Optimal Algorithm Design for Transfer Path Planning for Unmanned Aerial Vehicles

by

Andrew George Pollock

A thesis submitted in partial fulfilment for the degree of Doctor of Philosophy

in

Aerospace Sciences Research Division
School of Engineering

September 2014

University of Glasgow

Abstract

Aerospace Sciences Research Division
School of Engineering

Doctor of Philosophy

by Andrew George Pollock

Over the past three decades unmanned aerial vehicles (UAV) have seen significant development with a current focus on automation. The main area of development that is pushing automation is that of path planning allowing a UAV to generate its own path information that it can then follow to carry out its mission. Little work however has been carried out on transfer path planning. This work attempts to address this shortcoming by developing optimal algorithms for a path planning task to move on to a circular flightpath to carry out a target tracking mission.

The work is developed in three main sections. Firstly the transfer algorithm itself is derived including gradient analysis for the cost function being applied, adaptation of this cost function into two separate minimising actions and analysis of a cost function issue that introduces a separation distance constraint. The algorithm is tested proving correct constraint activation and cost selection. The second part of this work looks at validating the results of the transfer algorithm against the Dubin's car result and a receding horizon approach when applied to the transfer operation. Utilising the cost results from the transfer algorithm an efficiency analysis against the equivalent costs from the other methods is carried out. Lastly this work looks at the comparison between the developed transfer algorithm and a more flexible transfer approach by developing a new cost function form. A switching cost function is introduced where environmental parameters from the target tracking mission (i.e target position and velocity) are used to switch between a number of applicable cost functions (time minimal, distance minimal and minimum speed transfer). An analysis is carried out to investigate the performance of both the original algorithm and the newly developed switching function based on key target tracking parameters . . .

Declaration of Authorship

I, Andrew George Pollock, declare that this thesis titled, 'Optimal Algorithm Design for Transfer Path Planning for Unmanned Aerial Vehicles' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: [illegible]

Date: 09 September 2014

"I love deadlines. I especially like the whooshing sound they make as they go flying by."

Dilbert (by Scott Adams)

Acknowledgements

My thanks goes to my supervisors Dr. Jongrae Kim and Dr. Euan McGookin for their continued guidance and support throughout this research, as it would never have began without them and certainly never continued to its completion without their technical expertise and their ability to listen to me rant.

In addition I would like to thank the Aerospace Sciences Division within the School of Engineering at the University of Glasgow as without them this research would not have been possible. I would also like to thank the EPSRC for their funding of this work.

I would also like to thank Dr Dave Anderson and Dr Douglas Thomson for their guidance, assistance, support and advice throughout this project.

A huge thank you goes to my family, my mum, Elizabeth Pollock and dad, George Pollock for believing in me and for the support and encouragement they have given throughout this work.

A final special thanks goes to Keith Armstrong, without whom this work would of never been finished. For his ability to cheer me up when I am down and to give me a kick when I'm being stupid. . .

Contents

List of Figures

List of Tables

Abbreviations

UAV	**U**nmannned **A**erial **V**ehicle
TPBV	**T**wo **P**oint **B**oundary **V**alue
wrt	**w**ith **r**espect **t**o
LOS	**L**ine **o**f **s**ight

Symbols

t_0	Initial Time	s
t_f	Final Time	s
0_2	2x2 Zero Matrix	
I_2	2x2 Identity Matrix	
x^B	Body x Coordinates	m
y^B	Body y Coordinates	m
x	Inertial x Coordinates	m
y	Inertial y Coordinates	m
v_x^B	Body x Velocity	m/s
v_y^B	Body y Velocity	m/s
v_x	Inertial x Velocity	m/s
v_y	Inertial y Velocity	m/s
u_x^B	Body x Control	m/s^2
u_y^B	Body y Control	m/s^2
u_x	Inertial x Control	m/s^2
u_y	Inertial y Control	m/s^2
V_{max}	Max Velocity	m/s
V_{min}	Min Velocity	m/s
U_{max}	Max Control	m/s^2
U_{min}	Min Control	m/s^2
$\mathscr{H}$	Hamiltonian	

For my Gran. She isn't here to see the end of this, but she at least saw the beginning. I know this would make her proud...

Chapter 1

Introduction

1.1 Preface

Over the past three decades unmanned aerial vehicles (UAV) have seen significant development and are now being utilised for a number of different tasks [1]. Due to the adaptability of UAVs they are readily used by many different organisations worldwide. The drive to remove the human factor has pushed UAV development throughout history and is continuing to do so with the advent of autonomous UAV platforms and systems.

UAVs and UAV research have become hot topics over recent years stemming from requirements to provide remote operating platforms for many different tasks. Originally used for military applications such as surveillance and reconnaissance, recent developments have seen the growth of UAV use in the civilian sector as well for tasks including search and rescue, police use [2] and even weather monitoring [3].

Modern UAVs are capable of extended operational time scales (in the region of days) so having a UAV that can carry out tasks automatically and react to changes in its environment without human intervention is highly beneficial. To this end recent UAV research has seen a shift towards the development of algorithms that can plan and perform UAV manoeuvres to carry out missions. Some of the more common operations include target tracking, collision avoidance and area reconnaissance. These algorithms are all examples of path planning where the goal is to produce the path that the UAV will fly to perform its set mission.

