VELOCITY BASED POTENTIAL FIELD METHOD FOR
COLLISION AVOIDANCE OF AUTONOMOUS AERIAL VEHICLES

A Thesis in
Electrical Engineering
by
Sagar Lakhmani

© 2018 Sagar Lakhmani

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2018
The thesis of Sagar Lakhmani was reviewed and approved* by the following:

Minghui Zhu
Dorothy Quiggle Assistant Professor of Electrical Engineering
Thesis Co-Advisor

Alan Wagner
Hartz Family Assistant Professor of Aerospace Engineering
Thesis Co-Advisor

Jack Langelaan
Associate Professor of Aerospace Engineering

Asok Ray
Distinguished Professor of Mechanical Engineering and Mathematics

Kultegin Aydin
Professor of Electrical Engineering
Head of Electrical Engineering Department

*Signatures are on file in the Graduate School.
Abstract

The projected increase in the number of uninhabited air vehicles (for civilian applications such as package delivery and emergency response) and the potential rise of personal air vehicles means that the airspace will become very crowded. Safely managing these aircraft will require scalable, safe methods for collision avoidance.

The primary focus of this thesis is to develop a method which is scalable to multiple vehicles and can be implemented in real time. State-of-the-art techniques for obstacle avoidance produce good results with static obstacles. However, with the increase in the number of vehicles and dynamic obstacles, the complexity of these algorithms increases. This makes these algorithms impractical for online implementation.

The thesis proposes a velocity based approach derived from potential fields for obstacle avoidance. It uses the velocity of the aerial vehicles to generate a spherical safe zone around the vehicle. Other vehicles (which act as dynamic obstacles) get repelled by the safe zone. Given the vehicles are able to follow the commanded velocities, the vehicles will safely reach their goal location.

The method is first implemented in simulation and scalability is tested for around 50 vehicles at a time. Then, the approach is tested in an indoor environment with multiple UAVs trying to avoid each other. Results for both simulation and physical experiments are presented. Parameters such as separation distance, closest approach, and the number of collisions are used to demonstrate the feasibility of this approach.
# Table of Contents

List of Figures vii

List of Tables xi

Acknowledgments xii

Chapter 1

**Introduction**

1.1 Motivation ........................................ 2
1.2 Problem Description ................................. 3
1.3 Related Work ........................................ 4
1.3.1 Grid based methods ................................. 4
1.3.2 Cell decomposition methods ...................... 5
1.3.3 Roadmap approaches ............................... 7
1.3.4 Potential Field methods ........................... 10
1.4 Contributions ....................................... 11
1.5 Reader’s Guide ...................................... 11

Chapter 2

**Potential Field Methods** 13

2.1 Introduction ....................................... 13
2.2 Potential Functions ................................. 14
2.3 Potential Field Formulation ......................... 15
2.3.1 General idea ..................................... 15
2.3.2 Attractive potential ............................. 16
2.3.3 Repulsive Potential .............................. 18
2.4 Potential field based Path planning ............... 19
2.5 Advantages and Disadvantages of Potential Field Methods .......... 21
2.6 Summary ........................................... 23

Chapter 3

**Velocity Based Approach** 24

3.1 Introduction ....................................... 24
Appendix A

Experiment Procedure

A.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
A.2 System Requirements . . . . . . . . . . . . . . . . . . . . . . . . . 72
A.3 Hardware Configuration . . . . . . . . . . . . . . . . . . . . . . . . 73
   A.3.1 Router Configuration . . . . . . . . . . . . . . . . . . . . . . 73
   A.3.2 AR Drone Configuration . . . . . . . . . . . . . . . . . . . . . 73
A.4 Pre-experimental procedure . . . . . . . . . . . . . . . . . . . . . . 76

Bibliography 78
List of Figures

1.1 Grid based path planning algorithm in a static obstacle scenario. Black grid cells represent obstacle in the space. Pink and green circles represent robot’s initial and goal locations [1]. ........................................ 5

1.2 The figure demonstrated path planning using exact cell decomposition method. Map is divided into triangular and trapezoidal cells. A connectivity graph is constructed by representing the adjacency relation between cells. The green line showing the final path is constructed by connecting the initial and goal configuration through the midpoints of the intersections of every two successive cells [2]. ........................................ 7

1.3 The figure illustrates approximate cell decomposition method. The configuration space is initially decomposed into 4 identical rectangles. These rectangles are then recursively decomposed into 4 identical rectangles unless the interior of a rectangle has an obstacle. The black cells show the obstacle region and gray cells represent partial obstacles segments. A path extracted out of this decomposition is represented by grid contours in bold [3]. ........................................... 8

1.4 The figure shows a visibility graph which is one of the roadmap methods. Blue polygon shapes represent the obstacles. Red and green circles represent the initial and goal location of the robot. Dashed lines represent the roadmaps constructed by connecting the edges of each and every obstacle. The robot can take any of the roadmaps to reach the goal location [1]. ........................................... 9

1.5 The figure illustrates path planning using Potential field methods. The first graph shows the attractive potential generated by the goal. The second graph shows the repulsive potential generated by the obstacles. The third graph represents the combined potential of the map [4]. ....................................................... 10

2.1 The figure shows attractive potential around the goal. The blue circle shows the goal location. The quiver points around it show the velocity vector at all the points in configuration space. We can see that velocity magnitude decreases as we start moving towards the goal. ........................................... 17
2.2 The figure shows the repulsive potential around an obstacle. The red circle in the center represents the obstacle. The quiver points around it show the velocity vector at all the points in configuration space. We can see that velocity magnitude increases as we start moving towards the goal and is almost zero when obstacle is at a certain distance from the robot.

2.3 Figure depicts the combined potential field around the goal and the obstacle. The quiver plots show that at any point in space, the robot will be attracted towards the goal and repelled by the obstacles.

2.4 The obstacle is placed in a way that it lies along the shortest path of the robot towards the goal. The path shows that the robot is being repelled by obstacle shown in red and is attracted to goal represented by blue circle.

2.5 Figure illustrates local minima problem. \( q_{\text{goal}} \) shows the goal location and black curve represents the path followed by the robot. As gradient is minimum at goal location and at the point inside the box, robot assumes the goal is inside the box when following the shortest path.

3.1 Figure shows the streamlines for collision avoidance in a head on approach. Each streamline shows the collision avoidance trajectory for an array of starting locations.

3.2 Figure shows the streamlines for collision avoidance in an oblique approach. Each streamline shows the collision avoidance trajectory for an array of starting locations.

3.3 Flight paths (upper plot) and minimum separation distance for four vehicles on intersection trajectories \( (r_{\text{safe}} = 30 \text{m}) \). Only one trace is visible in the lower plot because of symmetry in the scenario definition.

4.1 A high-level schematic of the potential field collision avoidance and navigation system is presented.

4.2 This figure shows an overview of the hardware system used for the experiments. The figure shows three components: the ground station responsible for velocity calculations; the Vicon motion capture system for streaming the positions of vehicles in real-time; the aerial vehicles following the velocity commands sent by the ground station.

4.3 The AR Drone 2.0 aerial vehicle used for the physical experiments is depicted [5].

4.4 The diagram shows the controller design for experiments. Three essential components of the controller are the P controller, PI controller and a Kalman filter.
5.1 The diagram shows a snapshot of the nominal paths and vehicle locations at two different time stamps. The red open circles show vehicles that are within 100 meters of a nearest neighbor. Although there were many instances where avoidance maneuvers based on potential flow occurred, there were no collisions.

5.2 The plot shows the path taken by all the vehicles in the simulation experiment. We can see that some portion of these paths are curved. This indicated vehicles are trying to avoid other obstacle vehicles.

5.3 The figure shows a plot of number of active aircraft with respect to the simulation time. As random start and goal locations are chosen, not all the aircraft are active at one time. The maximum number of active aircraft here is 50.

5.4 The figure shows the plot of separation distance between all the vehicles with time. We can see that none of the plots reach a separation distance of 0. An indication of no collisions.

5.5 The image shows 3 AR drones flying in the Vicon motion capture studio at Penn State.

5.6 A plot showing path of two vehicles in a physical experiment. The colored lines show the path of the two vehicles. The time stamps indicate vehicle position at a particular time. Although the vehicles cross paths, the crossing happens at different time step.

5.7 Separation plots of two vehicle experiments. The blue line shows the separation distances for the path plots above. The grey lines indicate the separation results from other trials.

5.8 The figure shows snapshots of two drone experiment. The snapshots show the location of the drones at time stamps t0(initial location), t1, t2 and so on.

5.9 The figure shows quiver(velocity) plots along with the paths for a two drone experiment. The arrows represent velocity magnitudes at all time stamps. Red color plots shows the velocity and path of drone 1 and green color shows the velocity and path of drone 2. The time stamps are represented by t1, t2, and so on.

5.10 The figure shows initial velocity vectors from the figure 5.9. Left figure represents the velocity vectors of drone 1 and right figure represents velocity vectors for drone 2.

5.11 The figure shows velocity plot of a two drone experiments. Red plot shows the velocities and path followed by the drone 1. Green plot represents the same for drone 2.

