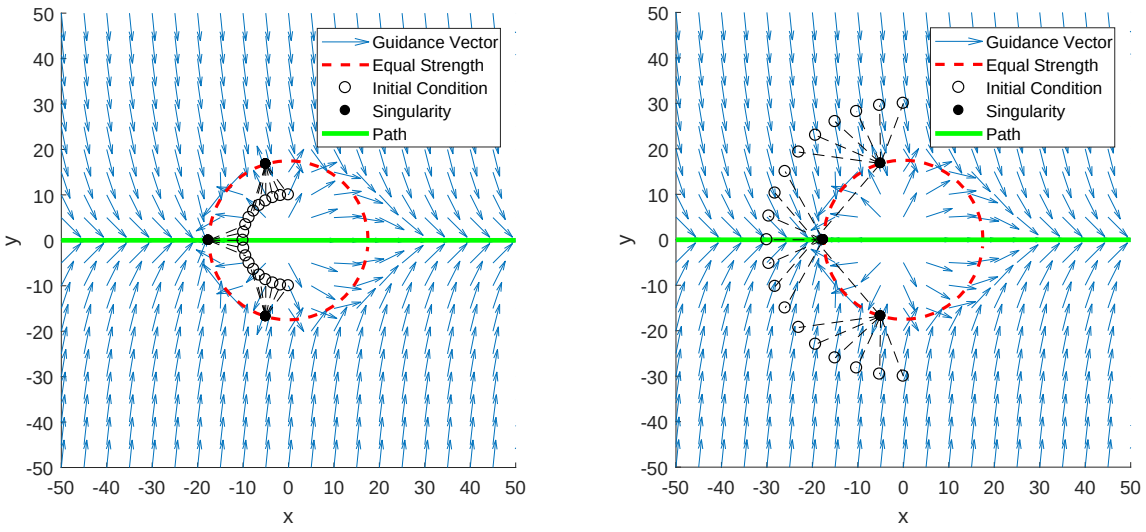


Figure 3.5: GVF Converging and Circulating a Circular Path

Singularities are caused by the two vector fields directly opposing each other, described in Equation 3.20. Adding circulation to the repulsive GVF may prevent

the summed vectors from canceling out completely. To demonstrate the prevention of vector cancellation, the above scenario is repeated with equal magnitude convergence and circulation components. Note that by adding circulation to the obstacle vector field singularities are prevented from forming along the route the UAV is likely to take, shown in Figure 3.6. Additionally, adding circulation provides a deterministic guidance on how to circumnavigate an obstacle. In literature, obstacles directed the UAV away and relied on the path following field for direction.

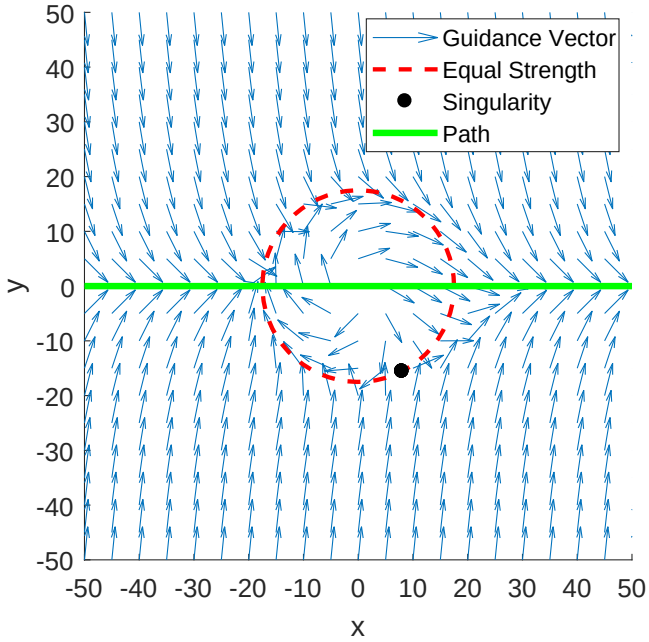


Figure 3.6: Singularity in Summed Field with Equal Magnitude Convergence and Circulation

3.3 Phase II: Optimization of Obstacle Field

The objective of Phase II was to determine a combination of GVF circulation and decay radius for an optimized circular obstacle avoidance. Demonstration of a UAV modeled as a Dubin's vehicle with maximum turn rate constraints converging and

following a straight path for various target path circulations was provided. A circular obstacle represented by a strictly repulsive GVF was added to the path and the avoidance observed. Obstacles radius and their associated GVF decay radius were represented in terms of the UAVs turning radius for convenience. Next, a path deviation cost function was described and was to minimized to provide an optimized GVF obstacle avoidance guidance. Reduction in path deviation cost was achieved for a worst-case obstacle avoidance scenario presented. Lastly, a method for solving circulation and decay radius numerically was presented.

3.3.1 Vehicle and Obstacle Definition

As discussed in literature and Phase I, GVF can be used to guide a UAV to converge and circulate a path. An example of a UAV traveling at constant speed $u = 20m/s$ and fixed turn rate $\dot{\theta} = 20deg/s$ converging and circulating a straight path using the GVF guidance described in Phase I is shown in Figure 3.7. Starting at an initial position of $(-45, 20)$ m and heading θ of 45°, the UAV is shown converging and following a straight GVF path with weights $G = 1, H = 5$.

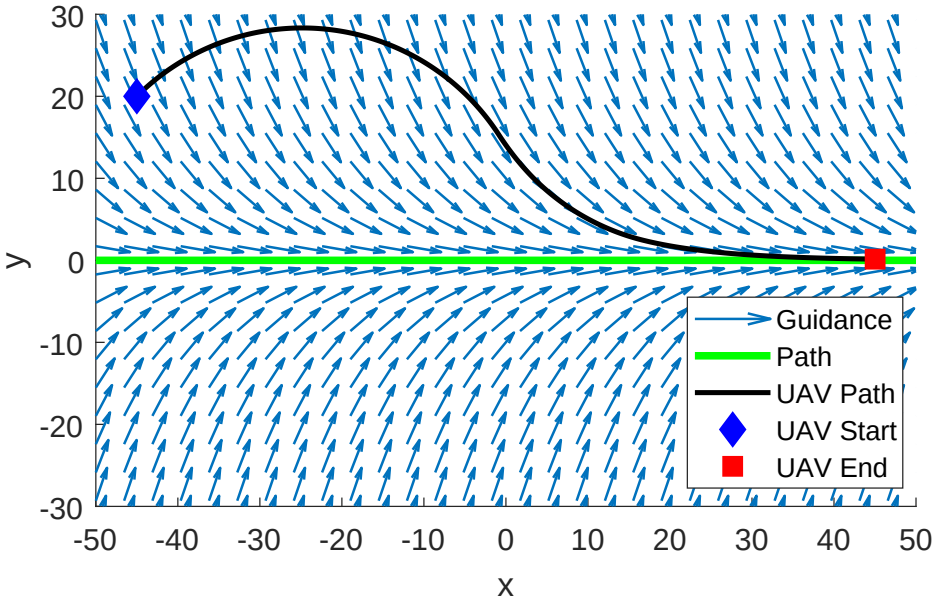


Figure 3.7: Fixed Wing Converging and Following a Path Simulation

The effect of modifying G and H has not been well documented in literature when the GVF is normalized for a heading guidance. Various path circulation weights for a fixed UAV velocity were conducted and the lateral error with respect to time recorded to determine an appropriate relationship between circulation and UAV velocity. Increased circulation results in the vector field transitioning to path following more quickly. Low circulation weights allows convergence to dominate closer to the path. The UAV's route for converging and following a path with $G = 1$ and multiple path circulation weights H is shown in Figure 3.8. Low circulation results in a UAV route that quickly approaches the path, however begins to oscillate and has a larger stead-state error compared to fields with larger circulation. The largest circulation field $H = 50$, does not oscillate, but takes considerably longer to converge. Circulation approximately the same value as the UAV's velocity results in a compromise between fast convergence and no oscillation. The lateral error from the path is shown in Figure 3.9.

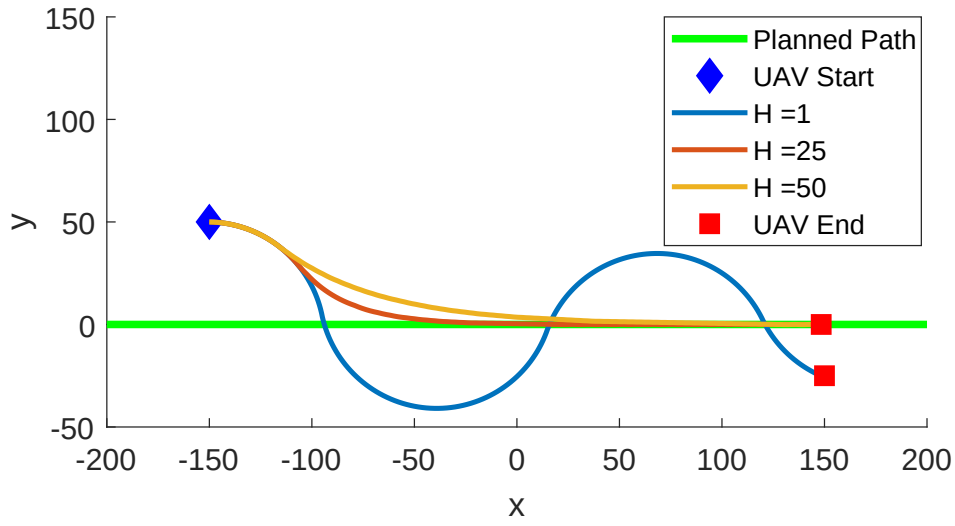


Figure 3.8: Fixed Wing Converging and Following a Path

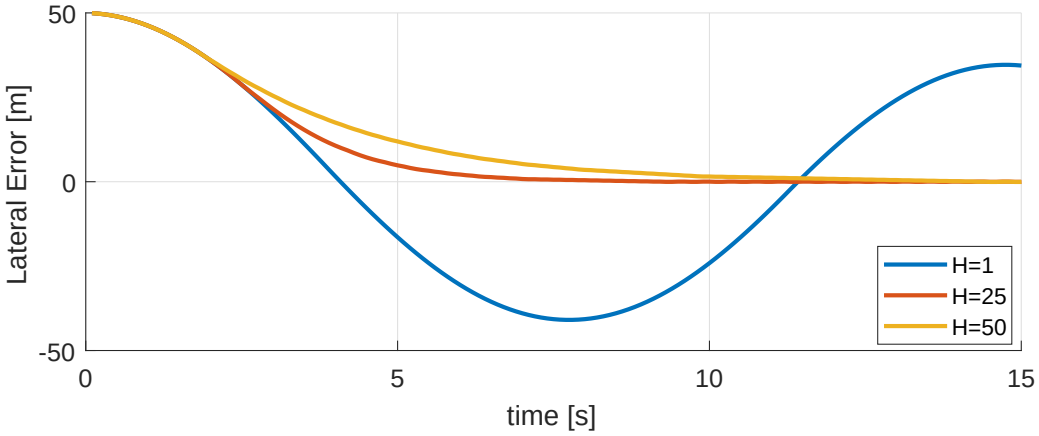


Figure 3.9: Lateral Error for Fixed Wing Guided by GVF Guidance of Multiple Circulations

From the above simulation it is observed that a circulation near the UAVs velocity provides a flight route that is a compromise between the high oscillation and high overshoot of a circulation $H = 1$ while not having a longer settling time with high circulation $H = 50$.

For future simulations the circulation of the path following guidance will be assumed to be equal to that of the UAVs velocity, $H = u$.

3.3.2 Obstacle on Planned Path

As the UAV travels the path using GVF guidance, an obstacle may be encountered that, without intervention, the UAV may collide with. Obstacles along the pre-planned path are described using two parameters, the obstacle radius r_o and the lateral distance from the path y_o. Representing the obstacle's position with respect to the path is convenient because it allows for a single optimized GVF solution to be applicable for multiple path angles δ.

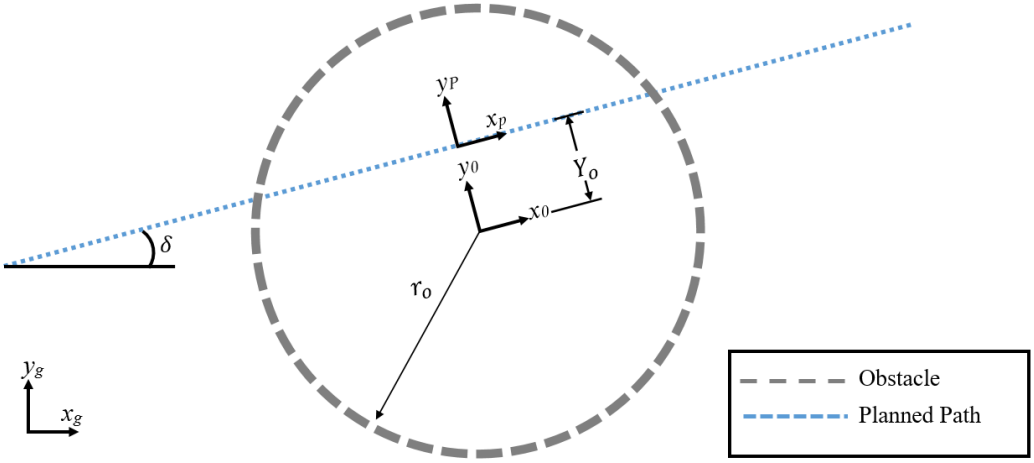


Figure 3.10: Circular Obstacle Along Planned Path

It is assumed here that the radius of the obstacle is no smaller than the turn radius of the UAV θ_r, which is calculated in Equation 3.22. The obstacles radius and lateral position can be expressed in terms of the UAVs turning radius by the ratios m and c shown in Equations 3.23 and 3.24 respectively. Representing the obstacle by these ratios is convenient because it allows for a single set of optimized parameters to be used for a UAV of multiple velocities.

$$\theta_r = \frac{u}{\dot{\theta}} \tag{3.22}$$

$$r_o = m\theta_r \tag{3.23}$$

$$Y_o = c\theta_r \tag{3.24}$$

The decay radius of the repulsive vector field, R, was defined in k multiples of the obstacle's radius, shown in Equation 3.25.

$$R = kr_o \tag{3.25}$$

A cost function can be used to measure the deviation from a planned path while avoiding an obstacle with GVF in Equation 3.26. The lateral distance of the UAV to the planned path was represented by y and r_o is the obstacle radius.

$$\gamma = \frac{1}{r_o}\int_0^{tf} ydt \tag{3.26}$$

Strictly repulsive GVF obstacle fields can provide avoidance, however may cause excess deviation from the pre-planned path, cause unnecessary turns, and flight routes near or through guidance singularities. A UAV traveling at a speed of $10m/s$ and a turn rate of $20deg/s$ is given a summed path following and obstacle avoidance guidance $\overrightarrow{V}_g$, defined in Equation 3.6 in Phase I. An obstacle centered on the path $y_o = 0$ and radius $r_0 = \theta_r$ is to be avoided. The decay radius multiplier k was increased manually over several simulations until avoidance was achieved. The flight path of the UAV with summed guidance is shown in Figure 3.11. The UAV experiences excessive path deviation and takes considerable time to settle back to the planned path. Additionally, the guidance directs the UAV to pass directly through a singularity and passes near an additional singularity towards the top of

the obstacle. The cost, calculated from Equation 3.26, for avoidance using the strictly repulsive guidance for head on collision scenario is $\gamma = 25$.

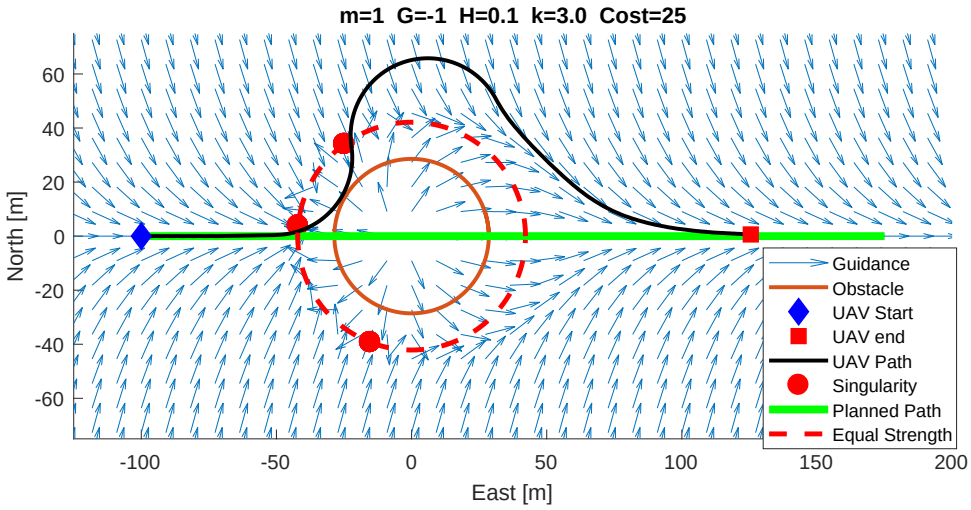


Figure 3.11: UAV Encountering a Circular Obstacle Centered on Pre-planned Path, No Circulation

Adding circulation to the obstacle guidance $\overrightarrow{V}_o$, as described in Phase I, may remove singularities from the UAVs route and reduce deviation from the planned path. Additionally, circulation adds deterministic information on how to circumnavigate an obstacle whereas strictly repulsive fields rely on the path following field to determine circumnavigation direction. Equal magnitudes convergence G_o and circulation H_o removes the guidance singularities from the UAV's path, adds a more gentle transition between fields shown in Figure 3.12. The cost of the head on collision avoidance using GVF guidance with obstacle circulation results in a cost of $\gamma = 7$.