5.12 The figure shows initial velocity data recorded during a two drone experiment.
5.13 A plot showing path of three vehicles in the physical experiment. The colored lines show the path of the three vehicles. The time stamps indicate vehicle position at a particular time. Although the vehicles cross paths, the crossing happens at different time steps.

5.14 Separation plots of three vehicle experiments. The one in colored lines shows the separation distances for the path plots above. The grey lines indicate the separation results from other trials.

5.15 The figure shows snapshots of three drone experiment. The snapshots show the location of the drones at time stamps t0(initial location), t1, t2 and so on.

5.16 The figure shows quiver(velocity) plots along with the paths for a two drone experiment. The arrows represent velocity magnitudes at all time stamps. Red color plots shows the velocity and path of drone 1, green color shows the velocity and path of drone 2, blue color shows the velocity and path of drone 3. The time stamps are represented by t1, t2, and so on.

5.17 The figure shows initial velocity vectors from the figure 5.16. Left figure represents the velocity vectors of drone 1, middle figure represents velocity vectors for drone 2 and right figure represents velocity vectors for drone 3. Again, the time stamps are represented by t1, t2 and so on.

5.18 The figure shows velocity plot of a three drone experiments. Red plot shows the velocities and path followed by the drone1. Green and blue plots represent the same for drone 2 and drone 3 respectively. The time stamps are represented by t1, t2, and so on.

5.19 The figure shows initial velocity data recorded during a three drone experiment.

5.20 The figure shows velocity plot of drones 2 and 3 extracted out of figure 5.18. The green and blue vectors represent the velocity of the two drones at every time step.

A.1 The figure shows a snapshot of router configuration which was used to run physical multi drone experiments.
List of Tables

4.1 Experimental details. . . . . . . . . . . . . . . . . . . . . . . . . . . 42
5.1 Closest approach for experiments. . . . . . . . . . . . . . . . . . . 64
Acknowledgments

I would like to express my deep gratitude to Dr. Alan Wagner and Dr. Jack Langelaan, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. I would also like to thank Dr. Minghui Zhu, for his advice and assistance in keeping my progress on schedule. My grateful thanks are also extended to Dr. Asok Ray for his help in providing me suggestions for future work in this area.

I thank my fellow lab mates for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last two years. I am thankful to my good friends Mollik, Sachin, Vidullan, Kalki and Abhishek for helping me out in situations where I got stuck with my research. It was a great learning experience setting up a new lab for Dr. Wagner during first semester. Thanks to Kalki for making me a part of AHS MAV competition team. Coming from electrical and software background, MAV project was a very good introduction to quadcopters for me.

Nobody has been more important to me in the pursuit of this project than the members of my family. Finally, I would like to thank my parents, whose love and guidance are with me in whatever I pursue. They are the ultimate role models and a constant source of inspiration.
Dedication

To my family, friends and REAL lab.
Chapter 1

Introduction

In the near future our skies could become crowded with Unmanned Aerial Vehicles (UAVs) [6]. UAVs may soon deliver food, medicine and packages to customers [7–9]. Moreover, companies are developing personal air vehicles such as on demand air taxis, air ambulances, flying cars [10]. The future airspace could soon be filled with both autonomous and human-piloted systems. Critically, these vehicles will be operating in the same airspace as other general aviation aircraft such as air ambulances, police helicopters, and commercial jets.

This thesis develops a multi-vehicle obstacle avoidance method for aerial vehicles. The method is designed for future airspaces which must handle a mix of human piloted and autonomously controlled vehicles. The thesis approaches this problem by using potential fields methods as a means for effective path planning and collision avoidance. The design discussed in the thesis focuses on creating a framework that is scalable, collaborative, and interactive with both novice and experienced users.
Scalability is achieved with the use of computationally efficient potential fields methods. Preliminary simulation experiments that we conducted have demonstrated that the methods discussed in the thesis can control 100 aircraft in real time using only a single laptop. The approach presented is also scalable in the sense that the same control techniques can be used to perform collision avoidance for a variety of different classes of aircraft.

1.1 Motivation

The challenge of safely managing an eclectic collection of manned and unmanned vehicles presents important infrastructure challenges that must be dealt with in the near-term. Techniques and methods must therefore be developed that can handle both autonomous UAVs and manned aircraft while still obeying airspace rules and regulations. A major aspect of integrating autonomous UAVs into airspace is path planning and collision avoidance.

In robotics, scalability represents the capability of a system, or an approach to handle large number of vehicles/robots. Given the number of vehicles that will soon crowd our skies, the development of scalable methods for collision avoidance is critical. Therefore, there is a need to develop methods which could serve as a basis for collision avoidance control and airspace regulation for dense heterogeneous airspaces.
1.2 Problem Description

The primary focus of this thesis is to develop and test a system with multiple aerial vehicles flying in an airspace while successfully avoiding collisions and moving to individual goal positions. The aerial vehicles are tasked with navigating to the goal location without colliding with the other aerial vehicles. This problem presents several inherent challenges. Some of these challenges are discussed below.

- **Potential field formulation:** The potential field methods make use of attractive and negative potential to guide the robot towards the goal. The method must be implemented on multiple vehicles that will all be moving within the experimental space. Since, the robots act as dynamic obstacles to each other, the potential field keeps changing with the time. The method proposed within this thesis generates a velocity based navigational potential field that is robust, time-variant and can be scalable to multiple vehicles flying in the airspace.

- **Real-time velocity estimation:** A vehicle’s velocity plays an important part in generating the potential field around it. Any noise in velocity measurements may cause the vehicles to collide. Hence, a robust velocity estimator is needed to determine the velocity of an obstacle vehicle.
1.3 Related Work

A number of methods for path planning have been developed [11–22]. Not all of the methods can be generalized to all situations. For instance, some methods require a map in order to generate a collision free path [13,15]. While some can only work with static obstacles and cannot handle problems with dynamic obstacles [23,24]. It is common to divide the methods into four common approaches: Grid based methods, cell decomposition methods, roadmap approaches, and potential field methods. The methods are discussed briefly in following subsections:

1.3.1 Grid based methods

Grid based methods divide the map into a fixed sized grid cells. A robot moves from grid cell to grid cell in order to reach a goal location. Typically, the robot can only move to adjacent grid cells and search algorithms like A* [13] are used to find the obstacle free path from a start location to the goal location.

The figure 1.1 shows the path generated by a grid based path planning algorithm in static obstacle scenario. The method is fast when the number of cells is small but slows significantly as the number of cells increases. On the other hand, increasing the number of cells allows the vehicle to navigate through narrow corridors without colliding with an obstacle. Although this method is fairly efficient for static obstacles, it fails when obstacles are dynamic. Also, this method fails when scaled
Figure 1.1. Grid based path planning algorithm in a static obstacle scenario. Black grid cells represent obstacle in the space. Pink and green circles represent robot’s initial and goal locations [1].

to multiple vehicles/robots.

1.3.2 Cell decomposition methods

Cell decomposition methods [25] consist of decomposing the robot’s free space into regions called cells. A connectivity graph is generated by connecting the adjacent cells. The cells that consist of obstacle free paths are connected by a link and the outcome of each cell is used to create an obstacle free path. Cell decomposition
methods consider the geometry of the obstacle when generating a path whereas grid based methods do not consider the geometry of the obstacles. The majority of cell decomposition methods can be divided into the following two types of methods:

- Exact Cell Decomposition: The map is divided into cells whose union is the free space in the map. The cell boundaries represent sudden change in robot motion while exploring the free space.

- Approximate cell decomposition methods produce cells of predefined shapes whose union is less than free space. The cell boundaries do not relate to robot motion while exploring the free space.

Figure 1.2 illustrates an exact cell decomposition method in a two-dimensional configuration space. The free space is exactly divided into trapezoidal and triangular cells. The cells are built by drawing vertical rays from the obstacles’ vertices. Two cells are said to be adjacent if they share a common portion of an edge. A free path is computed after the connectivity graph is constructed.

Figure 1.3 shows the approximate cell decomposition method. The free space is bounded externally by rectangles and internally by polygons. The rectangles are further divide into small rectangles. A free path can be found if the graph search executes successfully. For example, in the space depicted by figure 1.3 the graph and path depicted in the figure would be generated.

Because cell decomposition methods require knowledge of the map before hand, they can be computationally expensive when the obstacles are moving i.e. the
Figure 1.2. The figure demonstrated path planning using exact cell decomposition method. Map is divided into triangular and trapezoidal cells. A connectivity graph is constructed by representing the adjacency relation between cells. The green line showing the final path is constructed by connecting the initial and goal configuration through the midpoints of the intersections of every two successive cells [2].

environment is dynamic. Also, since cell based methods require a lot of computation power, the methods become inefficient when scaled to multiple vehicles.

1.3.3 Roadmap approaches

The roadmap approach creates roadmaps, which are made by connecting the robot’s free space in a network of one dimensional curves. These roadmaps consist of standardized paths and any of the paths can be chosen by connecting the robot’s goal and its initial location.
Figure 1.3. The figure illustrates approximate cell decomposition method. The configuration space is initially decomposed into 4 identical rectangles. These rectangles are then recursively decomposed into 4 identical rectangles unless the interior of a rectangle has an obstacle. The black cells show the obstacle region and gray cells represent partial obstacles segments. A path extracted out of this decomposition is represented by grid contours in bold [3].

The roadmap approach is further divided into different types of roadmap methods: Visibility graphs, Voronoi diagrams, freeway nets and silhouettes. Figure 1.4 shows the visibility graph approach for a 2D space with obstacles. A visibility graph is a non-directed graph whose nodes are the initial and goal locations of a robot along with the vertices of all the obstacles in the map. Roadmaps are
Figure 1.4. The figure shows a visibility graph which is one of the roadmap methods. Blue polygon shapes represent the obstacles. Red and green circles represent the initial and goal location of the robot. Dashed lines represent the roadmaps constructed by connecting the edges of each and every obstacle. The robot can take any of the roadmaps to reach the goal location [1].

constructed by connecting the vertices of obstacles with the goal and initial location of the robot. From the set of roadmaps one can be chosen for an obstacle free path.

Although roadmap methods are capable of generating multiple paths, these methods do not consider the geometry of the robot. Moreover, these methods fail when the obstacle is of non-convex shape. Roadmap methods are computationally inefficient. Because of this inefficiency, these methods are hard to scale for multiple vehicles.
Figure 1.5. The figure illustrates path planning using Potential field methods. The first graph shows the attractive potential generated by the goal. The second graph shows the repulsive potential generated by the obstacles. The third graph represents the combined potential of the map [4].

1.3.4 Potential Field methods

In order to address the challenges associated with managing a crowded airspace of independent vehicles, a navigation and obstacle avoidance method used must scale to a large number of vehicles. Potential field methods have been shown to scale linearly with respect to the number of vehicles [3].

Potential field methods define the robot as a point mass moving under the influence of attractive and repulsive potential. The attractive potential guides the robot towards the goal and the repulsive potential keeps the robot away from obstacles. The method is described in detail in the next chapter. Figure 1.5 illustrates how the attractive and repulsive potential combine to yield a obstacle free path for the robot.
1.4 Contributions

The key contributions of this thesis are described as below:

- **Velocity based approach**: A velocity based approach for collision avoidance is presented in the thesis. This method generates real time velocity commands for multiple vehicles in a dynamic environment. This method is presented in detail in chapter 3.

- **Scalability**: The thesis makes use of potential field methods to achieve scalability. Scalability is demonstrated in simulation as well as in physical experiments. In simulation the method has been tested in an airspace with around 50 vehicles. In physical experiments, the approach has been tested for three vehicles. The theory behind scalability of the approach is presented in section 3.5. The results showing scalability of the algorithm are discussed in chapter 5.

- **Online collision avoidance**: Demonstration of online collision avoidance on physical platforms with two and three vehicles. The implementation and results for the online collision avoidance are discussed in chapter 5.

1.5 Reader’s Guide

The remainder of the thesis is organized as follows:
• Chapter 2 gives a mathematical description of potential field methods and how an obstacle free path can be constructed for a robot.

• Chapter 3 discusses the velocity based potential field approach used for the project and formulates a spherical obstacle avoidance method for multiple robots.

• Chapter 4 discusses the implementation of this method in simulation and physical experiments. Controller design and design parameters for the physical experiments are discussed in this chapter.

• Chapter 5 presents the results of experiments conducted in simulation and on real UAVs.

• Chapter 6 provides concluding remarks, limitations, assumptions and avenues for future research.
Chapter 2

Potential Field Methods

2.1 Introduction

The first chapter presented a variety of approaches for used for collision-free navigation. This thesis focuses on the challenge of scalable collision-free navigation. Potential field methods were briefly introduced in chapter 1. This method is used to achieve scalable collision-free multi-UAV navigation.

Potential field methods were originally developed by [26] as an on-line collision avoidance method. The method does not require a map or information about the location of obstacles. This information is acquired as the robot perceives the environment. On the other hand, vehicles using potential field methods can become stuck at local minima. Local minima can be tackled by constructing a potential function in a way so that it does not have any other local minima apart from goal location [26]. This chapter discusses the basic idea of potential functions, the use
of potential functions for path planning, and the advantages and disadvantages of doing so.

2.2 Potential Functions

In physics, force and potential energy are directly related. A net force acting on any object causes it to move. The net force is generated because of the potential energy the object acquires and is given by the equation, $\vec{F} = -\nabla U$. Where $U$ is the potential energy or field. In fluid mechanics, the potential flow is defined as the gradient of the velocity potential of the fluid. The velocity potential $\phi$ is a function of space and time. The flow velocity $v$ is a vector field equal to the gradient of velocity potential, $v = \nabla \phi$. In electrostatics, electric field is expressed as the gradient of a scalar function, $\phi$, called the electrostatic potential or voltage. The electric field can be mathematically expressed as: $\vec{E} = -\nabla \phi$. Similarly in magnetism, the magnetic potential field, $B$, is defined as the gradient of magnetic vector potential, $A$, and is given by the following equation: $\vec{B} = \nabla A$.

Potential functions known as artificial potential fields, based on these ideas, are used for path planning and collision avoidance in robotics. The next few sections describe the formulation of these artificial potential functions.
2.3 Potential Field Formulation

The use of potential field functions for robot navigation are based upon the following general idea: The robot should be attracted to its goal and should be repulsed by the obstacles. This section discusses the formulation of attractive and repulsive potential fields as originally proposed by Khatib [26].

2.3.1 General idea

Let \( U \) be a potential field function defined over the configuration space, \( C \), and \( \vec{F} \) be a force which is produced by differentiating \( U \) such that:

\[
\vec{F} = -\nabla U
\]  

(2.1)

where \( \nabla U \) denotes the gradient vector of \( U \) at any point. We are concerned with a 3 dimensional space, thus the gradient at any point, \((x, y, z)\), is given by the equation,

\[
\nabla U = \left( \frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, \frac{\partial U}{\partial z} \right)
\]  

(2.2)

The potential field function can be decomposed into two elementary potential
functions: attractive and repulsive potential functions, written formally as,

$$U = U_{\text{att}} + U_{\text{rep}}$$  \hspace{1cm} (2.3)

where $U_{\text{att}}$ is the attractive potential and $U_{\text{rep}}$ is the repulsive potential. Similarly, $\vec{F}$ is the sum of two force vectors,

$$\vec{F}_{\text{att}} = -\nabla U_{\text{att}}, \quad \vec{F}_{\text{rep}} = -\nabla U_{\text{rep}}$$  \hspace{1cm} (2.4)

where $F_{\text{att}}$ is attractive potential and $F_{\text{rep}}$ is repulsive potential.

2.3.2 Attractive potential

The attractive potential field $U_{\text{att}}$ depends, at any point, on the distance of the robot from the target and can be defined as a quadratic function of euclidean distance,

$$U_{\text{att}} = \frac{1}{2} \zeta \rho_{\text{goal}}^2$$  \hspace{1cm} (2.5)

where $\zeta$ is a positive scaling factor and $\rho_{\text{goal}}$ denotes the Euclidean distance. The function $U_{\text{att}}$ is zero when $\rho_{\text{goal}} = 0$, which is the minima for the attractive potential function. The function $\rho_{\text{goal}}$ is differentiable everywhere in the map configuration space. The attractive force $\vec{F}_{\text{att}}$ is therefore given by,

$$\vec{F}_{\text{att}} = -\nabla U_{\text{att}} = -\zeta \rho_{\text{goal}} \nabla \rho_{\text{goal}}$$  \hspace{1cm} (2.6)
Figure 2.1. The figure shows attractive potential around the goal. The blue circle shows the goal location. The quiver points around it show the velocity vector at all the points in configuration space. We can see that velocity magnitude decreases as we start moving towards the goal.

For $\rho_{goal} = ||q - q_{goal}||$ where $q$ is any point in the configuration space, $\vec{F}_{att}$ is,

$$\vec{F}_{att} = -\zeta(q - q_{goal}).$$  \hspace{1cm} (2.7)

The attractive potential can be visualized in figure 2.1. The blue circle denotes the goal location and vectors denote the force acting on the robot in the configuration space. The force vectors act on the robot only if the robot is present in that specific location in the map. Following these vectors a path can be constructed to reach
the goal location. From the figure we can see that magnitude of the force decreases as the distance between the robot and the goal decreases.

### 2.3.3 Repulsive Potential

The repulsive potential, $U_{\text{rep}}$, depends on the distance of robot from obstacles. The repulsive potential must be constructed in such a way as to create a boundary for robot. Moreover, the repulsive potential should not affect the motion of the robot when it is sufficiently far away from an obstacle. The repulsive potential $U_{\text{rep}}$ is defined as,

$$U_{\text{rep}} = \begin{cases} \frac{1}{2} \eta \left( \frac{1}{\rho} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho < \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases}$$

(2.8)

where $\eta$ is a positive scaling factor and $\rho_0$ is a positive constant called the safety distance or the distance of influence. The function is zero if the distance is greater than safety distance whereas it goes to infinity as robot gets closer to the obstacle.