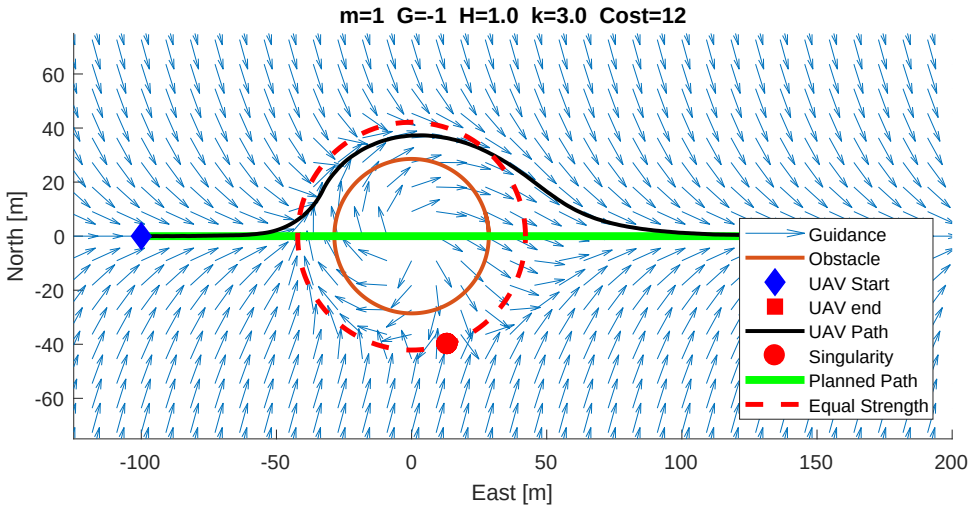


Figure 3.12: UAV Encountering a Circular Obstacle Centered on Pre-planned Path with Circulation

Adding equal magnitude parts G and H produce guidance with a 70% lower cost compared to strictly repulsive, however, k and H should be selected such that the cost is minimized. One method for selecting decay field radius and circulation parameters is to evaluate a large range of parameters and observe the cost for avoidance with each combination. The cost function shown in Equation 3.26 only penalized the UAV for path deviation, however should be modified to also be penalized for violating the obstacle space. The new cost function $\bar{\gamma}$ adds an additional piecewise function which increases the cost if the UAV is inside or on the obstacle's edge, shown in Equation 3.27

$$\bar{\gamma} = \frac{1}{r_o} \int_0^{tf} y dt + j(x, y) \tag{3.27}$$

$$j(x, y) = \begin{cases} 100dt & \sqrt{(x - xc)^2 + (y - yc)^2} \leq r_O \\ 0 & \sqrt{(x - xc)^2 + (y - yc)^2} > r_O \end{cases} \tag{3.28}$$

Using the above cost function $\bar{\gamma}$ in the same scenario presented in Figure 3.11, several simulations were conducted with identical UAV parameters and obstacle radius r_o

with varying k and H_o values. The cost of each simulation is shown in the heatmap in Figure 3.13.

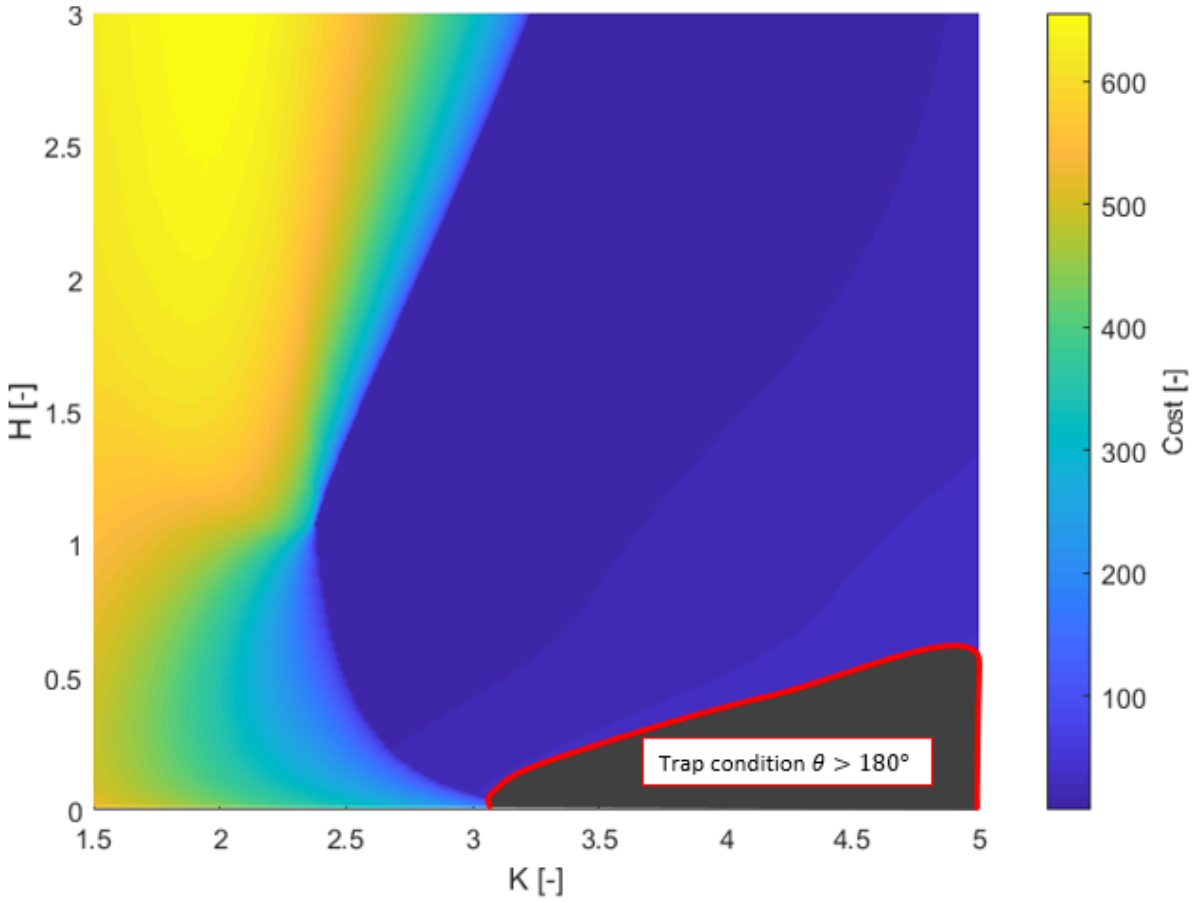


Figure 3.13: Heatmap of Cost as Function of k and H_o

Finding the minimum cost of the above heatmap and looking up the corresponding k and H_o values at which the minimum occurs can be used to provide an optimized avoidance. Solving the desired parameters for an optimal guidance using this search method for the general case is computationally expensive and can take many hundreds of simulations to evaluate the desired space. The problem becomes one of minimization of the cost function

and to find a solution numerically without solving for a large range of parameters, shown in Equation 3.29.

$$\underset{H,k}{\text{minimize}} \quad \frac{1}{r_o}\int_0^{tf} ydt + j(x,y) \tag{3.29}$$

A numerical solver which attempts to minimize a given cost function can be found in MATLAB called $fmincon()$. The minimizer operates on the following principle. Provided an initial condition array X_I, minimize the function $\bar{\gamma}$ by observing the change in $\bar{\gamma}$ within certain bounds. The reduction in cost as the minimizer runs is shown in Figure 3.14.

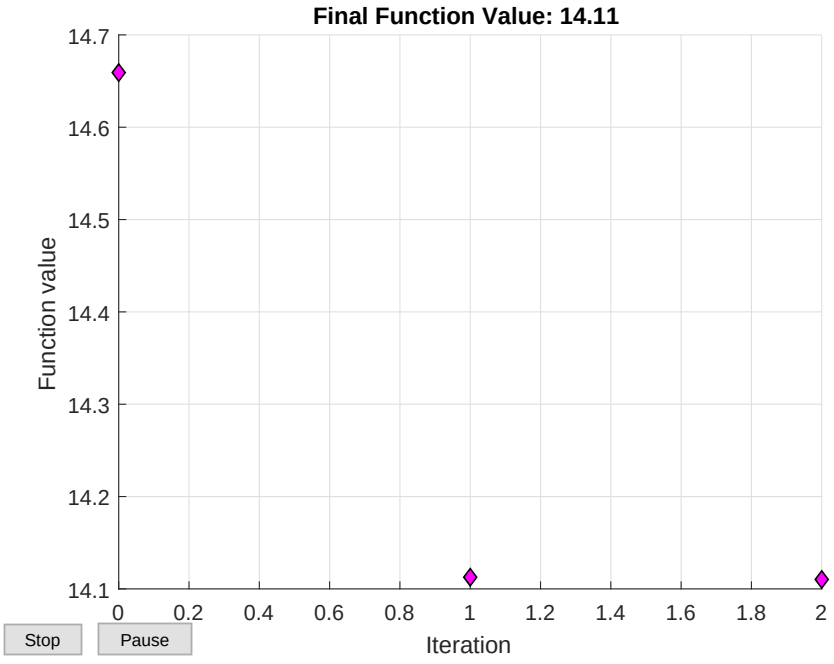


Figure 3.14: Cost Function Reduction in fmincon() MATLAB Function

The optimizer found a solution combination of $k = 2.7$ and $H_o = 1.8$ and results in a path with cost $\bar{\gamma} = 14$. The path can be seen below in Figure 3.15.

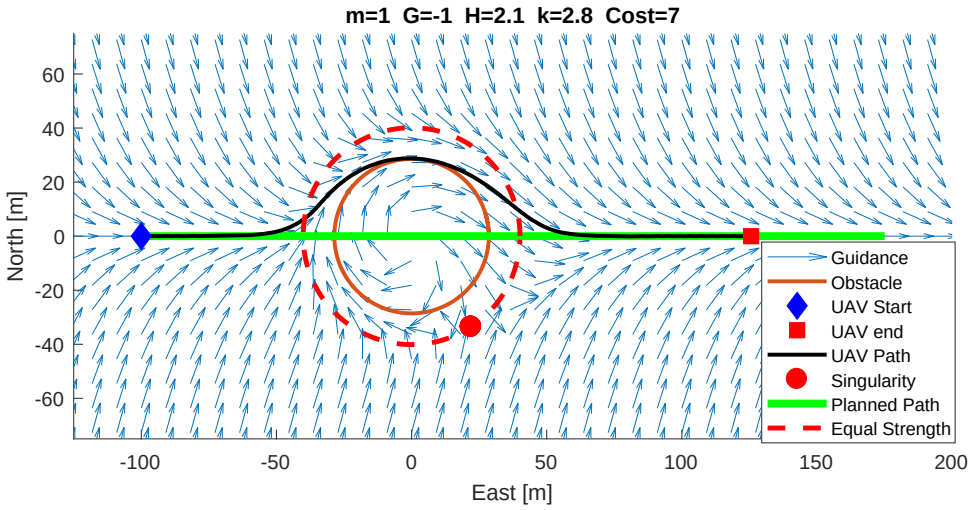


Figure 3.15: UAV Path From Optimized GVF

It was shown that avoidance can be achieved with a strictly repulsive GVF without the need to add circulation, however the simulated UAV encountered singularities along the route and had excess path deviation. Equal magnitudes convergence and circulation decreased the path deviation cost and no time was required to optimize the circulation weight. Searching the configuration space for a range of decay radius multipliers k and obstacle circulation weights H_o decreases the cost further compared to equal magnitude, however takes significant amount of time determine the cost for all combinations inside the configuration space. The optimized GVF guidance was shown to provide guidance with the same cost as the configuration space search with significantly reduced time from 10 hours to 5.2 seconds. All simulations were performed on an Intel i-7 7700k processor with 8Gb of ram.

Determining how GVF compares against other methods such as VFF, waypoint, and the optimal path around an obstacle is now discussed. A summary of the methods investigated for selecting GVF decay radius and circulation are shown in Table 3.1 below.

Table 3.1: Investigated Weighting Methods

GVF Method	Cost [-]	Solve Time
Strictly Repulsive	25	0
Equal Magnitude	12	0
Configuration Search	7	10.4 hours
Optimizer	7	5.2 seconds

3.3.3 Optimal Avoidance Route for Straight Path

A geometrically optimal route around a circular obstacle can be used to compare the performance of avoidance algorithms. The path for avoiding a circular obstacle while minimizing path deviation can be accomplished with three circular arc turns. The first and third arc utilize the UAVs minimum turning radius, θ_r, calculated in Equation 3.30. The start of the first minimum radius turn begins when the UAV's horizontal position x reaches $\tilde{x}$ from the path frame origin. At a horizontal position $-\hat{x}$ the UAV turns with a radius of the obstacle r_o and exits when the UAVs horizontal position reaches $\hat{x}$.

$$\theta_r = \frac{u}{\dot{\theta}} \tag{3.30}$$

The horizontal points $\tilde{x}$ and $\hat{x}$ are shown in Equations 3.31 and 3.32, respectively.

$$\widetilde{x} = -\sqrt{(\theta_r + R)^2 - (\theta_r - Y_o)^2} \tag{3.31}$$

$$\hat{x} = \frac{R\sqrt{(r + R)^2 - (\theta_r - Y_o)^2}}{R + \theta_r} \tag{3.32}$$

The avoidance path for navigating around a circular obstacle with minimum deviation from the planned path is defined in Equation 3.33 and shown in Figure 3.16.

$$y(x) = \begin{cases} \widetilde{y} - \sqrt{\theta_r^2 - (x - \widetilde{x})^2} & x < -\hat{x} \\ Y_o + \sqrt{R^2 - x^2} & -\hat{x} \leq x \geq \hat{x} \\ \widetilde{y} - \sqrt{\theta_r^2 - (x + \widetilde{x})^2} & x > -\hat{x} \end{cases} \tag{3.33}$$

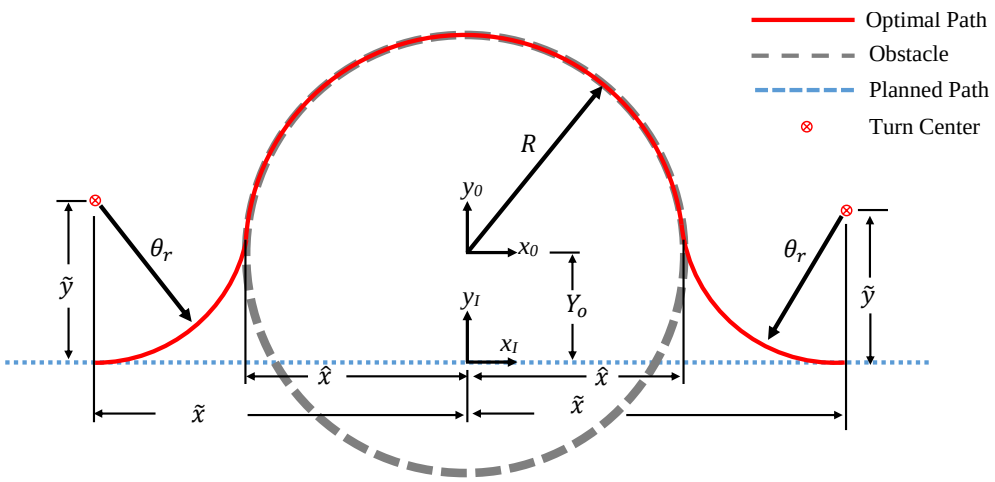
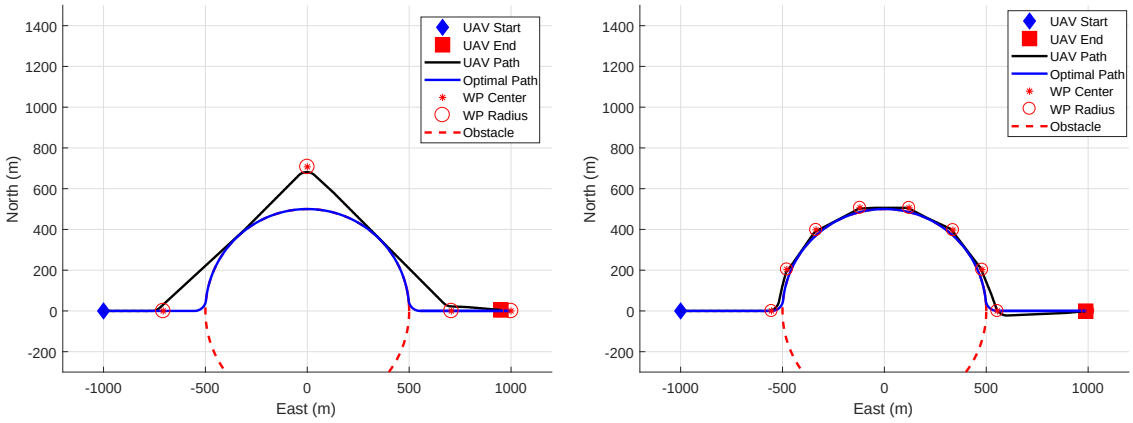


Figure 3.16: Optimal Kinematic Path Around Circular Obstacle

3.3.4 Avoidance Path and Waypoints

The number of waypoints that divert around an obstacle effects how closely the UAV tracks the outside of the obstacle and how much of the original path can be traveled. Few obstacle diversion waypoints leads to excess path deviation while increasing the number of diversion waypoints reduces path deviation, however has diminishing returns. The cost function in Equation 3.26 is used below to demonstrate how increasing the number of waypoints decreases the cost function, however, approaches an asymptote around six waypoints.



(a) (b)

Figure 3.17: Obstacle Diversion Waypoints

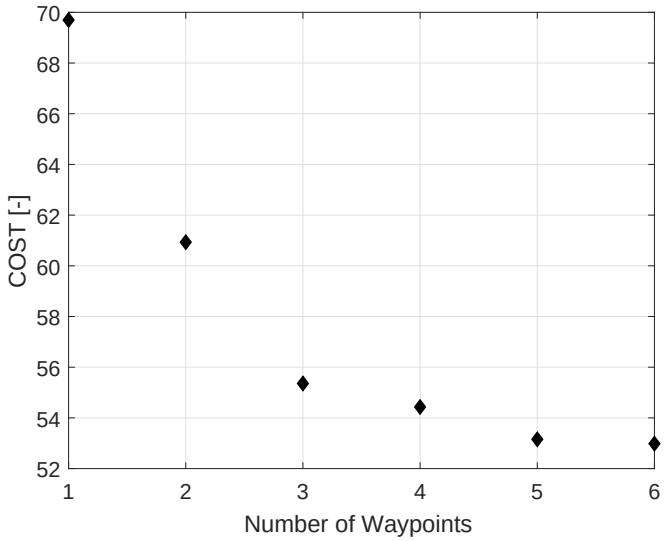


Figure 3.18: Cost Impact Versus Number of Waypoints

3.3.5 Circulation and Decay Lookup Tables

Solving for optimized circulation and decay radius during flight may be problematic since the time to reach a solution increases with the size of the obstacle. Optimized GVF

circulation and decay radius could be selected from a pre-computed lookup table in real time, removing the need to calculate the optimized parameters during flight. Lookup tables were generated by determining the optimized circulation and decay radius for a range of possible obstacle configurations. The obstacle's radius is again described in positive multiples of the UAVs turning radius, described in Equation 3.23. It is also convenient to describe the obstacles lateral position to the path in terms of the UAV's turning radius by the multiple c shown in Equation 3.34.

$$Y_o = c\theta_r \tag{3.34}$$

Obstacle configurations were evaluated for a range of radii m on the interval $[1, 5]$ with a step size of 0.5 and lateral positions c with an step size of 0.1. The UAV was assumed to travel at $10m/s$ with a maximum turn rate of $20deg/s$ with a time step dt of 0.01 seconds. Lookup tables for obstacle field circulation H_o and decay radius k are shown in the heatmaps in Figures 3.19 and 3.20 respectively.