The repulsive force $\vec{F}_{\text{rep}}$ can be derived from $U_{\text{rep}},$

$$\vec{F}_{\text{rep}} = -\nabla U_{\text{rep}} = \begin{cases} \frac{1}{2} \eta \left( \frac{1}{\rho} - \frac{1}{\rho_0} \right) \frac{1}{\rho^2} \nabla \rho & \text{if } \rho < \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases}$$

(2.9)

The repulsive potential is depicted in figure 2.2. The red circle denotes the obstacle location and the vectors denote the force acting on the robot in the configuration space. As stated above, these force vectors act on the robot only if the robot is
Figure 2.2. The figure shows the repulsive potential around an obstacle. The red circle in the center represents the obstacle. The quiver points around it show the velocity vector at all the points in configuration space. We can see that velocity magnitude increases as we start moving towards the goal and is almost zero when obstacle is at a certain distance from the robot.

present in that specific location in the map. Following these vectors a path can be constructed to avoid the obstacles. From the figure we can see that the force magnitude starts to increase as the distance between the robot and the obstacle decreases.

2.4 Potential field based Path planning

The formulation described above offers a simple and scalable method for robot path planning. Essentially, given a location in the environment and information about the goal position and obstacle locations, the robot can calculate the net attractive
Figure 2.3. Figure depicts the combined potential field around the goal and the obstacle. The quiver plots show that at any point in space, the robot will be attracted towards the goal and repelled by the obstacles.

and repulsive forces and use gradient descent to arrive at the goal. Figure 2.3 graphically depicts the net forces for an environment that includes both an obstacle and a goal. Figure 2.4 shows the path a robot would follow to reach a goal location and avoid an obstacle from a particular starting point.
Figure 2.4. The obstacle is placed in a way that it lies along the shortest path of the robot towards the goal. The path shows that the robot is being repelled by obstacle shown in red and is attracted to goal represented by blue circle.

2.5 Advantages and Disadvantages of Potential Field Methods

Like other methods of path planning, potential field methods have their own advantages and disadvantages. This section briefly discusses these advantages and disadvantages. A more thorough treatment of this topic can be found in [27].

Because the force computation requires only the calculation of the gradient,
potential field methods are a computationally efficient means of path planning. This computational efficiency means that these techniques are suitable for real-time applications or for path planning in the presence of dynamic obstacles. Moreover, path planning can be accomplished online using the limited computational resources available to a vehicle such as a UAV. Finally, because these techniques are computationally efficient we expect them to scale well to multi-robot situations.

Yet, potential field methods also suffer from several limitations. As with other planning techniques, potential fields are limited by the accuracy of their ability to perceive obstacles and to localize the robot. Inaccuracies in perception result in incorrect navigation. The presence of local minima is another notable limitation. The inclusion of attractive and repulsive forces creates an overall navigation space which includes (possibly many) local minima. Static local minima may cause a robot to become trapped at one location. Dynamic local minima may cause the vehicle to oscillate between various minima. Moreover, certain environments, such as boxed canyons, are more prone to local minima than others. Figure 2.5 depicts one such example. Here, the robot considers the point inside the bounding box as a goal and the potential field at that point is zero. One possible remedy to this problem is to design potential functions which have no local minima other than the goal. Another approach is to complement the basic potential field approach with heuristics to escape from local minima.
Figure 2.5. Figure illustrates local minima problem. $q_{\text{goal}}$ shows the goal location and black curve represents the path followed by the robot. As gradient is minimum at goal location and at the point inside the box, robot assumes the goal is inside the box when following the shortest path.

2.6 Summary

This chapter has presented the basic ideas and intuitions related to potential functions and potential field methods. We presented the background for this idea, a derivation for attractive and repulsive fields, and discussed the advantages and disadvantages of this method. In the next chapter we will discuss velocity based approach used for this project. The velocity based approach is, essentially, derived from the concepts presented in this chapter.
Chapter 3  |  Velocity Based Approach

3.1 Introduction

Most of the successful approaches [28] for path planning and collision avoidance in dynamic environments have been a derivation of potential field methods [3,26]. This chapter focuses on velocity based approaches derived from the potential field methods discussed in chapter 2.

To use a potential field method for path planning for an aerial vehicle, the aerial vehicle is represented as a point in the configuration space moving under the influence of artificial potentials produced by the goal and obstacles. As stated in chapter 2, the use of attractive and repulsive potentials generates a force that guides the vehicle towards the goal. These navigational potential fields are, essentially, velocity fields that are guaranteed to be collision free. Thus, as long as a vehicle is dynamically capable of following its commanded velocity, it will safely arrive at its
The first half of the chapter discusses the formulation of velocity based potential field functions. The second half focuses on collision avoidance using spherical fields for single and multiple vehicle scenarios.

### 3.2 Velocity Based Potential Field functions

Using a velocity based approach, velocity commands, derived from the potential field, are used to guide the robot. The robot takes in the velocity commands by taking the gradient of the potential field in the configuration space. The potential function used in this work is motivated by fluid mechanics concepts such as uniform flow and the doublet to define our potential function.

#### 3.2.1 Uniform flow and doublet

In fluid mechanics, uniform flow is a type of flow in which the conditions of the flow do not change with respect to time at any given point. Let \( \phi \) be the potential function for uniform flow and \( V \) be vector defining the uniform flow. Since the flow is assumed to be incompressible, the potential function is defined as,

\[
\phi = v \cdot r
\]  

(3.1)
where \( \mathbf{r} \) defines a location in the flow field and is given by\( \mathbf{r} = x \mathbf{e}_x + y \mathbf{e}_y + z \mathbf{e}_z \).

Hence, equation 3.1 becomes,

\[
\phi = v_x \cdot x + v_y \cdot y + v_z \cdot z
\]  
(3.2)

In fluid mechanics, a doublet is defined as the superposition of a source and a sink of equal strengths. Let \( \vec{\mu} \) be the doublet strength, the potential flow for doublet is given by,

\[
\phi = \frac{1}{4\pi r^2} \vec{\mu} \cdot \mathbf{r}
\]  
(3.3)

### 3.2.2 Potential function

For this research, the potential function is defined relative to a spherical safe zone with radius \( r_{safe} \) surrounding the vehicle. This can be described as the summation of a uniform flow and an opposing doublet located at the origin,

\[
\phi = \mathbf{v}_\infty \cdot \mathbf{r} - \frac{1}{4\pi r^2} \vec{\mu} \cdot \mathbf{e}_r
\]  
(3.4)

where \( \mathbf{v}_\infty = v_x \mathbf{e}_x + v_y \mathbf{e}_y + v_z \mathbf{e}_z \) is the flow velocity in the far field, \( \vec{\mu} = \vec{\mu}_x \mathbf{e}_x + \vec{\mu}_y \mathbf{e}_y + \vec{\mu}_z \mathbf{e}_z \) defines the doublet strength of the sphere and \( \mathbf{e}_r = \frac{\mathbf{r}}{r} \) is the unit vector pointing to a location in the flow field. The velocity field can be calculated by taking the gradient of potential field function and is given by,
\[ \mathbf{v} = \nabla \phi = v_x e_x + v_y e_y + v_z e_z + \frac{\mu_x e_x \cdot e_r}{2\pi r^2} e_r + \frac{\mu_x e_x \cdot \nabla e_r}{4\pi r^2} + \frac{\mu_y e_y \cdot e_r}{2\pi r^2} e_r + \frac{\mu_y e_y \cdot \nabla e_r}{4\pi r^2} + \frac{\mu_z e_z \cdot e_r}{2\pi r^2} e_r + \frac{\mu_z e_z \cdot \nabla e_r}{4\pi r^2}. \] (3.5)

### 3.2.2.1 Uniform flow - attractive potential

Uniform flow of the far field is similar to the attractive potential field discussed in chapter 2 and is responsible for guiding the robot towards the goal. Equation 3.4 defines the combined potential field where the first part of the equation is the uniform flow. The uniform flow represents the attractive potential discussed in chapter 2 and is given by the equation,

\[ \phi_{att} = \mathbf{v}_\infty \cdot r \] (3.6)

and the uniform flow velocity or goal velocity is given by the first three components of equation 3.5. This velocity attracts the robot toward the goal and hence we call it the goal velocity. The goal velocity depends on the distance of robot from the goal which is described by the equation 3.5.

### 3.2.2.2 Opposing doublet - repulsive potential

The opposing doublet creates a repulsive force around the obstacle. The doublet generates spherical streamlines around the obstacle as shown in figures 3.1 and 3.2. The opposing doublet in equation 3.4 denotes the repulsive potential and is given
by following equation,

\[ \phi_{\text{rep}} = -\frac{1}{4\pi r^2} \vec{\mu} \cdot e_r \]  

(3.7)

and the velocity induced by the doublet is given by the last six components of equation 3.5. The doublet strength, \( \vec{\mu} \), depends on the radius of the safe zone and the magnitude of the components of the velocity, so that, \( \vec{\mu}(\cdot) = 2\pi v(\cdot) r_{\text{safe}}^3 \). The velocity field generated by this component is induced by the other drones (acting as obstacles) and hence it is called the induced velocity.

### 3.3 Spherical Avoidance

For this work, a spherical avoidance safe zone is used around a vehicle to prevent collisions. The basic idea is to define a repulsive potential around an obstacle vehicle in such a way that the streamlines around the obstacle form a spherical shape when the distance to the obstacle approaches the \( r_{\text{safe}} \) value. The incoming robot can then follow these streamlines to avoid the obstacle robot.

Figure 3.1 shows the collision-free path for different starting positions on a simulated 150 meter plane for a head-on approach toward a moving obstacle. Figure 3.2 shows the collision-free path for different starting positions on a simulated 150 meter plane for an oblique approach to a moving obstacle. Each streamline shows the path of a converging vehicle in the body-fixed frame of the vehicle inside the virtual avoidance zone. Since, we are following a spherical avoidance approach, the
Figure 3.1. Figure shows the streamlines for collision avoidance in a head on approach. Each streamline shows the collision avoidance trajectory for an array of starting locations. Streamlines are similar for both head-on and oblique approaches when viewed with respect to the incoming vehicle.

3.4 Path Following

In addition to the problem of generating safe paths, any collision avoidance system must be able to follow to those paths. This has two implications: first, the trajectories must be kinematically and dynamically feasible; second, the flight
The controller must be robust enough to follow complex trajectories. Consider a kinematic vehicle,

\begin{align*}
\dot{x} &= v \cos \psi \cos \gamma \\
\dot{y} &= v \sin \psi \cos \gamma \\
\dot{z} &= -v \sin \gamma
\end{align*}

where $\psi$ is the heading and $\gamma$ is the flight path angle with respect to the local horizontal. The three inputs are acceleration, $v$, and the rates of change of the
heading and the flight path angle ($\psi$ and $\gamma$, respectively). Equations 3.8, 3.9, 3.10 can be used as inputs to an aircraft with an autopilot module that can follow speed rate, turn rate, and flight path angle rate commands. Turn rate commands can be computed based on the vehicle’s current velocity and the velocity field induced by the obstacles.

### 3.5 Multiple Vehicle Scenario

When managing $n$ vehicles, there are $n-1$ obstacles for any vehicle at a specific time. The navigation potential induced by the $n^{th}$ obstacle is given by,

$$
\phi_n = (v - v_n) \cdot r_n - \frac{1}{4\pi r^2} \vec{\mu}_n \cdot e_{r,n}
$$

(3.11)

Thus, the total potential induced by $N$ obstacles is

$$
\phi = \sum_{n=1}^{N} \phi_n
$$

(3.12)

The total velocity field can now be computed from the net potential field. Figure 3.3 shows trajectories for a set of four vehicles on a crossing-course (each vehicle starts at a green open circle and ends at a red open circle) as well as the minimum separation between vehicles. A low-order approach to trajectory following control is used here: it computes the desired angular rates based on the difference between
Figure 3.3. Flight paths (upper plot) and minimum separation distance for four vehicles on intersection trajectories ($r_{safe} = 30$m). Only one trace is visible in the lower plot because of symmetry in the scenario definition.

the vehicle’s current velocity vector and the velocity vector induced by the potential field. The four vehicles circle each other, ensuring collision avoidance. Note that this is an example of joint collision avoidance, each vehicle avoids the other three. The minimum separation distance is also shown. Because of symmetry in this
particular scenario, each vehicle’s minimum separation distance is equal to the others at each time step.

3.6 Summary

This chapter presents the theory underlying the velocity based approach to potential field methods. This approach uses concepts from classical fluid mechanics and potential field theory such as uniform fields and doublets. We also presented a derivation for attractive and repulsive velocities. We showed that these velocities could be combined in manner that guaranteed a collision-free pathway. Furthermore, we developed a spherical avoidance term that can be used to avoid collisions in single and multiple vehicle scenarios.

In the next chapter we will discuss our implementation of the ideas we presented in this chapter. The implementation and experimental results for both simulation and physical experiments will follow.
Chapter 4  
Implementation and Controller Design

4.1 Introduction

This chapter introduces the hardware and software implementation of the system described in Chapter 3. First part of the chapter focuses on the hardware implementation. Second part discusses the design of the vehicle controller for software implementation.

4.2 Hardware Implementation

As described in Chapter 3, the potential field generated by each robot is used to create an induced velocity that guides the vehicle. Figure 4.1 depicts a high level
Figure 4.1. A high-level schematic of the potential field collision avoidance and navigation system is presented.

schematic for the collision avoidance system. The important physical components of the system are the ground station, the communication link and the aerial vehicles receiving the velocity commands. Figure 4.2 shows an overall system overview.

4.2.1 The ground station

The ground station is responsible for sending the commands to the aerial vehicles. As shown in figure 4.2 the ground station receives the position data from the communication link server. The position inputs are used to calculate the goal and induced velocity on the ground station as quickly as possible. The goal velocity component comes from the final goal location of the robot. As shown in figure 4.1, the induced velocity component is generated by the potential field induced by the other vehicles flying in the airspace. The ground station is connected to each aerial vehicle via WIFI through a router. Commands to the aerial vehicles are sent using UDP at a 30 Hz frequency. The ground station is capable of reading the sensor data sent by the aerial vehicle through the same connection. The data includes
Figure 4.2. This figure shows an overview of the hardware system used for the experiments. The figure shows three components: the ground station responsible for velocity calculations; the Vicon motion capture system for streaming the positions of vehicles in real-time; the aerial vehicles following the velocity commands sent by the ground station.
images from the camera, the IMU and the accelerometer values. For this project, we only utilize the IMU measured velocity readings.

4.2.2 Vicon motion capture

For the physical experiments, a Vicon motion capture system was used to determine each of the vehicle’s positions. The Vicon motion capture system consists of a set of infrared cameras that can track the position of an object. For the physical experiments, the position coordinates of each aerial vehicle was sent to ground station at a frequency of 600 Hz. The Vicon datastream SDK was used to read the position values sent by the motion capture system. The SDK includes a set of functions that are used to interface with the motion capture system.

4.2.3 AR drone 2.0 vehicle

We used AR Drones 2.0 sold by Parrot (Figure 4.3) as the research vehicle. The AR Drone is open platform, affordable and has a wide range of onboard sensors. This vehicle has become a popular tool for research and education [29] and has been used in experiments in the field of vision-based autonomous navigation [30–32], autonomous surveillance [33], and human-machine interaction [34]. The AR Drone has an ARM Cortex A8 1 GHz 32-bit on-board processor, Linux operating system, WIFI capable of 30 HZ data transfer, a 720 pixel front camera with a 93° lens, recording up to 30 frames per second, and a downward looking camera with a 64°
Figure 4.3. The AR Drone 2.0 aerial vehicle used for the physical experiments is depicted [5].

lens that captures up to 60 frames per second. The AR Drone is fitted with a protective foam shroud to prevent damage. It also has a built-in flight controller that couples a PID controller to optical flow and pressure sensors. The typical battery life for AR Drone is 12 minutes when flying at a speed of 5 km/h. For indoor experiments, this speed is rarely achieved and the battery life is enough to run several trials of an experiment.

4.3 Controller Design

For the system shown in figures 4.1 and 4.2, a high level controller was designed to perform the experiments. The inputs to the controller were the position data received from Vicon motion capture system and IMU velocities received from the vehicles. The controller is able to drive the vehicles toward the goal, but, it can also
guide the vehicles away from dynamic obstacles. Figure 4.4 shows the controller implemented for the vehicles in the physical experiment. The main components of the controller are discussed below.

### 4.3.1 Proportional controller for position

In feedback control systems, the proportional control is used to achieve a desired state by reducing the error of the current state. This error is in proportion to the control commands generated out of the proportional controller. As shown in figure 4.4, the first part of our controller is a P controller, which takes the position errors and calculates the desired goal velocity for the vehicles. Continuous position values for all vehicles are streamed by the Vicon motion capture system to the vehicles. Error is calculated at every instant by subtracting the desired position from the current position. From this error the velocity commands are generated.
The proportional controller can be expressed as the following equation,

\[ v_{\text{des}} = K_p x_{\text{err}} \]  \hspace{1cm} (4.1)

where \( K_p \) is proportional gain of the controller.

### 4.3.2 Proportional-integral controller for velocity

Similar to the previous subsection, a proportional controller for velocity was implemented along with an integral component to generate acceleration commands for the vehicle. The velocity controller is needed for generating acceleration commands which act as input for the vehicles. This controller takes in the velocity errors at each time step and generates the acceleration commands for the vehicles. The integral term in the PI controller accumulates the error generated over time until the desired state is achieved. The integral term is proportional to both error and time duration of the error. The PI controller is described as,

\[ a = K_{p,v} v_{\text{err}} + K_{i,v} \int v_{\text{err}} dt \]  \hspace{1cm} (4.2)

where \( a \) is the commanded acceleration, \( K_{p,v} \) is proportional gain and \( K_{i,v} \) is integral gain. The desired velocity is calculated by the proportional controller discussed in the previous subsection. A Kalman filter estimates the current velocity from the position coordinates received by the Vicon motion capture and the IMU measured
velocity from the vehicle’s sensors. The error between current and desired velocity is fed to the PI controller which generates the acceleration values for the vehicles.