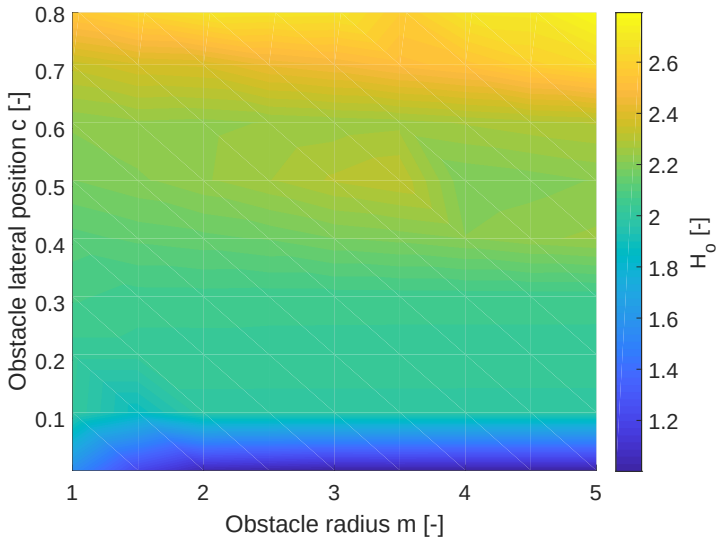


Figure 3.19: Obstacle Circulation Lookup Table, Smoothed by Interpolation

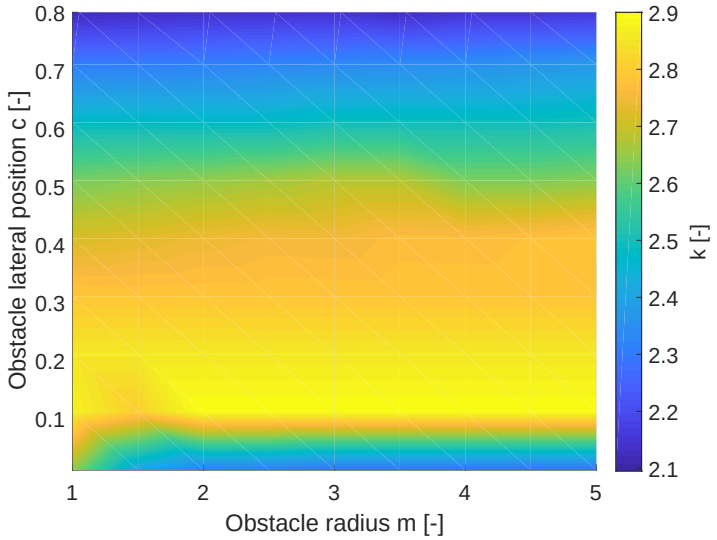


Figure 3.20: Obstacle Decay Radius Lookup Table, Smoothed by Interpolation

For obstacle configurations that were not optimized, circulation and decay radius may be determined by interpolation. To determine how the performance of interpolated solutions compare against an optimized solution, interpolation points were selected in a grid pattern between the tabulated values and the cost of each solution recorded. The interpolated points that were evaluated are shown in Figures 3.21 and 3.22 below.

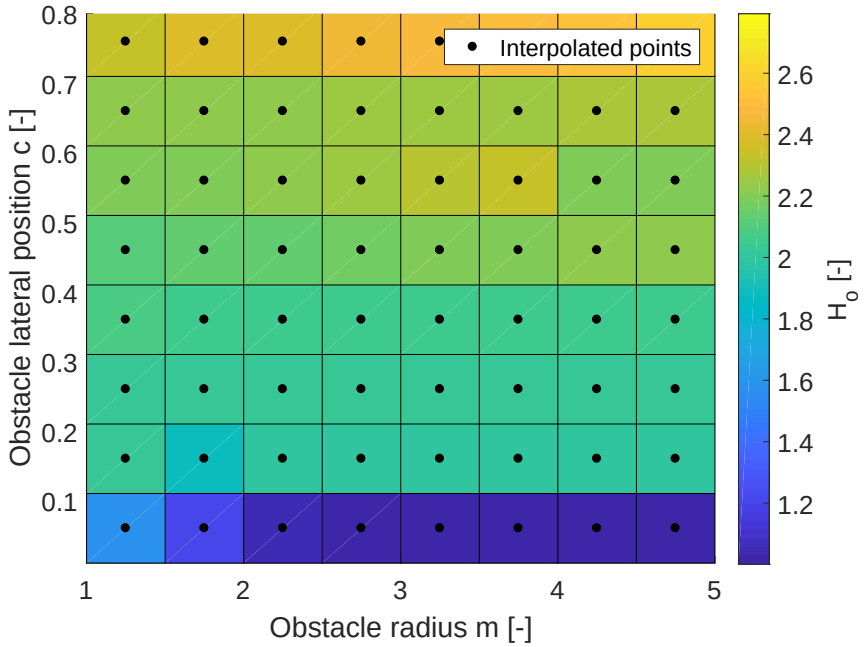


Figure 3.21: Obstacle Circulation Lookup Table with Interpolation Points

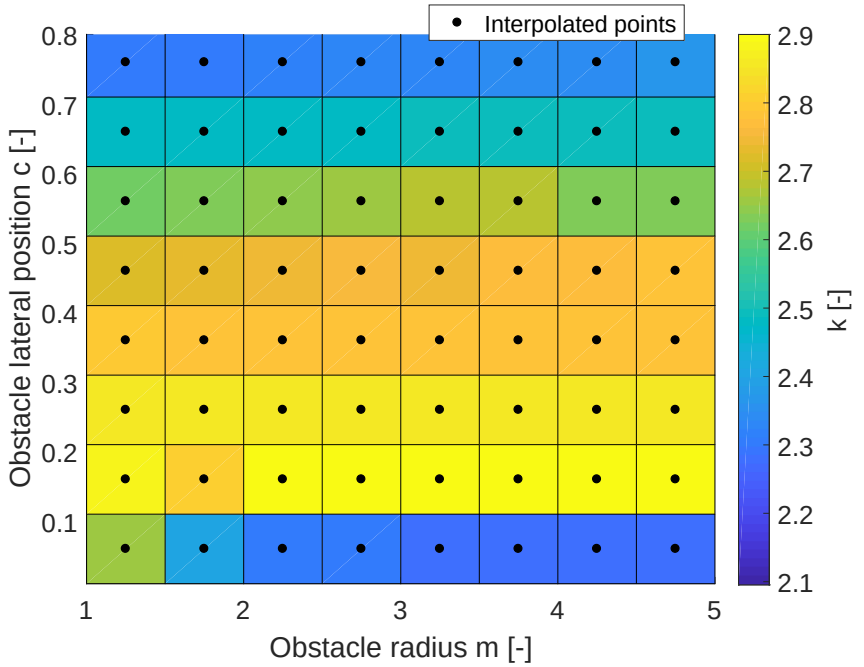


Figure 3.22: Obstacle Decay Radius Lookup Table with Interpolation Points

The cost of optimized versus the cost of interpolated solutions are represented as two planes overlaid in Figure 3.23 below. The interpolated solution resembles the same behavior as the optimized solution, and for the configuration evaluated has a mean square error of 2.5. Expressing the obstacle's radius and lateral position in terms of positive multipliers of the UAV's turning radius allows a single lookup table to be used for multiple UAV speeds, resulting in less scenarios to be evaluated.

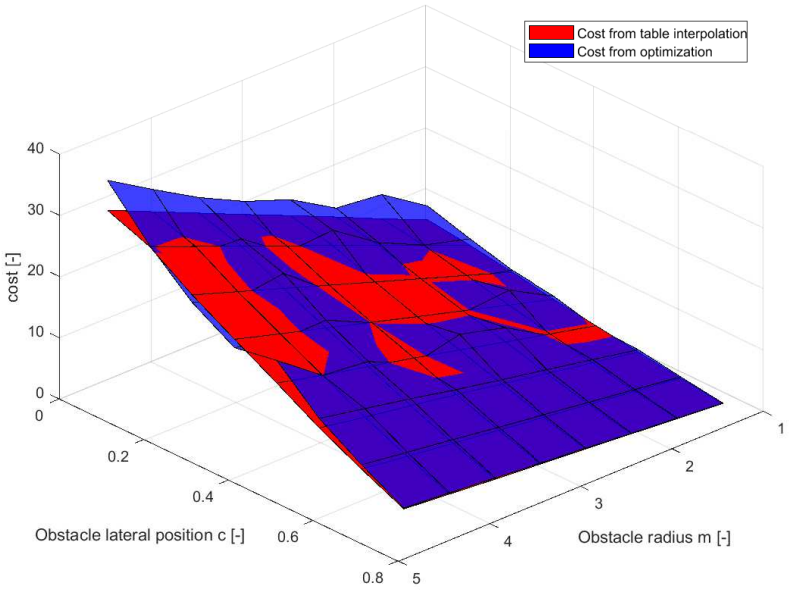


Figure 3.23: Cost of UAV Obstacle Avoidance Using Interpolated and Optimized Parameters

GVF guidance provided a low cost avoidance that quickly returns to the planned path, however the time to reach a solution using a numerical solver for the optimized set k and H_o increases as the size of the obstacle increases. Failure to reach a solution in adequate time may result in the UAV violating the no-fly zone or colliding with the obstacle. Lookup tables were constructed for both circulation weight H_o and obstacle decay radius k that were be referenced with minimal computation time compared to a single

numerical optimization solution. The obstacles configuration, radius and lateral position, were non-dimensionalized by expressing said parameters in multiplicative factors of the UAV's turning radius. Non-dimensionalizing the obstacles configuration allows for a single lookup table to be used for multiple UAV velocities.

3.4 Phase III: Flight tests

The objective of Phase III was to demonstrate the optimized gradient vector field guidance presented in Phase II on multirotor UAV flying with fixed wing turn-rate constraints. An overview of the experimental setup will be given for both the hardware and software that was implemented to achieve GVF guidance on a Crazyflie quadcopter. Python and MATLAB GVF guidances are compared to confirm that they are equivalent.

3.4.1 Experimental Overview

All of the scenarios discussed using GVF guidance have involved simulating a fixed wing UAV modeled as a Dubin's vehicle. The micro quadcopter shown in Figure 3.24 was used in place of a fixed wing UAV for experimental flight tests. There are several reasons why using an indoor quadcopter is advantageous for experimental flight tests. First, finding an airspace to safely test the guidance system with an adequate clearing for takeoff and landing can be difficult. Many environmental complications such as high winds, precipitation, and low visibility could delay or prevent flight tests all together. Flying indoors allows for more repeatable experimentation, environmental control, and use of high speed motion capture systems to provide position information to guidance and control systems without the complications of outdoor flight.

The micro quadcopter, Crazyflie 2.0, designed by Bitcraze was selected as the experimental flight vehicle due to its low cost and the ability to send the vehicle roll angle, pitch angle, yaw rate, and thrust percentage commands directly over radio. The control

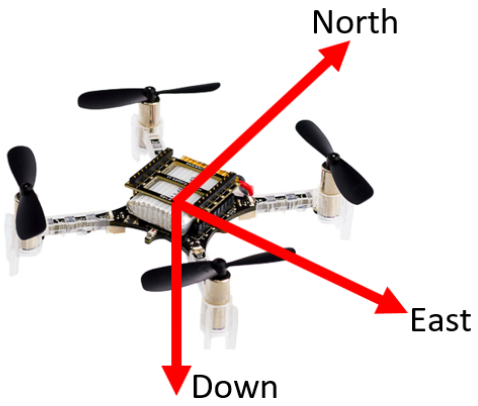


Figure 3.24: Micro Quadcopter Crazyflie 2.0 by Bitcraze

messages can be sent through the object oriented and scripted language Python, a language with syntax very similar to MATLAB. The UAV was viewed by 8 vicon vero cameras that detect infrared light reflected by small markers placed on the vehicle. Video captured by the cameras at 100Hz is sent to a PC with a software package that estimates the pose of the vehicle and sends that information over a local area network (LAN) to a ground station PC where guidance and control calculations are made. The command messages are then sent over radio to the Crazyflie where an on-board controller accepts the commands and outputs the necessary motor output to achieve the commands. A high level overview of the experimental framework described is shown in Figure 3.25. Each component of the framework will be discussed in more detail in the proceeding sections.

3.4.2 Crazyflie 2.0

The crazieflie 2.0 is a lightweight micro-quadcopter UAV weighing in at 27 grams and has an approximate payload capacity of 10 grams. On-board flight controller maintains vehicle stability by estimating it's attitude and making corrections to the four brushed motors. The software package, cfClient, interfaces with the Crazyflie to accept and transmits these control messages over radio.

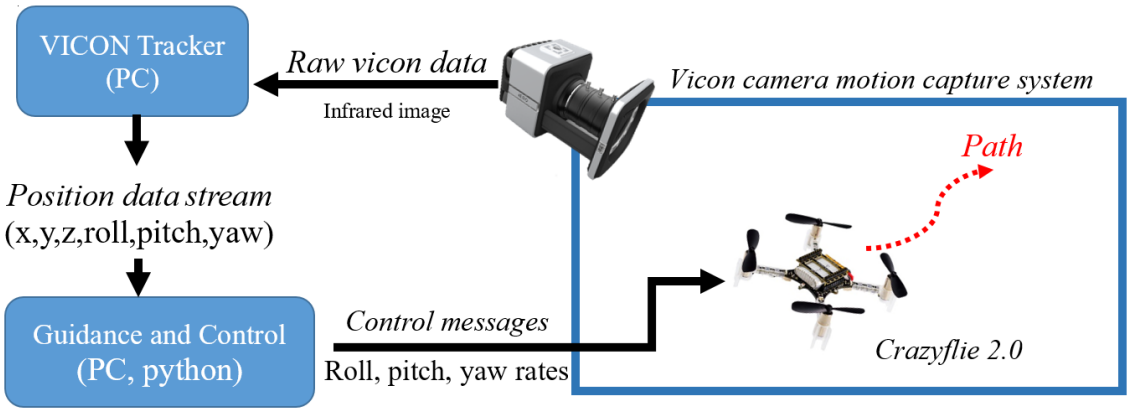


Figure 3.25: Indoor Quadcopter Flight Experimental Layout

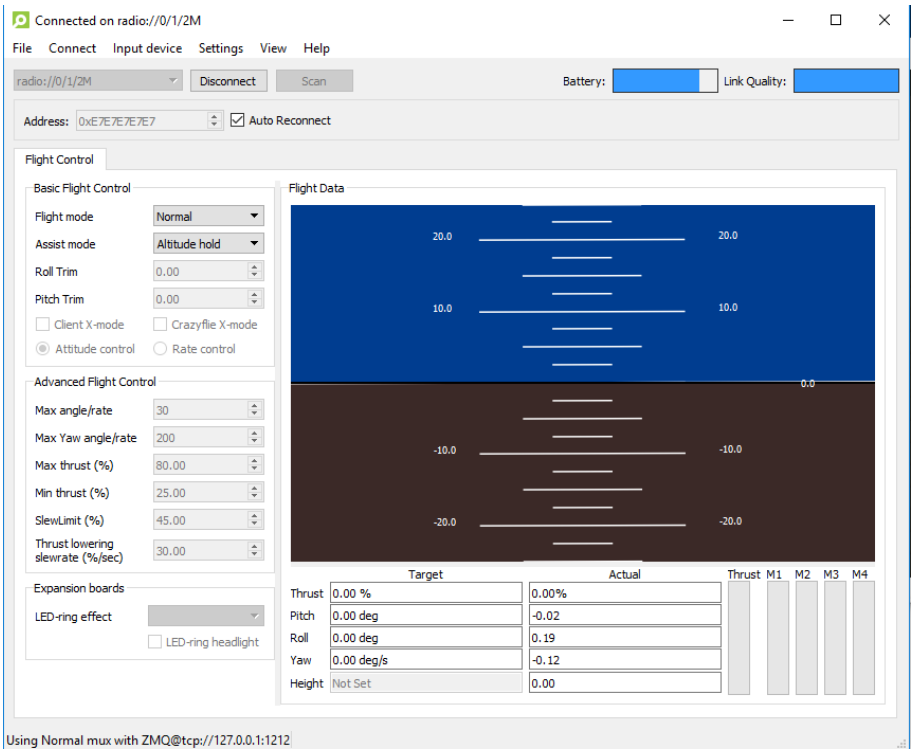


Figure 3.26: Crazieflie Client Radio Communication Software

The guidance and control software that calculates these command messages was hosted on a remote machine and not on-board the UAV, primarily for the convenience of

fast development. The guidance and control was written in Python, which is highly portable and could easily be implemented on-board a UAV.

3.4.3 Python Guidance and Control Ground Station

The guidance and control framework developed for experimentation resembles that described in Figure 2.3. Navigation script taps into a stream of data provided by the Vicon tracker software and distributes that data to guidance and control algorithms. Position (x, y, z) and yaw ϕ are provided to the control algorithm which consists of four PID controllers. The UAV was set to fly at constant altitude for all simulations and experimentation, therefore only planar position (x, y) were provided to the guidance system. Heading guidance from the optimized GVF was converted to a carrot located at a position (xc, yc) is sent to the control system to be used as set-points. Control messages are relayed to a radio interface software, cfClient, which communicates the control messages with the Crazyflie. An overview of the described system is shown in Figure 3.27.

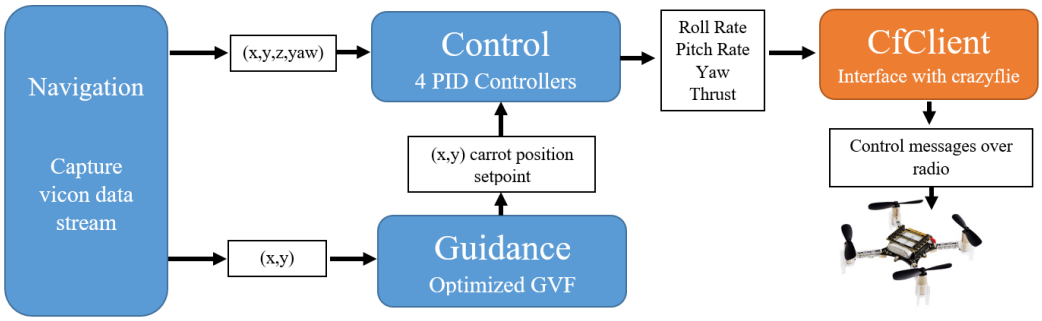


Figure 3.27: Crazyflie Guidance and Control Software Framework

The control algorithm consists of four PID controllers which are used to calculate roll rate, pitch rate, yaw, and thrust to drive the UAV to a desired setpoint. The error is measured by subtracting the measured state (x, y, z, yaw) from the desired setpoint. The

block diagram of the PID controller is shown in Figure 3.28. Gains P, I, and D for each controller is tabulated in Table 3.2 below.

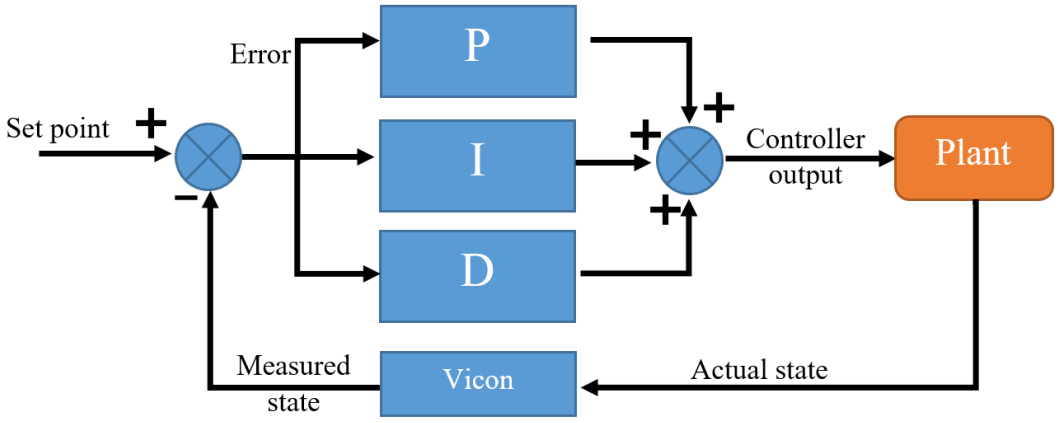


Figure 3.28: Crazyflie Guidance and Control Software Framework

Table 3.2: Tuned PID Gains for Roll, Pitch, Yaw Rate, and Thrust Controller

Control Parameter	P	I	D
Roll Angle	29	2.5	19
Pitch Angle	29	2.5	19
Yaw Rate	80	50	30
Thrust Percentage	100	90	70

Applying the Dubin's fixed turn rate constraints was accomplished by saturation control. The current heading of the UAV, θ was determined from the velocity vector of the Crazyflie. The commanded heading from the optimized GVF was limited such that the change in heading was no greater than $\dot{\theta}$, in this case 20 degrees per second.

3.5 Python Guidance Validation

The optimized GVF guidance developed in MATLAB in Phases I and II was programmed into Python and compared under several scenarios to ensure that the guidance produced were identical. First, a path following GVF was calculated in Python for a straight

line and results overlaid with an identical scenario in MATLAB. The quiver plots show guidance calculated by MATLAB aligning with the guidance calculated in Python shown in Figure 3.29.

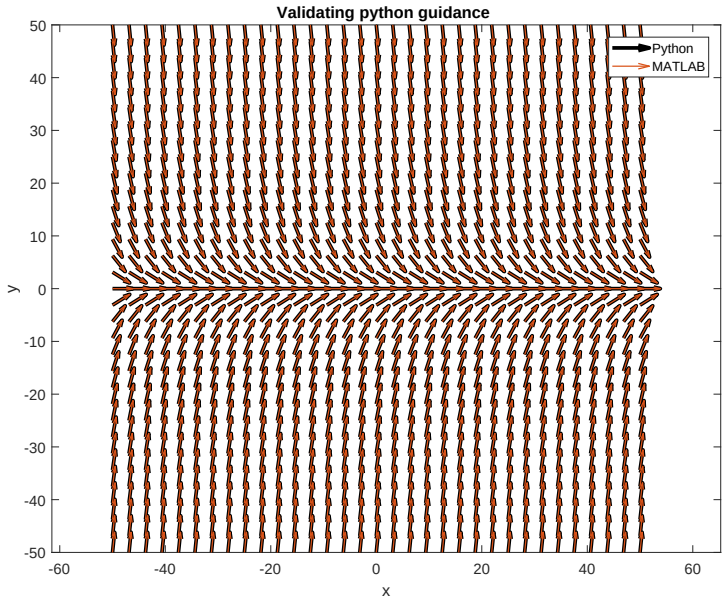


Figure 3.29: Validation of Python Straight Path Guidance Overlaid with MATLAB

Summing an avoidance path with the path following field with no circulation results in the guidance shown in Figure 3.30.

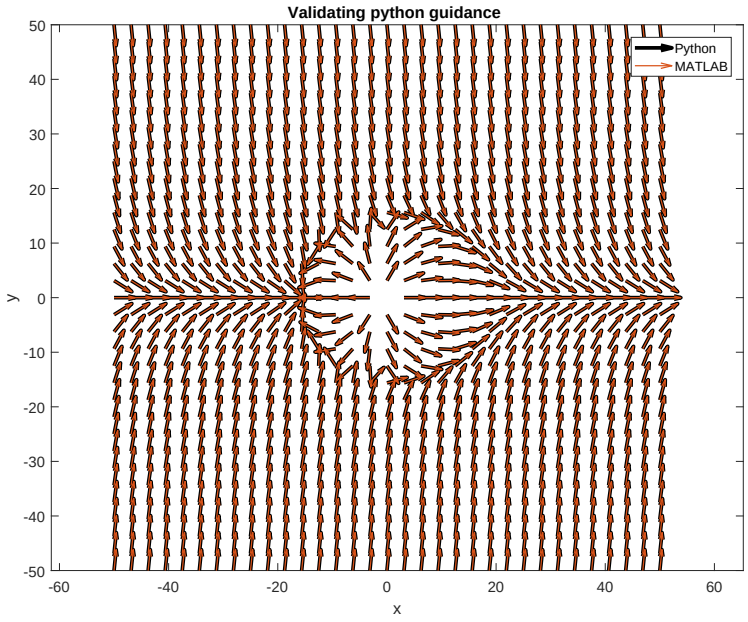


Figure 3.30: Validation of Python Summed Guidance Overlaid with MATLAB

Lastly, the Python guidance was simulated with an identical Dubins vehicle as MATLAB and shown to have identical paths in Figure 3.31

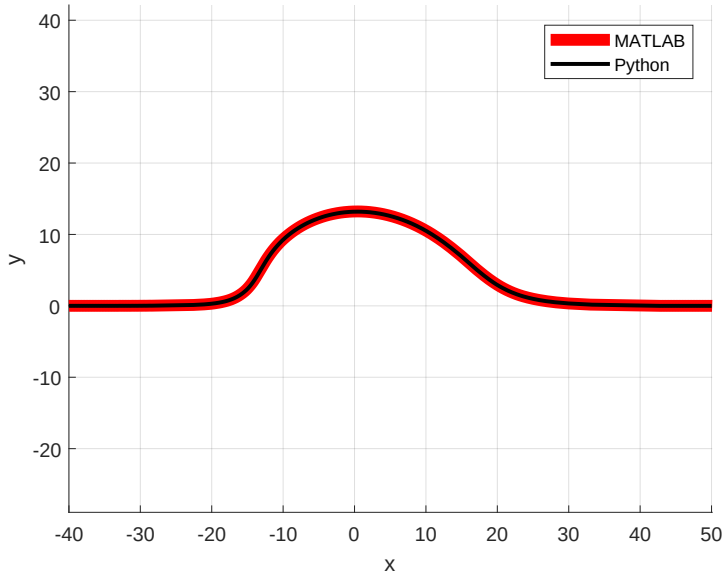


Figure 3.31: Validation of Python Dubins UAV Route Overlaid with MATLAB

Additional validation quiver plots with the obstacle field, decay field, and circulation in a obstacle field can be found in the appendix.

3.6 Summary of Methodology

Equations for path following and obstacle vector field guidance were presented. Singularities in a summed field were discussed and a method for locating them numerically was provided. The fixed wing UAV modeled as a Dubin's vehicle with fixed turn rate constraint was guided by a path following guidance to converge and follow a straight line. Obstacles were defined in terms of UAV turning radius for convenience and the obstacle field summed with path following field. Cost functions for evaluating the performance of obstacle avoidance in terms of path deviation was provided. Paths with strictly repulsive guidance were shown to guide the UAV away from the obstacle but with excess path deviation, unnecessary turns, and slow path convergence. Adding circulation to the obstacle field improved performance and reduced the overall cost. A large space of circulation and decay multipliers were evaluated and the cost displayed in a heatmap. The combination of parameters that provide the least cost guidance can be selected to provide an optimized guidance, however takes significant computation time to evaluate the large parameter space. A numerical method for determining parameters was presented and an example of it's execution provided. Lastly, the optimized GVF guidance was implemented into python to be used for real-time obstacle avoidance guidance for the Crazyflie quadcopter. Next, results for each phase will be discussed.

4 Results

4.1 Introduction to Results

The singularity detection method discussed in Phase I and optimized path following and obstacle avoidance guidance presented in Phase II were demonstrated in four scenarios. Both path centered and off-centered obstacles of various radii are evaluated with a simulated fixed wing UAV. The optimized GVF circulation H_o, decay radius multiplier k, and path deviation cost $\bar{\gamma}$ is documented for each scenario. The optimized GVF guidance implemented on a Crazyflie quadcopter is then used in four scenarios consisting of centered, off-centered, and various radii obstacles. Path deviation cost for experimental flights and simulation are compared for their percent difference in cost.

4.2 Phase I & II

The methods discussed in Phase I and Phase II for detecting singularities in a summed GVF and optimizing obstacle field decay radius and circulation will now be demonstrated for several scenarios involving a Dubin's vehicle modeling a fixed wing with turn rate constraints. Avoidance scenarios that represent the possible configurations of an obstacle that lies along a pre-planned path consist of small obstacles, large obstacles, path centered obstacles, and path offset obstacles. small obstacles are considered those with radius $r_o = \theta_r$ while large obstacles have a radius $r_o \geq 5\theta_r$. Centered obstacles represent the worst case avoidance scenario where the UAV must deviate at least 50% of the obstacle's radius in order to successfully avoid. Counterclockwise and clockwise avoidance have identical costs for the path centered obstacle since the routes determined from the optimizer are symmetric. In the centered obstacle scenarios presented, it is assumed that clockwise avoidance is desired and the initial conditions and bounds were set accordingly. Table 4.1 lists four avoidance avoidance scenarios that were evaluated.

Table 4.1: GVF Avoidance Scenarios

Scenario Number	Obstacle radius multiplier m	Obstacle lateral position y_o
1	1	0
2	5	0
3	1	$0.5r_o$
4	5	$0.5r_o$

4.3 Worst Case Avoidance Scenario

A worst case avoidance scenario will be used to compare the optimized GVF with waypoint, VFF, and the optimal path with respect to the path deviation cost function. A circular obstacle centered on the path, $y_o = 0$, requires a deviation from the path of at least 50% of the obstacles radius. A fixed wing UAV at an initial position $(-400, 0)$ and heading $\theta = 0^\circ$ follows the straight path connecting the points $(-400, 0)$ and $(400, 0)$ respectively. Traveling at a constant speed $u = 25m/s$ and with a fixed turn rate of $\dot{\theta} = 20deg/s$ the UAV must avoid an obstacle with radius $2\theta_r$ located at the origin $(0, 0)$. The VFF guidance from [37] is used with an obstacle window radius of $\theta_r + r_o$, a cell repulsion $Fr = -3$, attraction force $F_t = 0.8$, range exponent $n = 2$, and a goal located at $(700, 0)$. For LOS waypoint guidance, 7 waypoints with a small waypoint radius of $10m$ was chosen. Each diversion waypoint added drives the guidance closer to optimal, however has diminishing returns past $6 - 7$ waypoints. GVF guidance with a circular repulsive field was assigned a convergence weight $G = -1$ and circulation and decay radius coefficient k were determined by evaluating the cost function in Equation 3.26 with initial conditions $k_i = 2$ and $H = 2$. The GVF solution was bounded such that $2 \leq k \leq 4$ and $1 \leq H \leq 6$. Minimizing the cost function resulted in a decay radius coefficient $k = 2.78$ and a circulation value $H = 1.88$. The Dubin's paths for the three guidance methods discussed is shown in Figure 4.1.

VFF results in a UAV route that has excess deviation from the planned path with excessive turns. Waypoint guidance returns to the path more quickly than VFF, however deviates from the planned path farther then necessary. GVF leaves the path before waypoint

guidance and tracks the outside of the obstacle closely and then quickly converges back to the pre-planned path. The cost of each method, defined in Equation 3.26, is displayed in the bar plot shown in Figure 4.2.

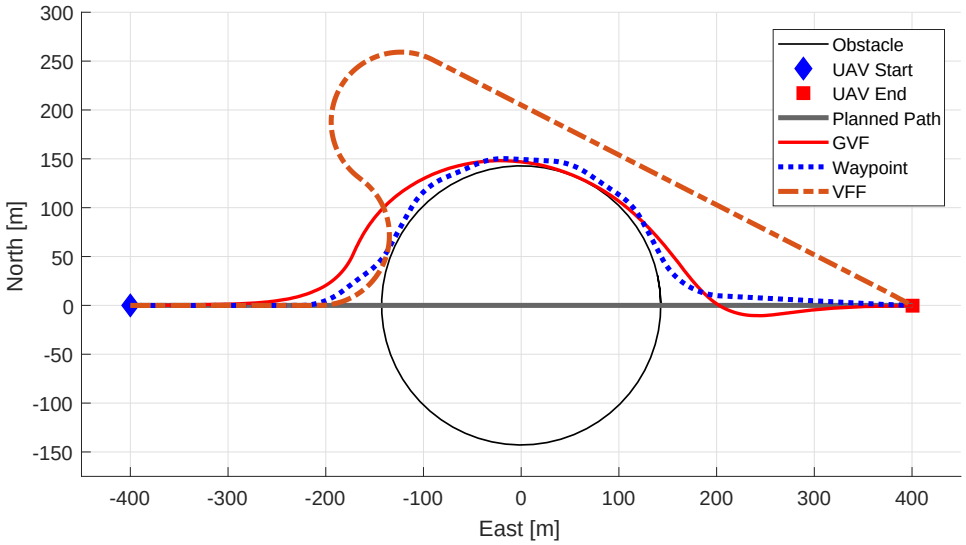


Figure 4.1: Path of UAV Guided by Guidance Methods

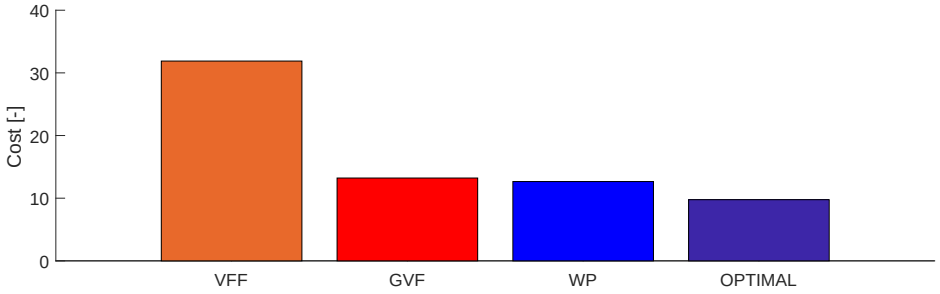


Figure 4.2: Cost Performance for Various UAV Guidance Methods

Gradient vector field for path following and circular obstacle avoidance was optimized and shown to have similar performance to waypoint avoidance without the need to re-plan the mission path. The obstacle avoidance GVF acts on the same principle as

VFF, however guidance vectors are given circulation which aid in returning the UAV back to the planned path without intervention.

4.4 Four Avoidance Scenarios

For each scenario a fixed wing UAV is assumed to be following a pre-planned path traveling at constant speed $u = 15m/s$ with a turnrate $\dot{\theta} = 20deg/s$. The UAV's initial position is set to $(100m, 0)$ and heading $\theta = 180°$, where m represents the obstacles radius multiplier. Each scenario tabulated in Table 4.1 are discussed and obstacle GVF decay radius, circulation, and cost are summarized.

Scenario 1 consists of a path centered obstacle with a radius equal to that of the UAV's turning radius θ_r. Minimizing the path deviation cost function $\bar{\gamma}$ in Phase II results in a route that avoids the obstacle and quickly returns to the planned path. The total cost of scenario 1 for avoiding the obstacle is $\bar{\gamma} = 7$ for decay radius multiplier $k = 2.8$ and circulation $H_o = 2.1$, shown in Figure 4.3. A single singularity is found from searching for solutions to the singularity condition defined in Equation 3.18 in Phase I. The singularity is removed from the UAVs route through the addition of circulation, as discussed in Phase I.

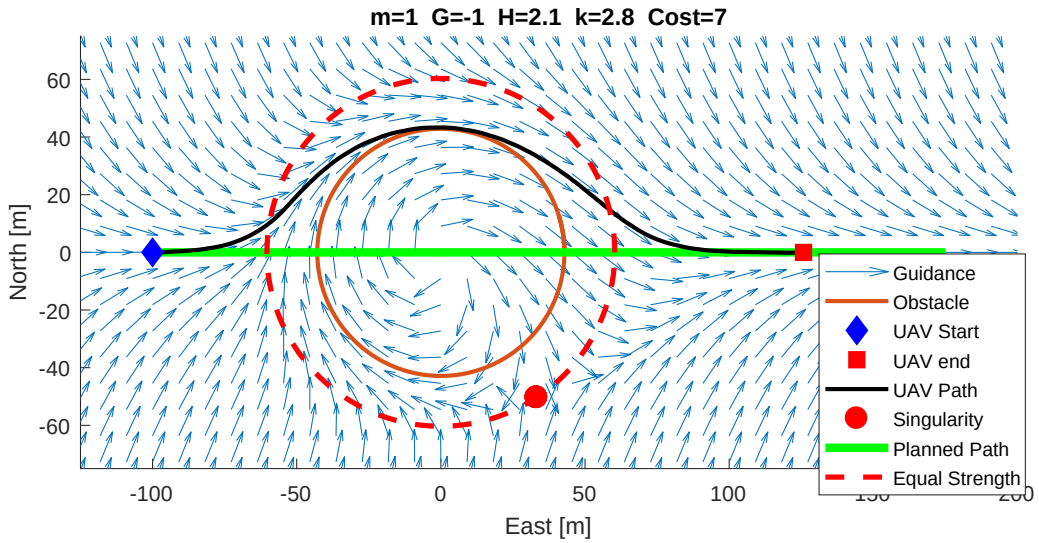


Figure 4.3: Centered Obstacle With Small Radius Avoided by Dubin's UAV in Simulation

Scenario 2 consists of a path centered obstacle with a radius equal to that of 5 times the UAV's turning radius θ_r. The total cost of scenario 2 for avoiding the obstacle is $\bar{\gamma} = 37$ for decay radius multiplier $k = 2.8$ and circulation $H = 2.1$, shown in Figure 4.3. A single singularity is detected on the radius of equal strength, however does not lie along the UAVs path.

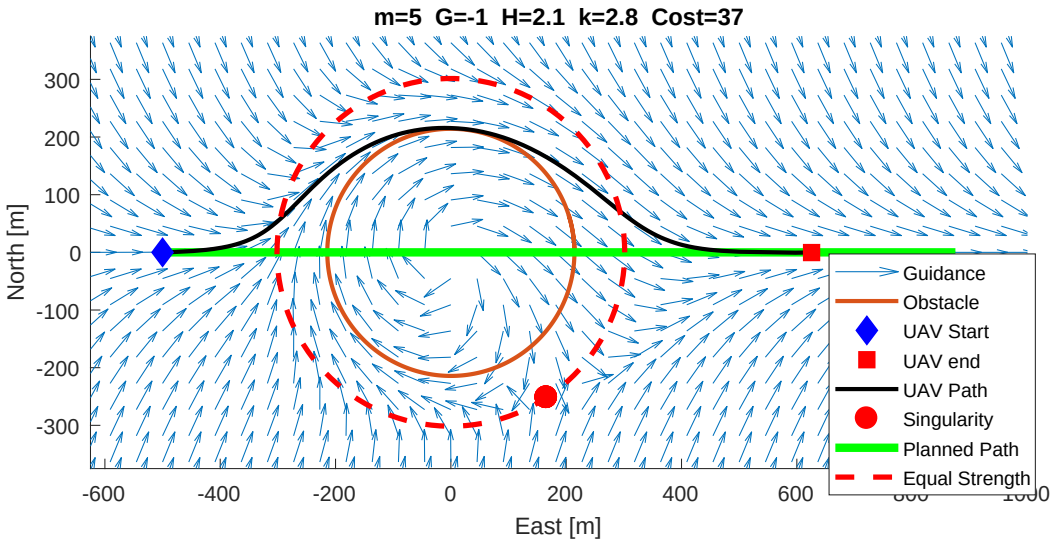


Figure 4.4: Centered Obstacle with Large Radius Avoided by Dubin's UAV in Simulation

Scenario 3 consists of a path off centered obstacle with a radius equal to that of the UAV's turning radius θ_r. The shortest direction around the obstacle can be determined from inspection, therefore negative circulation conditions and bounds are given to the minimizer to produce the guidance shown in Figure 4.5 for counterclockwise circumnavigation. The total cost of scenario 3 for avoiding the obstacle is $\bar{\gamma} = 3$ for decay radius multiplier $k = 2.4$ and circulation $H_o = -2.6$.