### 4.3.3 Kalman filter for velocity estimation

The Kalman filter [35], proposed in 1960, has long been used for state estimation and prediction tasks. The Kalman filter is an algorithm used to predict the value of an unknown variable in a dynamic system. Kalman filters use noisy sensor measurements taken over time to estimate the value of this unknown variable.

The Kalman filter used for this research takes in the position measurements over time. These measurements may contain noise and other inaccuracies. The velocities estimated by the filter tend to be more accurate than predicting velocities from a single measurement. A Kalman filter algorithm usually consists of two steps, a prediction step and a correction step. During the prediction step, an estimate of the current velocity/state is made along with an estimate of the uncertainties. In correction/update step, the velocity is updated after the next measurement has been observed. Let A be our state transition matrix for the system in figure 4.2, let B be the input vector containing the position and IMU measured velocities. As per [36], the prediction step is given by,

\[
\hat{x} = Ax + Bu \\
\hat{P} = APA' + Q
\] (4.3) (4.4)
where $x$ is the state vector and $\hat{x}$ is the estimated or predicted state vector. $P$ is the covariance of the state vector estimate and $\hat{P}$ is the estimated covariance. $Q$ represents the process noise covariance. The correction step is calculated as,

$$x = \hat{x} + K(Z - H\hat{x})$$ \hspace{1cm} (4.5)

$$P = \hat{P} - KH\hat{P}$$ \hspace{1cm} (4.6)

where $H$ is the observation matrix, $Z$ is the input values containing the current position and IMU measured velocities and $K$ is Kalman gain given by,

$$K = \hat{P}H'(H\hat{P}H' + R)^{-1}.$$ \hspace{1cm} (4.7)

The output state vector containing the estimated velocity is given by $X = H\hat{x}$.

The derivation of these equations is detailed in the book [36].

The Kalman filter design parameters and matrices used for the controller in the experiments is provided in table 4.1. The state transition matrix, the measurement

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-Controller, $K_p$</td>
<td>0.03</td>
</tr>
<tr>
<td>PI-Controller, $K_i$</td>
<td>0.01</td>
</tr>
<tr>
<td>PI-Controller, $K_p$</td>
<td>$2\sqrt{K_i} = 0.2$</td>
</tr>
<tr>
<td>Kalman filter, position process noise</td>
<td>0.002 m</td>
</tr>
<tr>
<td>Kalman filter, velocity process noise</td>
<td>0.002 m/s</td>
</tr>
<tr>
<td>Kalman filter, position measurement noise</td>
<td>0.1 m</td>
</tr>
<tr>
<td>Kalman filter, velocity measurement noise</td>
<td>0.01 m/s</td>
</tr>
</tbody>
</table>
matrix, and the process and measurement noise were formulated as,

\[
A = \begin{bmatrix}
1 & 0 & 0 & dt & 0 & 0 \\
0 & 1 & 0 & 0 & dt & 0 \\
0 & 0 & 1 & 0 & 0 & dt \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
H = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
Q = \begin{bmatrix}
0.002 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.002 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.002 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.002 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.002 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.002
\end{bmatrix}
\]
\[
R = \begin{bmatrix}
0.01 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.01 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.01 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.0001 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.0001 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.0001 \\
\end{bmatrix}
\]

where \( dt \) is the time step, \( A \) is the state transition matrix, \( H \) is the measurement matrix. \( Q \) and \( R \) are the process and measurement matrices respectively. For the Vicon motion capture system, the \( Q \) and \( R \) matrices were determined experimentally. The variable \( Z \) in equation 4.5 is the input vector which contains measured position and velocity values at every time step,

\[
Z = [x \ y \ z \ vx \ vy \ vz]^T
\]

### 4.4 Summary

This chapter presented the hardware and implementation details for the method developed in Chapter 3. We discussed the hardware components such as the ground station, the communication link and the vehicle. An overview of how these components work together was described. The controller design and the underlying math was also developed. A P-controller was created for controlling the vehicle’s
position and a PI controller was developed for controlling the vehicle’s velocities. A Kalman filter was synthesized to estimate the vehicle’s velocities. In the next chapter, the theory developed in chapter 3 and the hardware implementation developed in this chapter are used to perform a series of experiments focused on the research question.
Chapter 5

Simulation and Experimental Results

5.1 Introduction

This chapter presents a variety of different experiments to test and validate the velocity based potential field methods developed in chapter 3. Our initial experiments were in simulation. Simulation experiments are easier to quickly prototype than real robot experiments. Moreover, these simulation experiments allowed us to test conditions, such as having 100 aircraft per kilometer, which could not be recreated in the real world.

Real world experiments, on the other hand, are critical for testing our collision avoidance methods on real hardware. These experiments were generally used to verify the simulation results. For all of the experiments conducted as part of this
thesis, closest approach and number of collisions served as the primary measures of success.

This chapter presents the results of simulation and physical experiments performed on multiple UAVs. Parameters such as closest approach and separation distance are discussed in detail. The results presented are compared with contemporary methods of air traffic control and management.

5.2 Simulation Experiments

In order to validate the method presented in Chapter 3, a Matlab simulation was developed. The spherical collision avoidance scheme discussed in section 3.3, with a kinematic aircraft was implemented in simulation. This simulated airspace, which we call BusyWorld, was implemented to test the basic feasibility of the algorithms in a crowded airspace. In BusyWorld, vehicles fly between randomly generated start and goal locations (with the proviso that the flight path must traverse a significant distance across the airspace). The aircraft fly at constant speed (chosen from a uniform distribution over a predetermined interval) and have safe zones that can be randomly selected.

5.2.1 Experimental parameters

As described in chapter 3, in theory there should be no collisions between the aerial vehicles in the simulation experiments. The independent variables chosen for the
experiment was the predetermined safety radius for each vehicle, the size of the environment and the maximum velocity a vehicle can achieve. The initial and goal location of each vehicle is generated randomly at the start of the program. The path followed by every vehicle was the dependent variable. The minimum closest distance between the vehicles can be extracted from these paths.

The experiments were run on 2016 Dell XPS desktop, 2013 MacBook Pro and 2016 Lenovo Ideapad. The type of computer had no noticeable impact on the experiments. Figure 5.1 shows two snapshots during a representative simulation of a 1000m x 1000m x 500m volume airspace. In this particular run, aircraft fly at constant speed, chosen with uniform probability between 10 m/s and 30 m/s. The airspace is quite dense, with several vehicles separated by less than 100 meters. The safety radius for these experiments was 50 meters.

5.2.2 Results

Figure 5.2 shows the path followed by all the vehicles after the simulation was complete. We can see that most of the paths are curves rather than straight lines. This is due to the fact that the vehicles are avoiding nearby vehicles while following the curved path. Figure 5.3 shows the number of aircraft active at any one time. Figure 5.4 shows the distances of closest approach for the same simulation. All aircraft have safety radius of 50 meters. We can see that none of the plots show a minimum separation distance of zero, thus validating our hypothesis that there
Figure 5.1. The diagram shows a snapshot of the nominal paths and vehicle locations at two different time stamps. The red open circles show vehicles that are within 100 meters of a nearest neighbor. Although there were many instances where avoidance maneuvers based on potential flow occurred, there were no collisions.

Figure 5.2. The plot shows the path taken by all the vehicles in the simulation experiment. We can see that some portion of these paths are curved. This indicated vehicles are trying to avoid other obstacle vehicles.
Figure 5.3. The figure shows a plot of number of active aircraft with respect to the simulation time. As random start and goal locations are chosen, not all the aircraft are active at one time. The maximum number of active aircraft here is 50.

There were no collisions. Clearly there are cases where safety zones are violated. This is due to the low-order path following controller. However, even with this low-order controller, the nearest approach distance between any two aircraft is 30 meters. A higher-order controller (for example, one that computes turn rates based on curvature of a local streamline) is likely to give better results.
Figure 5.4. The figure shows the plot of separation distance between all the vehicles with time. We can see that none of the plots reach a separation distance of 0. An indication of no collisions.

5.2.3 Comparison with the state-of-the-art

The average vehicle density for this experiment was 90 vehicles per cubic kilometer. For comparison, current operations at commercial airports typically space incoming aircraft three nautical miles in trail for approach to a single runway and 1.5 nautical miles in trail for approach parallel runways that are 2500 feet (760 meters) apart. This is equivalent to 1.6 aircraft per cubic kilometer if 300 meters vertical separation is maintained. As a measure of traffic density, note that increasing the safety radius
to only 125 m would result in a densely packed world, with aircraft unable to move at all. Note that this could be used as a method to determine when traffic metering should occur: when the packing fraction approaches densely packed, take-offs should be metered. Further, as a (very) imprecise measure of scalability, BusyWorld runs in roughly real time in MatLab on a 2013 MacBook Pro, with no effort spent on optimizing code - i.e. no vectorization, no parallel processing.