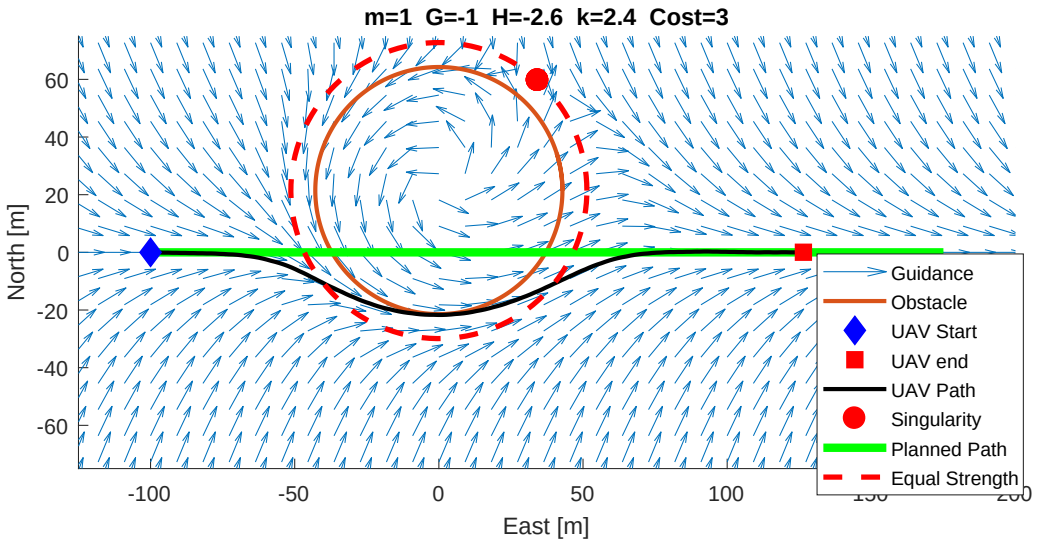


Figure 4.5: Off Centered Obstacle with Small Radius Avoided by Dubin's UAV in Simulation

Scenario 4 consists of a path off centered obstacle with a radius equal to 5 times that of the UAV's turning radius θ_r. Again, the shortest direction around the obstacle is determined from inspection. The total cost of scenario 4 for avoiding the obstacle is $\bar{\gamma} = 14$ for decay radius multiplier $k = 2.4$ and circulation $H_o = -2.7$.

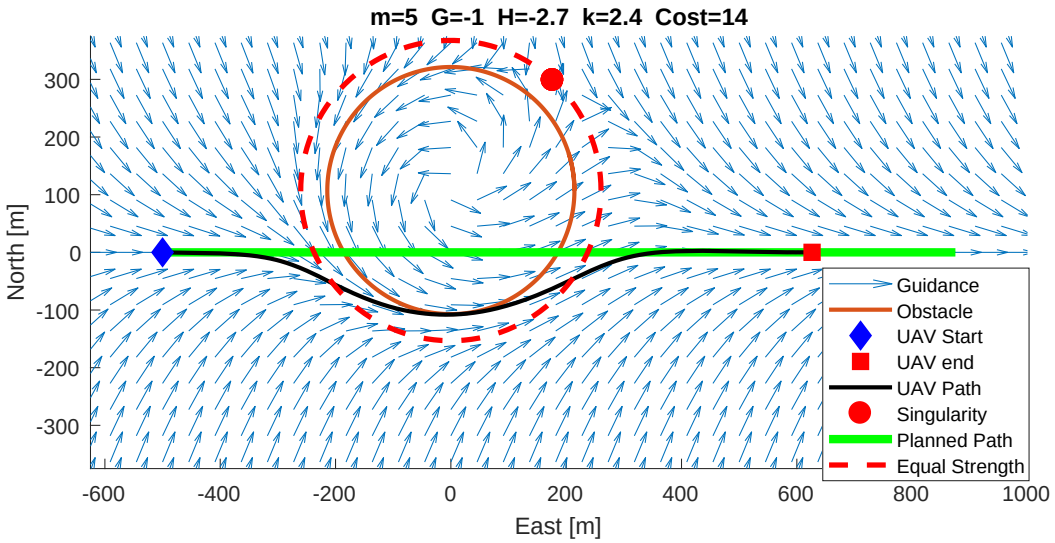


Figure 4.6: Off Centered Obstacle with Large Radius Avoided by Dubin's UAV in Simulation

Scenarios involving both path centered and off centered obstacles of various radii were investigated. For each simulation a UAV traveling along the path is guided around an obstacle and then back to the planned path without violating the obstacle, traveling through or near singularities, and without excess path deviation. The optimized GVF was then implemented on a Crazyflie 2.0 quadcopter flying under fixed wing turn rate constraints.

4.5 Phase III

The optimized GVF guidance was used to guide a turn rate constrained Crazyflie 2.0 along a path and around a virtual obstacle that lied along the planned path. Several obstacle avoidance scenarios consisting of path centered, off centered, and multiple radii obstacles were conducted using the GVF guidance. The four avoidance scenarios tested are summarized in Table 4.2. Due to space limitations, obstacles larger than $1.5\theta_r$ were not considered. Each scenario is presented followed by a vector quiver plot containing both simulation and experimental UAV flight route. For the first scenario, both speed and

PID controller response plots will be discussed. Speed and controller response plots for scenarios 2,3 and 4 can be found in the appendix.

Table 4.2: Experimental Scenarios Conducted for Crazyflie Quadcopter

Scenario Number	Obstacle radius multiplier m	Obstacle lateral positiony_o	k	H_o
1	1	0	2.2	2.9
2	1	$0.5\theta_r$	2.0	-4.6
3	1.5	0	2.4	2.7
4	1.5	$0.5\theta_r$	2.1	-3.5

Scenario 1 consisted of a quadcopter placed at initial conditions $(-2, 0)$ m along a pre-planned path and traveled at a mean speed of $0.15m/s$. An obstacle of radius $r_o = \theta_r$ was avoided by summing a repulsive obstacle GVF with the path following GVF. Cost for the simulated Dubin's UAV was $\bar{\gamma} = 7.6$ and experimental flight $\bar{\gamma} = 10.3$, a 31% difference. The experimental UAV avoids the obstacle and returns to the pre-planned path, exhibiting a similar flight path as the simulated UAV. The route of the experimental flight and simulation are found in Figure 4.7 below.

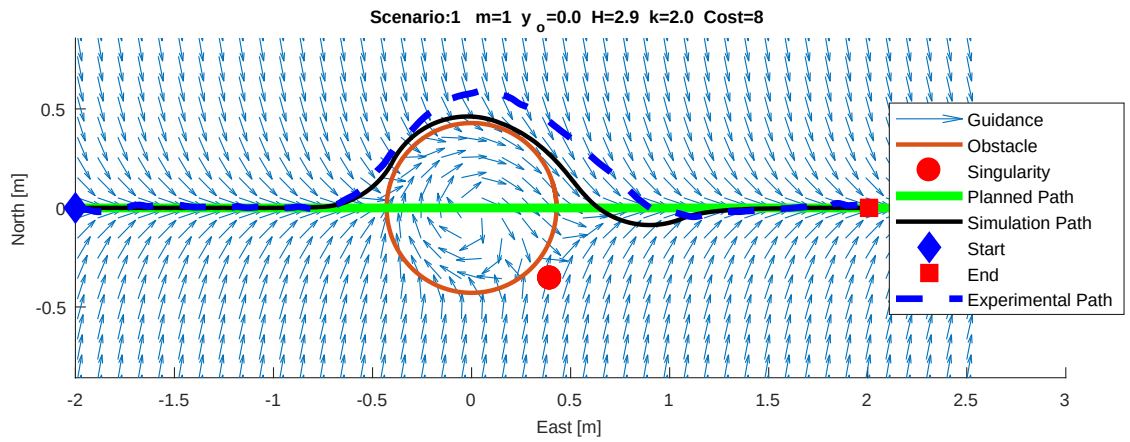


Figure 4.7: Simulated and Experimental UAV Routes for Path Centered Obstacle with Small Radius

The most likely reason for the difference between simulation and experimental cost is the lack of a velocity controller. Carrot chasing set-points does not guarantee that a fixed speed u can be maintained, even if the set-points are of equal distance away from the UAV. Additionally, in order to properly apply the Dubin's constraints of allowing finite changes in the UAVs heading, an accurate measurement of the current heading is necessary. Heading is estimated from the numerical derivative of the UAV's position at the current and last time step t, $t-1$, and time between updates dt. Due to both measurement and process noise from Vicon and the inherent instability of quadcopters respectively, measuring the state at lower speeds had significant uncertainty. The estimated speed from the numerical derivative of position is shown in Figure 4.8 below. The mean path following speed is the mean of the speed estimates ranging from the start to the end of path following and obstacle avoidance. Takeoff and landing speeds prior to path following were not included when calculating the mean speed.

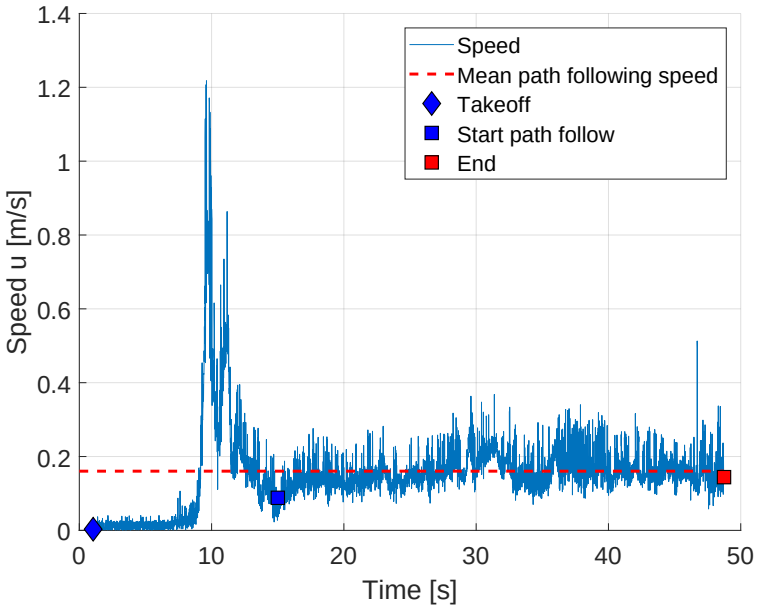


Figure 4.8: UAV Speed Versus Time During Flight Experiment of Scenario 1

The planar position states x ,y, altitude z, and yaw rate for scenario was is shown in Figure 4.9. The PID controllers were sent planar carrot set-points based on the GVF guidance, shown in red, and the state response measured with Vicon is shown in blue.

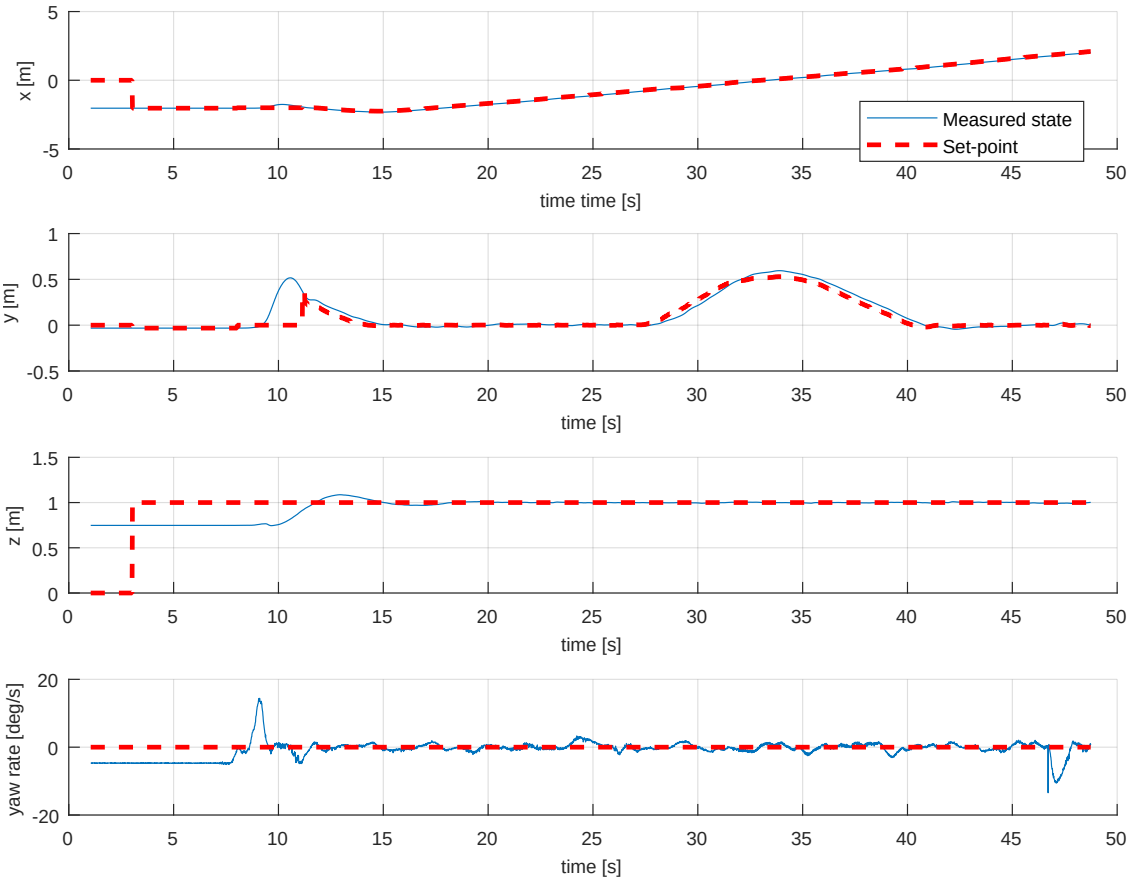


Figure 4.9: State Response Plots for Scenario 1

Scenario 2 consisted of a quadcopter placed at initial conditions $(-2, 0) meters$ along a pre-planned path and traveled at a mean speed of $0.16 m/s$. An obstacle of radius $r_o = \theta_r$ and lateral position $y_o = 0.5\theta_r$ was avoided by summing a repulsive obstacle GVF with the path following GVF. Cost for the simulated Dubin's UAV was $\bar{\gamma} = 3.6$ and experimental

flight $\bar{\gamma} = 5.0$, a 32% difference. The experimental UAV avoids the obstacle and returns to the pre-planned path, exhibiting a similar flight path as the simulated UAV. The route of the experimental flight and simulation are found in Figure 4.10 below. The combination of the path following and obstacle fields did not result in any GVF singularities in this scenario.

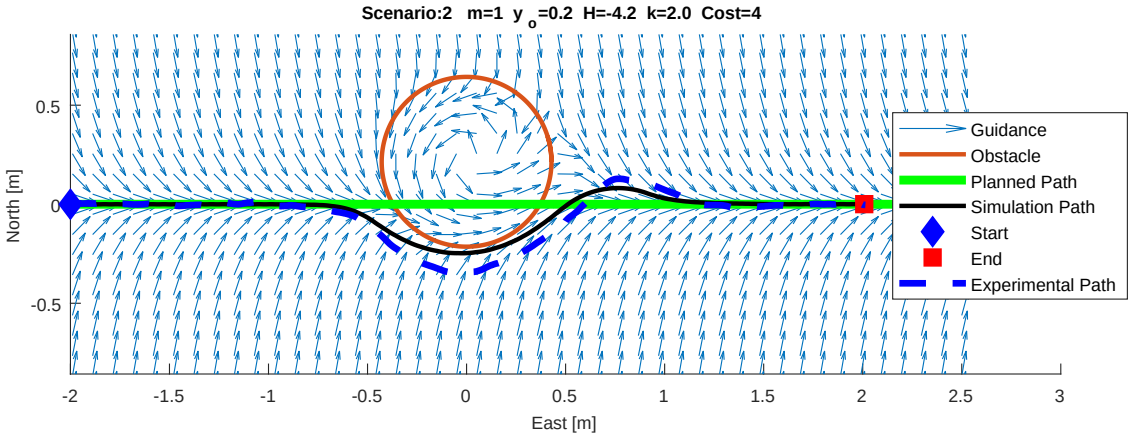


Figure 4.10: Simulated and Experimental UAV Routes for Off Centered Obstacle with Small Radius

Scenario 3 consisted of a quadcopter placed at initial conditions $(-2, 0) meters$ along a pre-planned path and traveled at a mean speed of $0.16 m/s$. An obstacle of radius $r_o = 1.5\theta_r$ and lateral position $y_o = 0$ was avoided by summing a repulsive obstacle GVF with the path following GVF. Cost for the simulated Dubin's UAV was $\bar{\gamma} = 13.4$ and experimental flight $\bar{\gamma} = 13.7$, a 2.5% difference. The experimental UAV avoids the obstacle and returns to the pre-planned path, exhibiting a similar flight path as the simulated UAV. The route of the experimental flight and simulation are found in Figure 4.11 below.

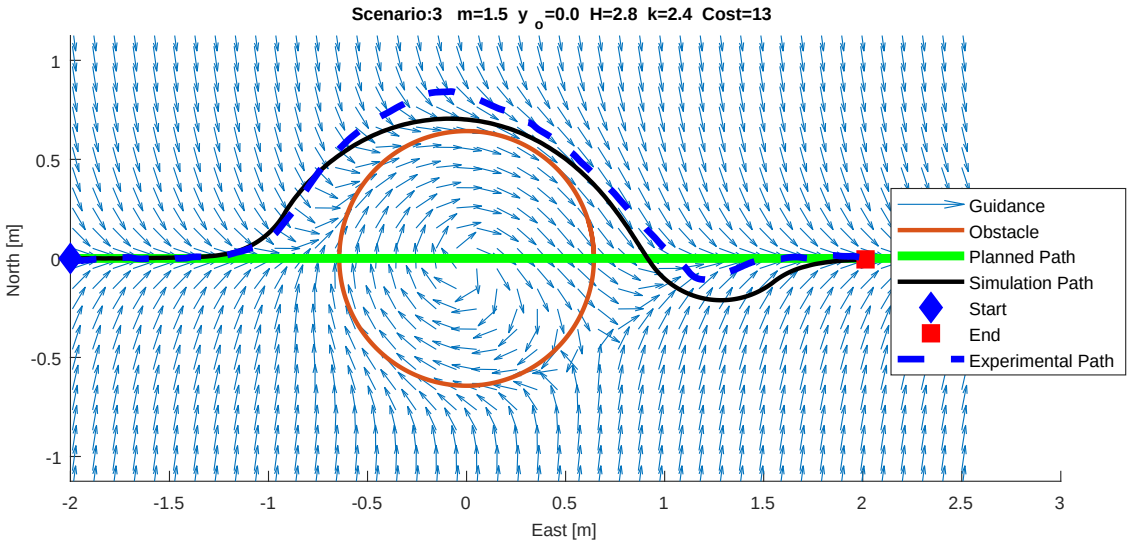


Figure 4.11: Simulated and Experimental UAV Routes for Path Centered Obstacle with Large Radius

Scenario 4 consisted of a quadcopter placed at initial conditions $(-2, 0) meters$ along a pre-planned path and traveled at a mean speed of $0.14 m/s$. An obstacle of radius $r_o = 1.5\theta_r$ and lateral position $y_o = 0.5 r_o$ was avoided by summing a repulsive obstacle GVF with the path following GVF. Cost for the simulated Dubin's UAV was $\bar{\gamma} = 5.2$ and experimental flight $\bar{\gamma} = 9.2$, a 56% difference. The experimental UAV avoids the obstacle and returns to the pre-planned path, exhibiting a similar flight path as the simulated UAV. The route of the experimental flight and simulation are found in Figure 4.12 below.

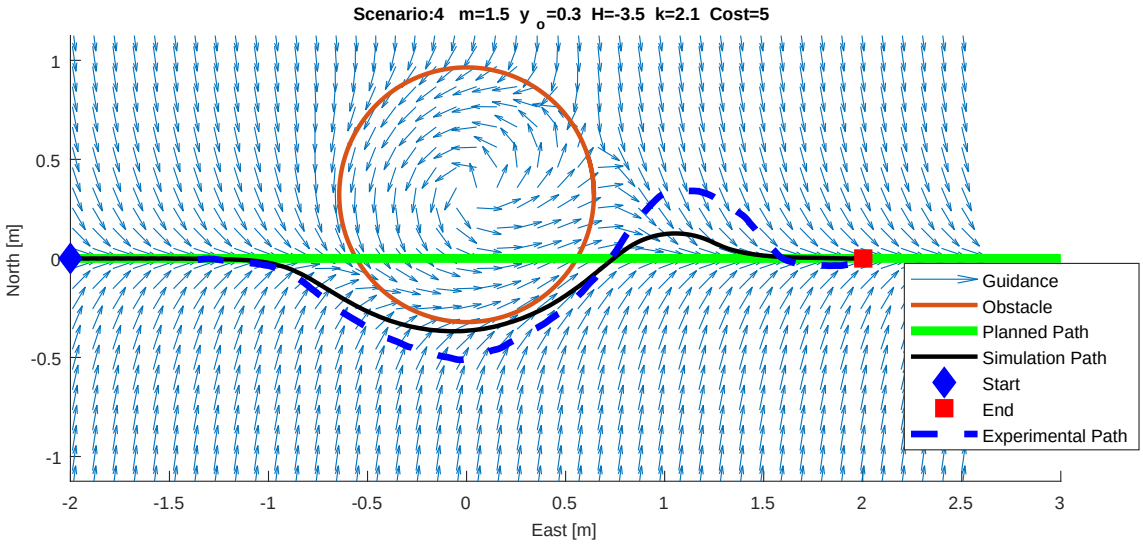


Figure 4.12: Simulated and Experimental UAV Routes for Off Centered Obstacle with Large Radius

A summary of the scenarios tested and the cost associated with each simulation and flight test are shown in Table 4.3 below.

Table 4.3: Simulation and Experimental Cost Comparison Table for Scenarios 1-4

Scenario	Simulation Cost	Experiment Cost	Cost % Difference	Mean Experiment Velocity [m/s]
1	7.6	10.3	31	0.15+-0.1
2	3.6	5.0	32	0.16 +- 0.1
3	13.4	13.7	2.5	0.16+-0.1
4	5.2	9.2	56	0.14 +- 0.1

4.6 Summary of results

An optimized GVF for path following and obstacle avoidance with singularities removed from the UAV's path was presented. Singularities in the summed GVF were

detected numerically by searching for the location of null magnitude vectors in the summed field with initial conditions placed on the radius of equal strength. Decay radius and circulation weights were determined by minimizing a cost function that penalizes the UAV for deviating from the path and violating the obstacle's space. The optimized GVF was simulated for a fixed wing UAV under various scenarios consisting of path centered, off centered, and multiple radii obstacles. Experimental flight tests were conducted with a Crazyflie 2.0 quadcopter with fixed wing constraints and the flight path cost was compared against that of simulation.

5 Conclusions

A GVF for path following and circular obstacle avoidance was optimized and shown to have similar performance to waypoint avoidance with the benefits of VFF. The obstacle avoidance GVF acts on the same principle as VFF, however guidance vectors are given circulation which aid in guiding the UAV back to the planned path without intervention. Singularities in summed GVFs were identified by numerically finding locations where GVF magnitudes were null in a summed field. It was shown that optimizing circulation and decay radius of a repulsive GVF removed singularities from a UAVs path and reduced path deviation cost function compared to VFF. Lookup tables for both circulation and decay radius were constructed for a range of obstacle configurations which can be used real time. The optimized avoidance was then used to guide a multirotor UAV in several avoidance scenarios. Contributions from each phase is summarized below.

The objective of Phase I was to characterize and present a method of locating singularities in a summed GVF. Attractive path following and repulsive GVF guidance for avoiding circular obstacles along a planned path was achieved by modifying a circular GVF's decay radius, convergence, and circulation weights. Summing the attractive and repulsive GVFs was shown to result in guidance that directs the UAV along the planned path while directing away from the obstacle. Regions in the summed guidance where the path following and obstacle guidance directly oppose each other can result in vectors of zero length, called singularities. The location of singularities were predicted to be along the radius of equal strength, where both fields have equal vector magnitudes. The location of singularities were determined by numerically solving for the location of vectors with null magnitude with initial conditions placed around the radius of equal strength.

The objective of Phase II was to determine a combination of circulation and decay radius for a circular obstacle GVF that produces an optimized obstacle avoidance. It was shown that equating the path's circulation weight to that of the UAV's

speed provides a balance between rise time and overshoot when converging and following a path. A Repulsive GVF was added to the path following GVF and an array of decay radii and circulation weights were evaluated for their path deviation costs in simulation for a worst case head on collision scenario. The minimal cost from a configuration search could be used to provide an optimal guidance, however evaluating a large array of parameters takes significant time and may be unique to each obstacle size, position, and UAV characteristics. An optimization problem was described and a method of solving for GVF circulation and decay weight with reduced computation time was presented. Optimizing GVF parameters real-time is problematic for large obstacles because computation time increases as the obstacle's radius increases. Pre-computing possible obstacle configurations and storing circulation and decay radius into a lookup table was shown to provide similar performance to optimization alone.

The objective of Phase III was to demonstrate optimized GVF guidance on multirotor UAV flying with fixed wing turn-rate constraints. The optimized GVF for path following and obstacle avoidance was implemented on a Crazyflie 2.0 quadcopter for real time guidance while flying under fixed wing constraints. Several path following and circular obstacle avoidance scenarios were tested consisting of path centered, off center, and multiple radii obstacles. Flight test and simulation flight routes were compared for both their general behavior and costs. The quadcopter followed the same general path as the simulated UAV, however had an average of 30% higher cost compared to the simulated path. The difference in cost is attributed to both the lack of a velocity controller and the high process and measurement noise when measuring the UAV's velocity.

UAVs typically rely on path planning algorithms to provide an obstacle free and flyable path prior to flight. In the event that unplanned obstacles are encountered a new path may have to be generated which may not be possible if communication with a ground station is lost. Path following and obstacle avoidance was achieved without the need to

re-plan the mission path by using optimized GVF decay radius and circulation parameters. Simulations were conducted with a fixed wing UAV modeled as a Dubin's vehicle using the GVF guidance and circulation and decay radius were selected which minimized a path deviation cost function. Singularities in the GVF were characterized and located numerically. The optimized GVF guidance was implemented in an indoor quadcopter with turn-rate constraints to emulate a fixed wing UAV. Results comparing simulations and indoor flight tests were compared.

Guidance for path following and obstacle avoidance without the need to re-plan mission paths has the potential to aid in increasing unmanned systems autonomy. Obstacles can be avoided without the intervention of a human operator and they can do so at minimal deviation from the planned path. Planned paths represent a route in which a task must be completed, therefore, maintaining a low lateral error with respect to the path increases the UAVs overall effectiveness.

References

[1] Fernandez Galarreta, J., Kerle, N., and Gerke, M., "UAV-based urban structural damage assessment using object-based image analysis and semantic reasoning," *Natural Hazards and Earth System Science*, Vol. 15, No. 6, June 2015, pp. 1087–1101.

[2] Remondino, F., Barazzetti, L., Nex, F., Scaioni, M., and Sarazzi, D., "UAV Photogrammetry for Mapping and 3D Modeling - Current Status and Future Perspectives," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXVIII-1/C22, Sept. 2012, pp. 25–31.

[3] Ariyur, K. B. and Fregene, K. O., "Autonomous tracking of a ground vehicle by a UAV," *American Control Conference, 2008*, IEEE, 2008, pp. 669–671.

[4] Teuliere, C., Eck, L., and Marchand, E., "Chasing a moving target from a flying UAV," *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, IEEE, 2011, pp. 4929–4934.

[5] Frew, E. W., "Cooperative standoff tracking of uncertain moving targets using active robot networks," *Robotics and Automation, 2007 IEEE International Conference on*, IEEE, 2007, pp. 3277–3282.

[6] Oh, H., Kim, S., Shin, H.-S., Tsourdos, A., and White, B., "Coordinated standoff tracking of groups of moving targets using multiple UAVs," *Control & Automation (MED), 2013 21st Mediterranean Conference on*, IEEE, 2013, pp. 969–977.

[7] Hyondong Oh, Seungkeun Kim, Hyo-sang Shin, and Tsourdos, A., "Coordinated standoff tracking of moving target groups using multiple UAVs," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 51, No. 2, April 2015, pp. 1501–1514.

[8] Oliveira, T., Aguiar, A. P., and Encarnacao, P., "Moving Path Following for Unmanned Aerial Vehicles With Applications to Single and Multiple Target Tracking Problems," *IEEE Transactions on Robotics*, Vol. 32, No. 5, Oct. 2016, pp. 1062–1078.

[9] Osborne, J. and Rysdyk, R., "Waypoint Guidance for Small UAVs in Wind," American Institute of Aeronautics and Astronautics, Sept. 2005.

[10] Rhee, I., Park, S., and Ryoo, C.-K., "A tight path following algorithm of an UAS based on PID control," *SICE Annual Conference 2010, Proceedings of*, IEEE, 2010, pp. 1270–1273.

[11] Park, S., Deyst, J., and How, J. P., "Performance and lyapunov stability of a nonlinear path following guidance method," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 6, 2007, pp. 1718–1728.

[12] Ratnoo, A., Sujit, P., and Kothari, M., "Adaptive Optimal Path Following for High Wind Flights," *IFAC Proceedings Volumes*, Vol. 44, No. 1, Jan. 2011, pp. 12985–12990.

[13] Griffiths, S., "Vector Field Approach for Curved Path Following for Miniature Aerial Vehicles," American Institute of Aeronautics and Astronautics, Aug. 2006.

[14] Goncalves, V. M., Pimenta, L. C. A., Maia, C. A., and Pereira, G. A. S., "Artificial vector fields for robot convergence and circulation of time-varying curves in n-dimensional spaces," IEEE, 2009, pp. 2012–2017.

[15] Gonçalves, V. M., Pimenta, L. C., Maia, C. A., Pereira, G. A., Dutra, B. C., Michael, N., Fink, J., and Kumar, V., "Circulation of curves using vector fields: actual robot experiments in 2D and 3D workspaces," *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, IEEE, 2010, pp. 1136–1141.

[16] Gonçalves, V. M., Pimenta, L. C., Maia, C. A., Dutra, B. C., and Pereira, G. A., "Vector fields for robot navigation along time-varying curves in $ n $-dimensions," *IEEE Transactions on Robotics*, Vol. 26, No. 4, 2010, pp. 647–659.

[17] Nelson, D. R., "Cooperative control of miniature air vehicles," 2005.

[18] Nelson, D. R., Barber, D. B., McLain, T. W., and Beard, R. W., "Vector field path following for small unmanned air vehicles," *American Control Conference, 2006*, IEEE, 2006, pp. 7–pp.

[19] Nelson, D., Barber, D., McLain, T., and Beard, R., "Vector Field Path Following for Miniature Air Vehicles," *IEEE Transactions on Robotics*, Vol. 23, No. 3, June 2007, pp. 519–529.

[20] Panagou, D., "Motion planning and collision avoidance using navigation vector fields," *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, 2014, pp. 2513–2518.

[21] Wilhelm, J. P., "Circumnavigation and obstacle avoidance guidance for UAVs using Gradient Vector Fields," *AIAA*, 2019.

[22] Bone, E., "UAVs backbround and issues for congress.pdf," 2003.

[23] Wise, R. A. and Rysdyk, R. T., "UAV coordination for autonomous target tracking," *Proceedings of the AIAA Guidance, Navigation, and Control Conference, Keystone, CO, Aug*, 2006, pp. 21–24.

[24] Ulun, S. and Unel, M., "Coordinated motion of UGVs and a UAV," *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*, IEEE, 2013, pp. 4079–4084.

[25] Austin, R., *Unmanned aircraft systems: UAVS design, development and deployment*, Vol. 54, John Wiley & Sons, 2011.

[26] Beard, R. W. and McLain, T. W., *Small unmanned aircraft: theory and practice*, Princeton University Press, Princeton, N.J, 2012, OCLC: ocn724663112.

[27] DroneCode, "QGroundControl - Intuitative and powerful ground control station for PX4 and ArduPilot UAVs," 2018.

[28] Wilhelm, J., Clem, G., and Eberhart, G., "Direct Entry Minimal Path UAV Loitering Path Planning," *Aerospace*, Vol. 4, No. 2, April 2017, pp. 23.

[29] Technology, C. C., "Piccolo Autopilot," 2018.

[30] Martin, L., "Kestral Flight Systems And Autopilot," 2018.

[31] DroneCode, "Pixhawk 1 Flight Controller Guide," 2018.

[32] de Marina, H. G., Kapitanyuk, Y. A., Bronz, M., Hattenberger, G., and Cao, M., "Guidance algorithm for smooth trajectory tracking of a fixed wing UAV flying in wind flows," *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, 2017, pp. 5740–5745.

[33] Manjunath, A., Mehrok, P., Sharma, R., and Ratnoo, A., "Application of virtual target based guidance laws to path following of a quadrotor UAV," IEEE, June 2016, pp. 252–260.

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

[35] Rimon, E., "Exact Robot Navigation Using Artificial Potential Functions.pdf," 1992.

[36] Liu, Y. and Zhao, Y., "A virtual-waypoint based artificial potential field method for UAV path planning," *Guidance, Navigation and Control Conference (CGNCC), 2016 IEEE Chinese*, IEEE, 2016, pp. 949–953.

[37] Borenstein, J. and Koren, Y., "Real-time obstacle avoidance for fast mobile robots in cluttered environments," *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, IEEE, 1990, pp. 572–577.

[38] Borenstein, J. and Koren, Y., "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE transactions on robotics and automation*, Vol. 7, No. 3, 1991, pp. 278–288.

[39] Koren, Y. and Borenstein, J., "Potential Field Methods and their inherent limitions for morbile robot navigation.pdf," 1991.

[40] Kim, D. H., "Escaping route method for a trap situation in local path planning," *International Journal of Control, Automation and Systems*, Vol. 7, No. 3, June 2009, pp. 495–500.

[41] Goerzen, C., Kong, Z., and Mettler, B., "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance," *Journal of Intelligent and Robotic Systems*, Vol. 57, No. 1-4, Jan. 2010, pp. 65–100.

[42] Lei Tang, Songyi Dian, Gangxu Gu, Kunli Zhou, Suihe Wang, and Xinghuan Feng, "A novel potential field method for obstacle avoidance and path planning of mobile robot," IEEE, July 2010, pp. 633–637.

[43] Li, G., Yamashita, A., Asama, H., and Tamura, Y., "An efficient improved artificial potential field based regression search method for robot path planning," IEEE, Aug. 2012, pp. 1227–1232.

[44] Jung, W., Lim, S., Lee, D., and Bang, H., "Unmanned Aircraft Vector Field Path Following with Arrival Angle Control," *Journal of Intelligent & Robotic Systems*, Vol. 84, No. 1-4, Dec. 2016, pp. 311–325.

[45] Frew, E., "Lyapunov Guidance Vector Fields for Unmanned Aircraft Applications.pdf," .

[46] Chen, H., Chang, K., and Agate, C. S., "Tracking with UAV using tangent-plus-Lyapunov vector field guidance," *Information Fusion, 2009. FUSION'09. 12th International Conference on*, IEEE, 2009, pp. 363–372.

[47] Chen, H., Chang, K., and Agate, C. S., "UAV path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 49, No. 2, 2013, pp. 840–856.

[48] Liang, Y. and Jia, Y., "Tangent vector field approach for curved path following with input saturation," *Systems & Control Letters*, Vol. 104, June 2017, pp. 49–58.

[49] Pereira, G. A. S., Choudhury, S., and Scherer, S., "A framework for optimal repairing of vector field-based motion plans," IEEE, June 2016, pp. 261–266.

[50] Pimenta, L. C., Pereira, G. A., and Mesquita, R. C., "Fully continuous vector fields for mobile robot navigation on sequences of discrete triangular regions," *Robotics and Automation, 2007 IEEE International Conference on*, IEEE, 2007, pp. 1992–1997.

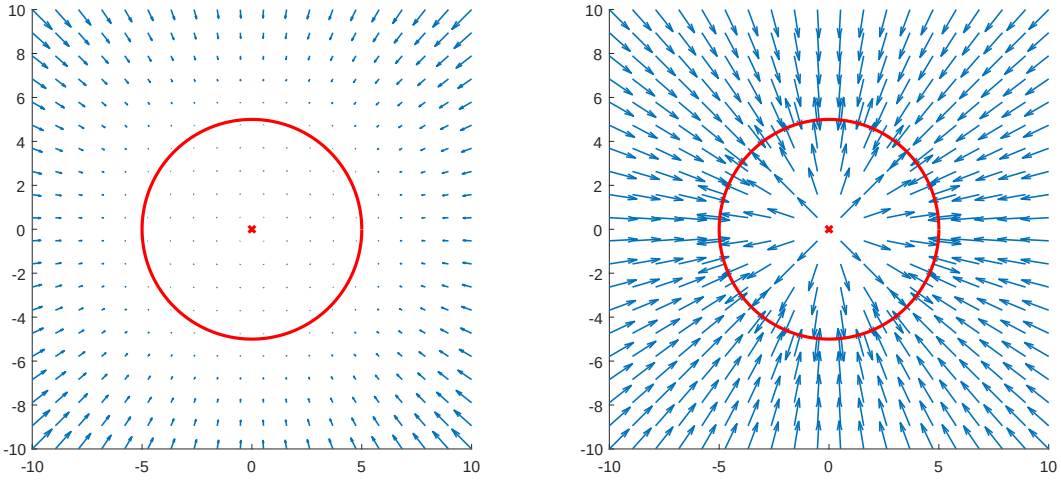
[51] Md, Z., Rg, C., and Dj, G., "Simplex Solutions for Optimal Control Flight Paths in Urban Environments," *Journal of Aeronautics & Aerospace Engineering*, Vol. 06, No. 03, 2017.