5.3 Physical Experiments

The system described in chapter 4 was used to conduct two physical experiments, one in which the airspace contained two vehicles and the second with an airspace of three vehicles. Random start positions were generated and each vehicle flew from the start position to a goal position which was created to force the vehicles to cross paths. The distance of closest approach was recorded using the Vicon motion tracking system within a space of 6m x 6m x 5m. Figure 5.5 shows three aerial vehicles flying in the motion capture studio.

5.3.1 Experimental parameters

Using the setup and algorithms described in chapters 3 and 4, we hypothesized that there would be no collisions between aerial vehicles for the physical experiments. The dependent variable for the experiment was the distance of closest approach and the number of collisions. The initial and goal location of each vehicle was
chosen to put the vehicles on the shortest path that would result in a collision. The minimum closest distance between the vehicles was extracted from these paths.

The safety radius chosen for all of the physical experiments was 0.5 meters. The random start and goal locations were chosen for the aerial vehicles by the experimenter. At the start of the experiment, the aerial vehicles were placed at their initial position. The collision avoidance program on the ground station was then started.

A 2016 Lenovo Ideapad computer served as the ground station for this experiment. The computer was connected to the aerial vehicles over WIFI, through
Figure 5.6. A plot showing path of two vehicles in a physical experiment. The colored lines show the path of the two vehicles. The time stamps indicate vehicle position at a particular time. Although the vehicles cross paths, the crossing happens at different time step.

As an initial experiment, two drones were flown between multiple sets of start and goal locations (selected to ensure a collision if each drone flew the shortest path between its points). A total of eight tests were performed with two drones. Figure 5.6 shows flight paths for one of the flights and figure 5.7 shows the separation.
Figure 5.7. Separation plots of two vehicle experiments. The blue line shows the separation distances for the path plots above. The grey lines indicate the separation results from other trials.

distance over time for all flights. Note that the colored line of the distance of closest approach plot corresponds to the flight paths in the upper plot. The minimum nearest separation distance was 0.66 meters; the average nearest separation distance was 1.13 meters.

Figure 5.8 shows the snapshots from a video captured during a sample two drone experiment. Figure 5.9 shows the velocity plots of the two drone experiment. Note that the first few velocity plots have a very high magnitude and undesired direction. The initial velocity vectors should be pointing towards goal. This anomalous initial
behavior occurred for each of the eight tests we conducted at the start of the experiment (figure 5.10 zooms in on the initial vehicle velocity vectors). This initial anomalous velocity occurred because the Kalman filter is initialized with the real time position and velocity values obtained during the experiment. The initial velocity of vehicles, however, is incorrectly estimated. Figure 5.12 shows the initial velocity values recorded during the experiment. We see that first value is on the order of thousands as compared to the remaining values. These values...
Figure 5.9. The figure shows quiver(velocity) plots along with the paths for a two drone experiment. The arrows represent velocity magnitudes at all time stamps. Red color plots shows the velocity and path of drone 1 and green color shows the velocity and path of drone 2. The time stamps are represented by t1, t2, and so on.

Figure 5.10. The figure shows initial velocity vectors from the figure 5.9. Left figure represents the velocity vectors of drone 1 and right figure represents velocity vectors for drone 2.
Figure 5.11. The figure shows velocity plot of a two drone experiments. Red plot shows the velocities and path followed by the drone1. Green plot represents the same for drone 2.

Figure 5.12. The figure shows initial velocity data recorded during a two drone experiment.
Figure 5.13. A plot showing path of three vehicles in the physical experiment. The colored lines show the path of the three vehicles. The time stamps indicate vehicle position at a particular time. Although the vehicles cross paths, the crossing happens at different time steps.

were used to initialize Kalman filter and hence effect the start of the experiment (figure 5.9). Furthermore, in figure 5.10, we see the smoothing behavior of the Kalman filter over next few measurements. These high initial velocity estimates occur because the position is divided by a very small time difference at the start of the experiment. Since a very little computation occurs initially, the time difference used to calculate velocity of drones is several orders of magnitude smaller than the standard time calculation. This problem can be fixed by initializing the Kalman filter with zero for velocity at the beginning of experiment. Unfortunately this error was not recognized until after all the experiments were conducted. It is important to note, however, that this initialization error did not effect the collision avoidance portion of the experiment or our conclusions. Figure 5.11 shows the two drone
Figure 5.14. Separation plots of three vehicle experiments. The one in colored lines shows the separation distances for the path plots above. The grey lines indicate the separation results from other trials.

velocity plots when the initial few estimates from Kalman filter are ignored. We can see that initially the magnitude of velocities is high and it decrease as the vehicle approaches the goal and this behavior is in accordance with our theory and hypothesis.

5.3.3 Three drone experimental results

Our demonstration of scalability was limited by the number of drones on hand (3). Tests of three Parrot AR drones flying between waypoints, again chosen with the
Figure 5.15. The figure shows snapshots of three drone experiments. The snapshots show the location of the drones at time stamps $t_0$ (initial location), $t_1$, $t_2$ and so on.

The constraint that a straight-line path between waypoints would result in collision, also demonstrated consistent, safe, operation. Figure 5.13 shows the flight paths for one of the flights and Figure 5.14 shows the separation distance over time for all flights. Note again that the colored line is the distance of closest approach plot corresponds to the flight paths in 5.13. The minimum nearest separation distance was 0.88 meters; the average nearest separation distance was 1.98 meters, significantly greater than the specified safe radius of 0.5 meters.
Figure 5.16. The figure shows quiver(velocity) plots along with the paths for a two drone experiment. The arrows represent velocity magnitudes at all time stamps. Red color plots shows the velocity and path of drone 1, green color shows the velocity and path of drone 2, blue color shows the velocity and path of drone 3. The time stamps are represented by t1, t2, and so on.

Figure 5.17. The figure shows initial velocity vectors from the figure 5.16. Left figure represents the velocity vectors of drone 1, middle figure represents velocity vectors for drone 2 and right figure represents velocity vectors for drone 3. Again, the time stamps are represented by t1, t2 and so on.
Figure 5.18. The figure shows velocity plot of a three drone experiments. Red plot shows the velocities and path followed by the drone1. Green and blue plots represent the same for drone 2 and drone 3 respectively. The time stamps are represented by t1, t2, and so on.

<table>
<thead>
<tr>
<th>Vx</th>
<th>Vy</th>
<th>Vz</th>
</tr>
</thead>
<tbody>
<tr>
<td>211.326016</td>
<td>99.337854</td>
<td>127.608270</td>
</tr>
<tr>
<td>0.091749</td>
<td>0.129565</td>
<td>0.116120</td>
</tr>
<tr>
<td>0.159017</td>
<td>0.303471</td>
<td>-0.953304</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vx</th>
<th>Vy</th>
<th>Vz</th>
</tr>
</thead>
<tbody>
<tr>
<td>187.234411</td>
<td>619.507583</td>
<td>93.416640</td>
</tr>
<tr>
<td>-0.375293</td>
<td>0.736807</td>
<td>-0.260625</td>
</tr>
<tr>
<td>-1.324859</td>
<td>2.344401</td>
<td>-0.934253</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vx</th>
<th>Vy</th>
<th>Vz</th>
</tr>
</thead>
<tbody>
<tr>
<td>-90.531583</td>
<td>349.446406</td>
<td>73.875634</td>
</tr>
<tr>
<td>0.356955</td>
<td>0.004730</td>
<td>-0.392701</td>
</tr>
<tr>
<td>0.985221</td>
<td>-0.170804</td>
<td>-0.914686</td>
</tr>
</tbody>
</table>

Figure 5.19. The figure shows initial velocity data recorded during a three drone experiment.
Figure 5.15 shows snapshots from a video captured during a sample three drone experiment. Figure 5.16 shows the velocity plots of the three drone experiment. As with case of two drones, we can see that first few velocity plots have a very high magnitude and undesired direction (figure 5.17). Figure 5.19 shows the initial velocity values recorded during the experiment. We see that first value is on order of thousands as compared to other values as in the case with two drone experiments. The same reason as discussed for the two drone experiment applies here, an incorrectly initialized Kalman filter leads to the initial incorrect vehicle velocity. Again, this mistake did not affect the collision avoidance part of the experiments. Figure 5.18 shows the two drone velocity plots when the initial few estimates from the Kalman filter are ignored. We can see that initially the magnitude of velocities is high and decrease as the vehicle approaches the goal.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Average Closest</th>
<th>Minimum Closest</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 UAVs(8 trials)</td>
<td>1.13m</td>
<td>0.66m</td>
</tr>
<tr>
<td>3 UAVs(9 trials)</td>
<td>1.98m</td>
<td>0.88m</td>
</tr>
</tbody>
</table>

Table 5.1 shows the average minimum distance and the minimum closest distance during the flight for all the experiments. Again, it can be seen that the minimum distance for the physical experiments is above the safety radius of 0.5 meters. Also, we can see that the average closest distance is above 1 meter which supports our hypothesis of no collisions.
5.3.4 Discussion and interpretation

As stated in chapter 3, the navigational velocity field developed in this thesis is guaranteed to be collision free. From the results discussed in previous subsections, it has been shown that there were few collisions in either the simulation or the physical experiments. The closest approach in the physical experiments was greater than the safety radius specified for spherical collision avoidance. Moreover, for all the trials we see that the method was able to maintain a safety separation distance between the vehicles. These results show that the velocity based approach has practicable applications for collision avoidance.