[52] Zhou, D. and Schwager, M., "Vector field following for quadrotors using differential flatness," IEEE, May 2014, pp. 6567–6572.

[53] Gerlach, A. R., *Autonomous Path-Following by Approximate Inverse Dynamics and Vector Field Prediction*, University of Cincinnati, 2014.

Appendix A: Methodology Phase I

The figures herein depict the effects of GVF normalization for both convergence $\overrightarrow{V}_{conv}$ and circulation $\overrightarrow{V}_{circ}$ components of a circular field as well the application of the decay function P. The non-normalized attractive field for converging and following a circular path is shown in Figure A.1a. Note that the vectors decay in magnitude as they approach the target curve due to the negative definite nature of the potential function V described in Equation 2.14. Normalizing the vector $\overrightarrow{V}_{c}onv$ produces vectors of equal length for the entire configuration space, shown in Figure A.1b.



(a) (b)

Figure A.1: GVF Circular Attractive Field without Normalization (a) and with Normalization (b)

Modifying the convergence weight G of the vector field shown in Figure A.1b results in a field that directs away from the target circular path, shown in Figure A.2. Note that vectors inside of the circular target path direct inwards towards a singular point, potentially leading to a trap situation. Reducing the target path's radius r to that of several

orders of magnitude smaller than the UAV's turning radius, θ_r, results in a field that directs away from a small point shown in Figure A.3.

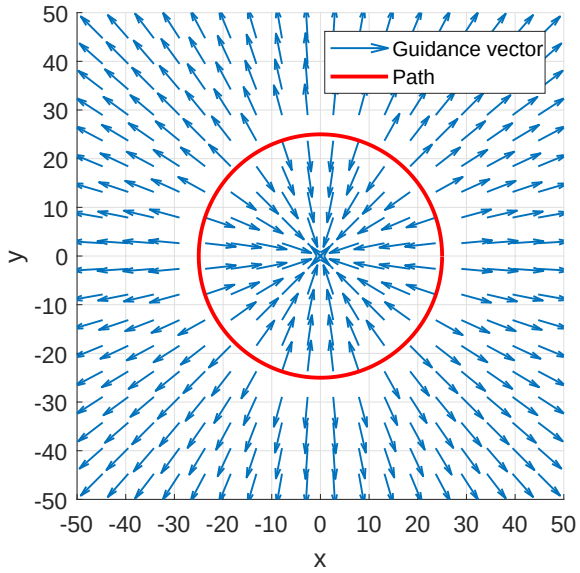


Figure A.2: Repulsive Circular Field with Large Radius

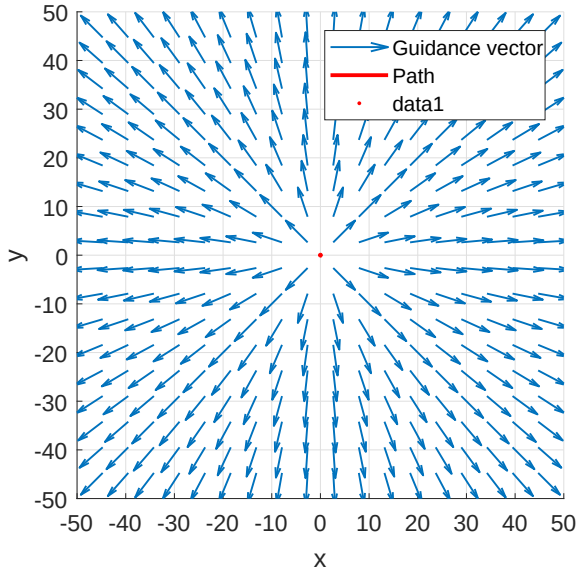
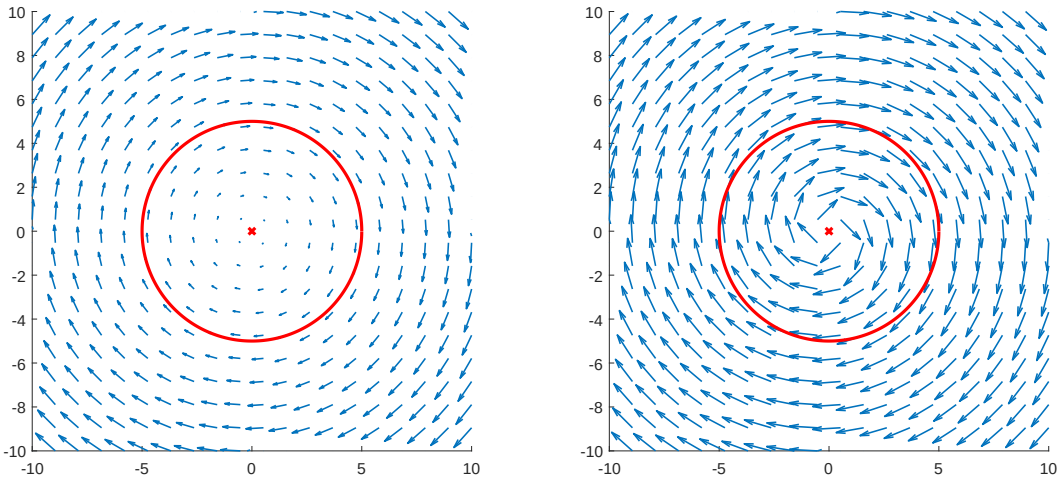


Figure A.3: Repulsive Circular Field with Small Radius

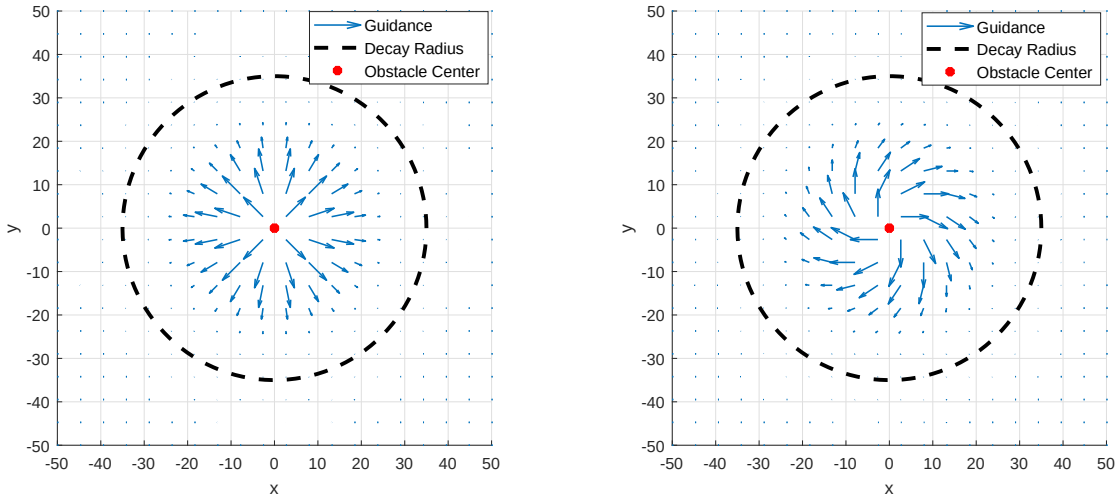
Circulation vectors of GVF decay in strength as they approach the center of the circular target curve shown in Figure A.4a. Normalizing the vectors $\overrightarrow{V}_circ$ forces the vectors to have length unity for the entire configuration space, shown in Figure A.4b.



(a) (b)

Figure A.4: Circular GVF without Normalization (a) and with Normalization (b)

Applying the decay function P described in Equation 3.15 limits the range at which the repulsive GVF has influence on the total guidance. Strictly repulsive GVF limited by decay function P is shown in Figure A.5a and equal magnitude convergence and circulation is shown in Figure A.5b.



(a) (b)

Figure A.5: Repulsive GVF a) No Circulation $H_o = 0$ and b) with Circulation $H_o = 1$

Appendix B: Methodology Phase III

Implementing the GVF guidance used in Phase I and II required that the guidance be programmed in Python. Phase III methodology show validation for straight line path and summed GVF by overlaying guidance produced in MATLAB and Python. Below are validation figures for circulation $\overrightarrow{V}_{circ}$ and the decay function P in Figures B.1 and Figure B.2 respectively.

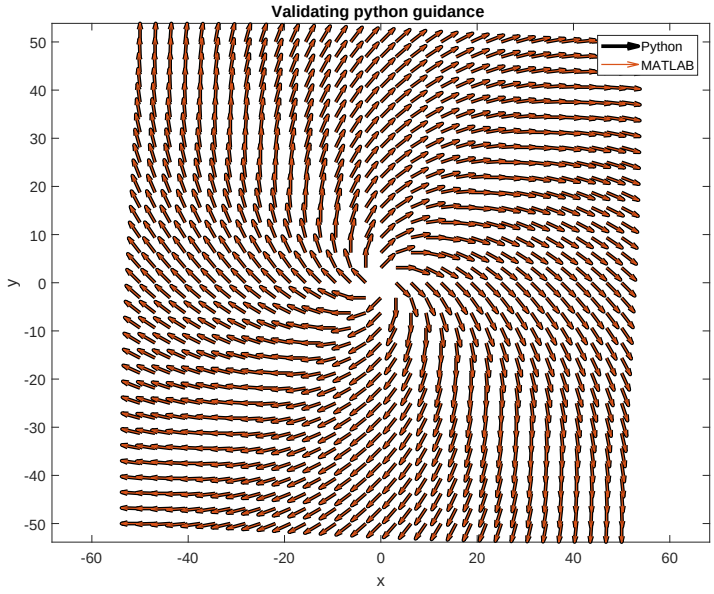


Figure B.1: Validation of Python Obstacle Guidance Overlaid with MATLAB

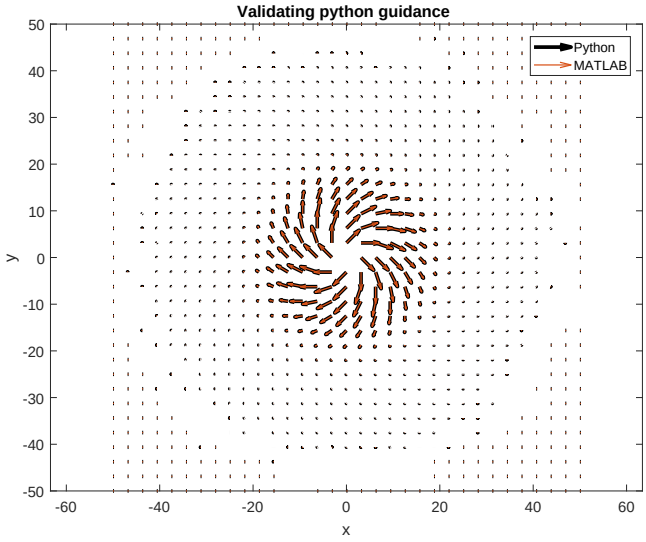


Figure B.2: Validation of Python Obstacle Decay Guidance Overlaid with MATLAB

Appendix C: Appendix - Results Phase III

Speed, u, and PID controller responses for scenarios two through four conducted in Phase III are presented herein. Speed was recorded from takeoff to the end of path following. The mean speed u in Table 4.3 was determined from the mean of speed u between the start of path following to the end in path following indicated by the blue and red markers in the speed plots below. Measured state and set-point plots for each roll, pitch, yaw rate, and thrust PID controller are shown as well.

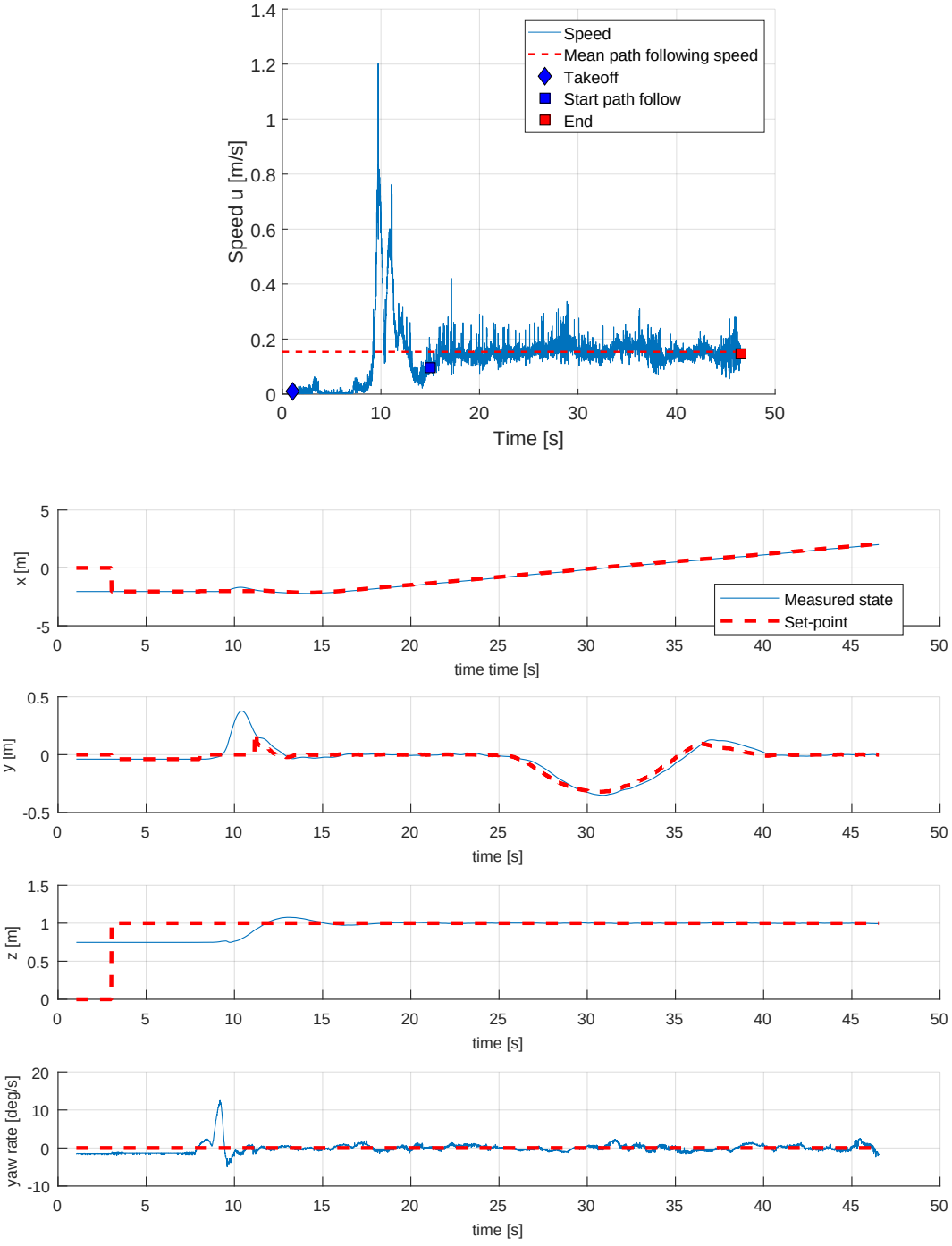


Figure C.1: Experimental Scenario 2 UAV Speed and Controller Response

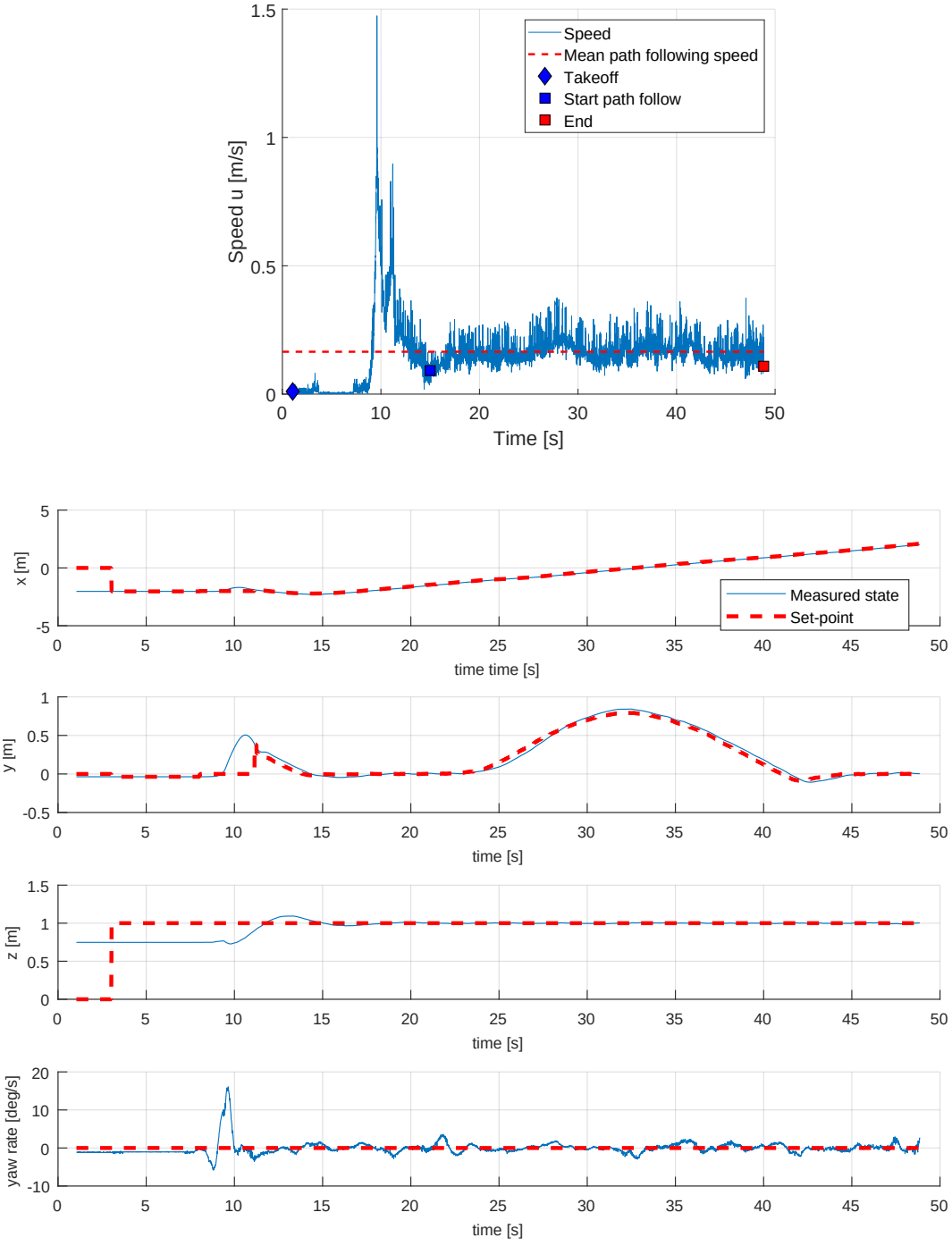


Figure C.2: Experimental Scenario 3 UAV Speed and Controller Response

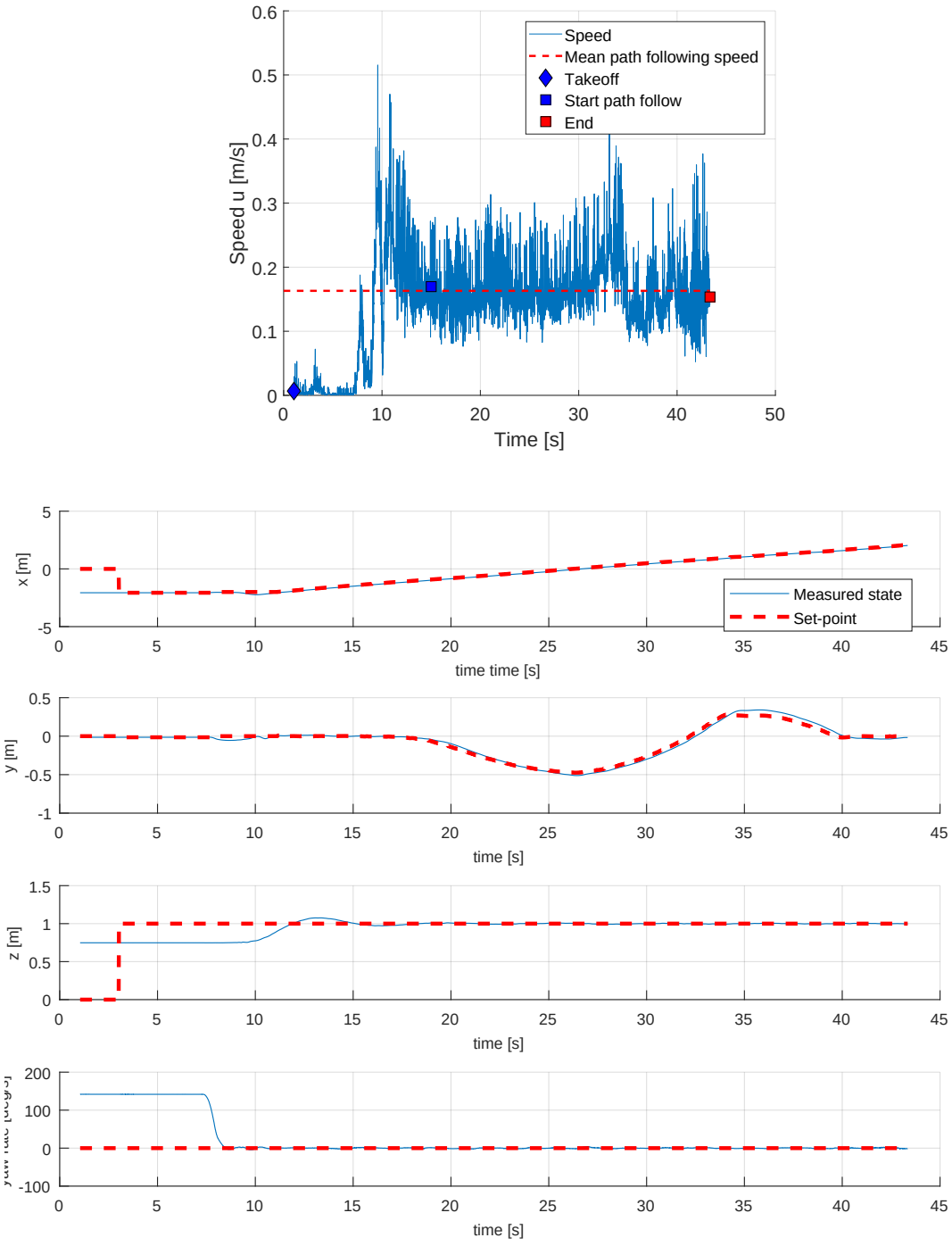


Figure C.3: Experimental Scenario 4 UAV Speed and Controller Response

APPENDIX D: OPTIMIZATION MATLAB SOURCE CODE

The script optimizedGVF.m contains the optimization algorithm to determine optimized set of obstacle field decay radius k and circulation H_o. Inputs are the UAVs initial state, constant speed u, and the obstacles configuration.

```
%=========================================================================
% optimizedGVF.m
%
% Script for optimized GVF decay radius multiplier k and circulation Ho.
% Results from thesis work. Usage:
%
%        INPUT                                          OUTPUT
% -------------------------------------------------------------------------
% UAV initial y position
% UAV velocity                                      path deviation cost
% UAV initial heading          ====> fmincon ===>   decay radius k
% Simulation dt                                     circulation  Ho
% Obstacle size (n)
% Obstacle Lateral position Yo
%
%                              Method and code developed by: Garrett Clem
%=========================================================================
clc
clear
close all

numSolved = true;       % true numerically solves for k and H

%UAV initial state
ys = 0;
velocity = 15;
heading = 0;
dt = 0.1;

%Obstacle Definition
n = 5;
obstR = n*velocity/0.35;
obstY =1/2*obstR;

xs = -100*n;
```

```
xf = 125*n;

if numSolved
    plotFinal = false;
    fr = @(X) GVF(X,velocity,dt,plotFinal,obstR,obstY,xs,ys,n,xf);

    % Optimization options
    options = optimoptions('fmincon','Display','final-detailed');
    options.DiffMinChange = 0.1;
    options.DiffMaxChange = 0.2;
    options.PlotFcn = @optimplotfval;
    options.StepTolerance = 1e-3;

    %Optimizer initial conditions and bounds
    x0 = [2,2];
    A = [];
    b = [];
    Aeq = [];
    beq = [];
    lb = [2,1];
    ub = [4,6];

    %Solve for k and H - record time to reach solution
    tic
    [Xsolved,costR] = fmincon(fr,x0,A,b,Aeq,beq,lb,ub,[],options);
    sim_time = toc;

else
    %If solver not used, manually enter k and H
    Xsolved = [2.95,0.1];
end

%Run optimized scenario and plot result
plotFinal = true;
disp('gvf parameters');
figure('pos',[10 10 900 600]);
fr = @(X) GVF(X,velocity,dt,plotFinal,obstR,obstY,xs,ys,n,xf);
fr(Xsolved);

function GVFcost = GVF(X,velocity,dt,plotFinal,obstR,obstY,xs,ys,n,xf)
obstX = 0;
```

```
%Parameters to solve for
k = X(1);

%Determine sign of circulation
if obstY>0
    H = -X(2);
else
    H = X(2);
end

%Setup vector field
vf = vectorField();

%Goal Path
vf = vf.navf('line');
vf.avf{1}.angle = pi/2;
vf.NormSummedFields = true;
vf.avf{1}.H = velocity*n;
vf.avf{1}.normComponents = false;
vf.normAttractiveFields = false;

%Obstacle
vf = vf.nrvf('circ');
vf.rvf{1}.r = 0.01;
vf.rvf{1}.H = H;
vf.rvf{1}.G = -1;
vf.rvf{1}.y = obstY;

%Setup UAV initial position (x will be changed later)
heading = 0;
uav = UAV();
uav = uav.setup(xs,ys,velocity,heading,dt);

%UAV plot settings
uav.plotHeading = false;
uav.plotCmdHeading = false;
uav.plotUAV = false;
uav.plotUAVPath = true;
uav.plotFlightEnv = false;
```

```
%Set decay field strength based on gamma ratio
vf.rvf{1}.decayR = k*obstR;
vf.rvf{1} = vf.rvf{1}.modDecay('hyper');

COST = [];
ERROR = [];

%Generate the optimal path
optPath = genOptPath(uav,obstR,0,obstY);

while uav.x<=xf

    %Distance to obstacle
    range = sqrt((uav.x-vf.rvf{1}.x)^2+(uav.y-vf.rvf{1}.y)^2);

    %Get heading
    [u,v]=vf.heading(uav.x,uav.y);
    heading_cmd = atan2(v,u);

    %Update UAV heading
    uav = uav.update_pos(heading_cmd);

    %Trap situation encountered if UAV turns around
    if uav.heading > deg2rad(175) && uav.heading <deg2rad(285)
        GVFcost = sum(COST);
        disp('trapped');
        return
    end

    %Determine cost and error
    [cost,error,location] = costANDerror(uav,obstR,obstX,obstY,optPath,dt);
    COST  = [COST;cost];
    ERROR = [ERROR;error];
end

%Determine total cost
GVFcost = sum(COST);

%Plot vector field, obstacle, UAV path, and singularities
if plotFinal == true
    vf.NormSummedFields = true;
```

```
vf = vf.xydomain(xs*(n+1),0,0,45);
plotHeat = false;
numericallyLocate = true;

cxs = obstR*cos(0:0.01:2.1*pi)+obstX;
cys = obstR*sin(0:0.01:2.1*pi)+obstY;

hold on
sing = locateSingularities(vf,plotHeat,numericallyLocate,obstR);
[X,Y,U,V] = vf.sumFields();
set(gca,'fontsize',12);
p1 = quiver(X,Y,U,V);
p2 = plot(cxs,cys,'linewidth',2);
try
    p6 = plot(sing(:,1),sing(:,2),'ro','markersize',10,'markerfacecolor','r'
        );
catch
    warning('error plotting singularities');
end
p7 = plot([uav.xs(1),175*n],[uav.ys(1),0],'g','linewidth',4);
p5 = plot(uav.xs,uav.ys,'k-','linewidth',2);
p3 = plot(uav.xs(1),uav.ys(1),'db','markersize',10,'markerfacecolor','b');
p4 = plot(uav.xs(end),uav.ys(end),'sr','markersize',10,'markerfacecolor','r'
    );

p8 = vf.rvf{1}.pltEqualStrength;
optPath = genOptPath(uav,obstR+1,vf.rvf{1}.x,vf.rvf{1}.y);

try
    h = legend([p1,p2,p3,p4,p5,p6,p7,p8],{'Guidance','Obstacle','UAV Start',
        'UAV end','UAV Path','Singularity','Planned Path','Equal Strength'},
        'Location','best');
    h.Position=[0.745555557095342 0.250416670441627 0.159999996920427 0.15];
catch
    warning('Error in plotting, may be due to no singularities');
end
axis equal
axis([xs*(n+1),xf*(n+1),-(obstR)*(n+1),(obstR)*(n+1)]);
```

```
    str = strcat('m=',num2str(n),{'  '}, 'G=',num2str(-1),{'  '},'H=',num2str(
        sprintf('%0.1f',H)),{'  '},'k=',num2str(sprintf('%0.1f',k)),{'  '},'Cost
        =',num2str(sprintf('%0.0f',GVFcost)));
    title(str);
    xlabel('East [m]');
    ylabel('North [m]');
end
end
```

Appendix E: Vector Field Construction source code

The MATLAB class vectorField.m manages the creation, summation, and plotting of GVF guidance. Instructions on how to use the class are provided in the comments above the class definition. Usage of the class is also shown in optimizationGVF.m script.

```
%Main vector field object to store, create, and modify multiple vector
%fields. Usage:
%
% vf = vectorField()    %Instance of class
% vf = vf.navf{'line'}  %Construct a (n)ew (a)ttractive (v)ector (f)ield
% vf = vf.nrvf{'circ'}  %Construct a (n)ew (r)epulsive  (v)ector (f)ield
% heading = vf.heading(x,y) %Get [u,v] heading components at (x,y)
% vf.pltff()            %Plots full quiver vector field

classdef vectorField
    %Master vector field class that handles all attractive fields and major
    %calling functions

    % =============== Functions inside of class instance =============
    % navf      - New attractive vector field
    % nrvf      - New repulsive vector field
    % xydomain  - Primarily for plotting purposes, updates xy space
    % sumFields - Sums together all attractive and repulsive vector fields
    % ======================================================================

    properties
        NormSummedFields     = true
        normAttractiveFields = true
        normRepulsiveFields  = false

        avfWeight = 1;
        rvfWeight = 1;
        avf = {};
        rvf = {};

        %Global properties
        xspace = linspace(-10,10,25);
        yspace = linspace(-10,10,25);
        n = 25;
    end
```

```
methods
    %Add a vector field
    function self = navf(self,type)
        if strcmp(type,'circ')==1
            self.avf{length(self.avf)+1} = cndr;
        end

        if strcmp(type,'line')
            self.avf{length(self.avf)+1} = gvfLine;
        end
        %Give new field default settings
        self.avf{end}.xspace = self.xspace;
        self.avf{end}.yspace = self.yspace;
    end

    function self = nrvf(self,type)
        if strcmp(type,'circ')==1
            self.rvf{length(self.rvf)+1} = cndr;
            self.rvf{end}.G = -1;
            self.rvf{end}.H = 0;
        end

        if strcmp(type,'line')
            self.rvf{length(self.rvf)+1} = gvfLine;
        end
        %Give new field default settings
        self.rvf{end}.xspace = self.xspace;
        self.rvf{end}.yspace = self.yspace;
    end


    %Change the x and y doman
    function self = xydomain(self,s,x_c,y_c,n_new)
        x_space_new = linspace(-s+x_c,s+x_c,n_new);
        y_space_new = linspace(-s+y_c,s+y_c,n_new);
        self.xspace = x_space_new;
        self.yspace = y_space_new;
        self.n = n_new;

        for i = 1:length(self.avf)
```

```
            self.avf{i}.xspace = x_space_new;
            self.avf{i}.yspace = y_space_new;
            self.avf{i}.n = n_new;
        end

        for i = 1:length(self.rvf)
            self.rvf{i}.xspace = x_space_new;
            self.rvf{i}.yspace = y_space_new;
            self.rvf{i}.n = n_new;
        end

    end

    function [X,Y,Ut,Vt] = sumFields(self)

        if length(self.avf) + length(self.rvf) < 1
            warning('No vector fields to sum');
            X = NaN;
            Y = NaN;
            Ut = NaN;
            Vt = NaN;
            return
        end
        %Get all of the attractive fields
        Usa = cell(length(self.avf));
        Vsa = cell(length(self.avf));
        for i = 1:length(self.avf)
            if self.avf{i}.active == true
                [X,Y,U,V] = self.avf{i}.ff;
                Usa{i} = U;
                Vsa{i} = V;
            end
        end
        %Sum all attractive fields into one set
        Uta = zeros(length(self.xspace));
        Vta = zeros(length(self.yspace));
        for i = 1:length(Usa)
            for j = 1:length(Usa{1})
                for k = 1:length(Usa{1})

                    Uta(j,k) = Uta(j,k)+Usa{i}(j,k);
```

```
            Vta(j,k) = Vta(j,k)+Vsa{i}(j,k);

            if self.normAttractiveFields
                N = norm([Uta(j,k),Vta(j,k)]);
                Uta(j,k) = Uta(j,k)/N;
                Vta(j,k) = Vta(j,k)/N;
            end

        end
    end
end

%Geta all of the repulsive vector vields
Usr = cell(length(self.rvf));
Vsr = cell(length(self.rvf));
for i = 1:length(self.rvf)
    if self.rvf{i}.active == true
        [X,Y,U,V] = self.rvf{i}.ff;
        Usr{i} = U;
        Vsr{i} = V;
    end
end

%Sum together all of the repulsive vector fields
Utr = zeros(length(self.xspace));
Vtr = zeros(length(self.yspace));
for i = 1:length(Usr)
    for j = 1:length(Usr{1})
        for k = 1:length(Usr{1})
            Utr(j,k) = Utr(j,k)+Usr{i}(j,k);
            Vtr(j,k) = Vtr(j,k)+Vsr{i}(j,k);

            if self.normRepulsiveFields
                N = norm([Utr(j,k),Vtr(j,k)]);
                Utr(j,k) = Utr(j,k)/N;
                Vtr(j,k) = Vtr(j,k)/N;
            end
        end
    end
end
```

```
    %Sum together attractive and repulsive vector field
    Ut = NaN(length(self.xspace));
    Vt = NaN(length(self.xspace));
    for i = 1:length(self.xspace)
        for j = 1:length(self.yspace)

            Ut(i,j) = self.avfWeight*Uta(i,j)+self.rvfWeight*Utr(i,j);
            Vt(i,j) = self.avfWeight*Vta(i,j)+self.rvfWeight*Vtr(i,j);

            if self.NormSummedFields
                N = norm([Ut(i,j),Vt(i,j)]);
                Ut(i,j) = Ut(i,j)/N;
                Vt(i,j) = Vt(i,j)/N;
            end
        end
    end
end

function [Ut,Vt] = heading(self,x,y)

    posx = x;
    posy = y;

    Usa = cell(length(self.avf));
    Vsa = cell(length(self.avf));
    for i = 1:length(self.avf)
        if self.avf{i}.active == true
            [U,V] = self.avf{i}.comp(posx,posy);
            Usa{i} = U;
            Vsa{i} = V;
        end
    end

    %Repulsive Vector Fields
    Usr = cell(length(self.rvf));
    Vsr = cell(length(self.rvf));
    for i = 1:length(self.rvf)
        if self.rvf{i}.active == true
            [U,V] = self.rvf{i}.comp(posx,posy);
            Usr{i} = U;
            Vsr{i} = V;
```

```
        end
    end

    Uta = zeros(1);
    Vta = zeros(1);
    for i = 1:length(Usa)
        for j = 1:length(Usa{1})
            for k = 1:length(Usa{1})
                Uta(j,k) = Uta(j,k)+Usa{i}(j,k);
                Vta(j,k) = Vta(j,k)+Vsa{i}(j,k);
            end
        end
    end

    Utr = zeros(1);
    Vtr = zeros(1);
    for i = 1:length(Usr)
        for j = 1:length(Usr{1})
            for k = 1:length(Usr{1})
                Utr(j,k) = Utr(j,k)+Usr{i}(j,k);
                Vtr(j,k) = Vtr(j,k)+Vsr{i}(j,k);
            end
        end
    end

    for i = 1:length(1)
        for j = 1:length(1)
            Ut(i,j) = self.avfWeight*Uta(i,j)+self.rvfWeight*Utr(i,j);
            Vt(i,j) = self.avfWeight*Vta(i,j)+self.rvfWeight*Vtr(i,j);
        end
    end
end

%=================== Plotting ====================================%
function fig = pltff(self)              %Plot full field
    [x,y,u,v] = self.sumFields;
    quiver(x,y,u,v,'linewidth',1);
    axis equal
    fig = gca;
end
```

```
        function pltPaths(self)                 %Plot attractive path
            for i = 1:length(self.avf)
                self.avf{i}.pltfnc;
            end
        end

        function pltDecay(self)                 %Plot repulsive decay
            for i = 1:length(self.rvf)
               self.rvf{i}.pltDecay;
            end
        end
    end

end
```