Figure 5.20 shows a zoomed in part of figure 5.18 which contains the velocity followed by two aerial vehicles in the three drone experiment. The purpose of the figure is to examine the collision avoidance theory discussed in chapters 2 and 3. The green and blue vectors represent the collision avoidance of the two vehicles. Note the magnitudes and direction of the velocity vector at various time steps. At time stamp t2, we see that green vectors are pointing towards the vehicle’s goal and the magnitude is high. As the vehicle approaches the other vehicle, the next few velocity vectors indicate that the vehicle is changing its path. The same can be observed for the vehicle represented by blue vectors. At time stamp t3, the vehicles are closest and we can see that the vehicle represented by the green vectors is moving away from vehicle represented by blue vectors. At time stamp t4, the vehicle represented in green again follows the shortest path to the goal.
Figure 5.20. The figure shows velocity plot of drones 2 and 3 extracted out of figure 5.18. The green and blue vectors represent the velocity of the two drones at every time step.

Moreover, from both the plots, we see that the velocity magnitudes are continuously decreasing as vehicle is approaching the goal. This is in accordance with the theoretical ideas discussed in chapter 2. The plot also suggests that the implementation and controller design discussed in chapter 4 operated as expected. As the error keeps on decreasing, the velocity commands generated by the controller also decreases.
5.4 Summary

This chapter has presented simulation and physical experimental results obtained by implementing the methods discussed in chapter 3. The simulation results described using plots of vehicle paths, their separation distance were presented. These results were also compared to a modern air traffic management system.

The chapter also presented the results from physical experiments with two vehicles and three vehicles. We find that the minimum and average separation distances for all the experiments were greater than the safety distance specified. The next chapter offers concluding remarks including an overall summary of contributions and future avenues of this research.
Chapter 6  
Conclusions

This thesis has presented a method for collision avoidance in high-density air traffic scenarios based on potential fields. Potential field methods are scalable, can be distributed, and a real-time implementation is straightforward. A known drawback of potential fields is the existence of local minima, but this can be addressed through the implementation of rules of flight to break symmetries [3]. The results of simulations in a high density airspace (90 aircraft per cubic kilometer) as well as hardware demonstrations with multiple UAVs have been presented. This chapter will first summarize the contributions, limitations and assumptions for this work. Future avenues of this research are then presented in the second half of the chapter.

6.1 Summary of Contributions

The key contributions of this thesis are:
• **Velocity based approach:** The thesis developed a velocity based approach for collision avoidance of multiple aerial vehicles. The velocity based approach is derived from potential field based methods proposed in 1980s. The method generates collision free navigational velocity commands for all the vehicles in the airspace. This was demonstrated in simulation as well as in physical experiments.

• **Scalability:** In simulation, scalability is achieved for up to 50 vehicles flying in the airspace. For physical experiments, we demonstrated scalability of the algorithm for up to three vehicles.

• **Online collision avoidance:** The thesis presented a successful implementation of collision avoidance in an online fashion. The results for online collision avoidance were presented in chapter 5.

### 6.2 Limitations and Assumptions

As with other methods, this velocity based approach has its own limitations. These limitations are:

• **Robust velocity estimate:** The approach relies heavily on knowing the velocity of the other vehicles, the algorithm requires information about the velocity of all the vehicles at all times. For indoor applications this problem is solved by using the Vicon motion capture system. Given the current status
of the Next Generation Air Transportation System (NextGen), it is likely that ADS-B will be required for all aircraft operating in the national airspace, and we can rely on the availability of ADS-B to provide position/velocity information for collision avoidance in outdoor scenarios [37].

- **Stagnation Streamline**: Given the set of initial configuration of the robot, one of the paths that robot can follow lies on stagnation streamline. There could be collisions if the vehicle follows that path. We never observed this situation in the physical experiments because of noise present in the system. This problem can be fixed by breaking the rule of symmetry and the vehicle can follow any other path apart from the one stagnation streamline.

### 6.3 Recommendations for Future Work

The work presented here opens up new avenues for future research in aerial vehicle collision avoidance. Some of these are recommendations are:

- **Multiple number of vehicles**: Given the scalability and efficiency of the system, the work can be expanded for multiple aerial vehicles in a physical experiment. In simulation, we showed that this method works for a high density airspace (90 aircraft per cubic kilometers). Future work should aim at achieving similar density for physical experiments.

- **Elliptical avoidance**: The work discussed here used spherical avoidance
zone around a vehicle to avoid the incoming obstacle vehicle. This can be further modified to create an elliptical avoidance zone so as to allow more freedom and space for other vehicles in the airspace. This should help to reduce collisions for very high density airspaces.

- **Real world implementation:** This work has examined the use of a homogeneous set of vehicles indoors. Future research will focus on implementing the system in an outdoor environment with vehicles varying in size, type and controllability. The important factors to be considered for outdoor implementation are the sensors to be used, the type and size of vehicle, etc. The controller design presented in the thesis will have to be modified so as to accommodate real world uncertainties such as wind gusts, weather changes, etc.

The thesis has presented a velocity based approach for collision avoidance for multiple vehicle scenarios. The method is implemented in simulation in MATLAB and was tested on aerial vehicles in an indoor environment. Results in simulation as well as for the physical experiments showed that the method is scalable and can be scaled up to 50 vehicles in simulation. As the number of aircraft are increasing day by day, we will need a system to effectively manage these aircraft. We believe that this method could serve as a basis for collision avoidance control and airspace regulation integration for dense heterogeneous airspaces.
Appendix A

Experiment Procedure

A.1 Introduction

This appendix describes the experimental procedure for running the physical drone experiments. The experiments were run in a Vicon motion capture environment. This Appendix describes the system requirements, the configuration of hardware components, and the pre-experiment procedures.

A.2 System Requirements

The components required for running the experiments are:

- A personal computer with either Windows, Linux or MacOS operating system.
  The computer acts as ground station. The system should have telnet and MATLAB softwares installed.
A router. For the experiments conducted, we used Linksys WRT54GL.

- Multiple Parrot AR Drone 2.0

- Vicon Motion Capture system

### A.3 Hardware Configuration

For running the physical experiments, AR drones and a router are two hardware components which require configuration. The configuration is described in next few subsections.

#### A.3.1 Router Configuration

A router is used to connect multiple drones to the ground station. The router is configured to have the IP address of 192.168.1.1 on subnet 255.255.255.0. Router configuration for the experiments is as shown in the figure A.1.

#### A.3.2 AR Drone Configuration

We created a file called wifi.sh on the drone which will make the drone connect to pre-defined WIFI network whenever triggered. The steps are as follows:

- Connect to drone’s wifi network.

- Telnet to the drone computer by using the below command:
Figure A.1. The figure shows a snapshot of router configuration which was used to run physical multi drone experiments.

C:\Users\sagar>telnet 192.168.1.1

Trying 192.168.1.1...

Connected to 192.168.1.1.

Escape character is '^]'.

74
BusyBox v1.14.0 () built-in shell (ash)

Enter 'help' for a list of built-in commands.

- Create a file named wifi.sh in /data/ directory on the drone:

  vi /data/wifi.sh

- Change network SSID in line 3 and IP number in line 4 of the following code, then copy and paste it into the file. The SSID is what you set up in router configuration. The IP should be unique for each drone from the reserved range.

  killall udhcpd

  ifconfig ath0 down

  iwconfig ath0 mode managed essid linksys

  ifconfig ath0 192.168.1.10 netmask 255.255.255.0 up

- Make the newly created file executable

  chmod +x /data/wifi.sh
• Exit the telnet connection

    exit

These steps have to be implemented to configure each individual AR drone. The IP specified in step 3 must be unique for each drone. This is the IP that we will later use in our MATLAB code to connect to the drones.

### A.4 Pre-experimental procedure

Before the experiment begins, the following steps are followed:

• Calibrate the Vicon motion capture system using tracker software.

• Connect the computer in motion capture lab to the ground station using the Ethernet cable.

• Place the drones in the desired initial location.

• Connect to each drone over wifi and run below commands in command prompt.

    telnet 192.168.1.1

    ./data/wifi.sh
This step asks the drone to connect to your router.

- Connect to wifi router and check pings for each and every drone for the IP address specified in wifi.sh file.

- Given the pings are successful, we can run the file Runbusyworld_2 in MATLAB. Before running the file specify the goal location for each vehicle. Goal locations have to be set as to ensure collisions. Make sure to chose a different name for log file for every experiment. So modify the name of the log files, if required.
Bibliography


[34] Ng, W. S. and E. Sharlin (2011) “Collocated interaction with flying robots,” in International Symposium on Robot and Human Interactive Communication (RO-MAN), Atlanta, GA, USA, IEEE, pp. 143–149.