1.2 UAV autonomy and the importance of path planning

The majority of operational UAV control is still carried out by a remote operator physically controlling the UAV from a base station. The next generation of UAVs however aim to take this to the next level. By providing a UAV with the ability to sense its surroundings and analyse the information it is possible for the UAV to generate its own control decisions based on mission parameters and inherent design limitations. If you consider the situation that automation is trying to replicate; a pilot flying an aircraft; the steps to full automation appear more complex than at first glance. Consider an obstacle avoidance task: A pilot is capable of seeing an obstacle and carrying out an avoidance manoeuvre and then continuing with the mission at hand. For an autonomous UAV (AUAV) this task is more complex: the AUAV first has to have some method through which to view its surroundings (usually some form of camera is used), it then has to take this information and identify the potential obstacles, next it must select an appropriate control action to avoid the obstacle and finally it has to recover from this avoidance manoeuvre back to its normal operating mode. A similar situation occurs for a UAV simply trying to travel between two points. The UAV is likely to be programmed with its desired destination but then is required to work out a way to travel from its current position to its destination and arrive in the correct configuration to carry out its next mission phase. It is also imperative that the flight limitations of the UAV are not broken during the transfer as this could result in the loss of the vehicle. For a human operator these tasks are relatively simple whereas it is difficult for a UAV.

The majority of automated tasks have one key component in common, that there exists a path available to the UAV that allows it to carry out its mission. Path planning underpins the degree of autonomy that a UAV can achieve. Without proper path information it is impossible for a UAV to be expected to be able to carry out its mission as it simply does not know how it should be flying. The starting point for flying any mission is the reference path information that the UAV needs to successfully complete mission phases and transfer between them.

It is desired to find the best path for any given operation. It is therefore safe to assume that there exists an optimal path for given mission parameters and UAV constraints. Optimal path planning therefore has important implications on the ability of such vehicles to perform missions successfully [4–8]. The ability to generate mission useful paths is key to the level of autonomy that a UAV can achieve.

1.3 Why transfer path planning?

It is useful to think of UAV automation in the form of a jigsaw puzzle with different pieces representing an algorithm designed to carry out a specific task. To complete this puzzle it would be expected that there exists an algorithm for each task that a UAV must carry out to complete a mission. Some of the pieces already exist with a lot of research having been carried out on what can be seen as the most desirable tasks, for example, collision avoidance or target tracking. Equally some tasks have been understudied such as algorithms for tasks like altitude changes as they are relatively trivial with few expected complexities. From research one such understudied task is that of transfer path planning. This made this task interesting for study as providing a path planning algorithm that allowed a UAV to fly to a destination is a key component in UAV automation. For example, if a UAV is on a mission to perform target tracking within a defined region the first question to be considered before thinking about the target tracking component of the mission is how should the UAV move from its current position to the desired region of interest? This is the question that this work investigated by attempting to solve a transfer problem for a UAV moving to a target tracking mission.

1.4 Objectives

The main objectives of this work can be summarised as:

- To research and design using optimal techniques a globally optimal path planning algorithm that carries out a transfer for a UAV from a known start point to an end configuration of a circular flight path used for target tracking.
- To compare the results from the developed algorithm against those of similar algorithms to obtain information about the developed algorithms efficiency
- Develop a new cost function form that allows switching between a number of appropriate cost functions depending on external stimulus.
- To compare the effect on key target tracking mission parameters of utilizing the switching cost function or the developed transfer algorithm.

1.5 Contribution to the field

This project is aimed at furthering the development of path planning algorithms that can be utilised for autonomous UAV operations. These contributions can be seen as:

- The development of a path planning algorithm that produces globally optimal solutions for a UAV transfer manoeuvre where the shortest path is the desired goal.
- An analysis of cost function gradient information for the above transfer manoeuvre.
- A study of cost function suitability for transfer path planning.
- A comparison of developed algorithm output against existing methods.
- The development of a switching cost function.
- Comparison of the effect of utilizing the a single cost function transfer algorithm against a that of a multi-cost function transfer algorithm for target tracking.

1.6 Thesis Structure

The focus of this work is to design and analyse algorithms for path planning for a UAV attempting to transfer between a known start point and a desired end configuration. This was carried out utilising MATLAB for algorithm coding and subsequent analysis of the designed algorithms' performance. The goal for the main parts of this work is to find globally optimal solutions to a path planning problem, however it should be noted that in this case globally optimal is inferred as globally optimal within the tolerances of the developed algorithms as these can effect the accuracy of solution. The following outlines how this work was achieved.

A discussion of the history of the field is shown in Chapter 2. Due to the large scope of UAV development the chapter first discusses UAV advancements throughout history including a brief description of common terminology. The chapter culminates in a discussion of a few of the most pertinent works on path planning in recent history and a look at some of the modern path planning methods readily applied today. It details the target tracking example used by this work when developing the transfer algorithms and details the choice of path planning methods to be used including the motivation and details behind them.

Chapter 3 discusses the development of a globally optimal algorithm that underpins the remaining analysis and work of this thesis. The mathematics for the design of two algorithms are considered the first using continuous equations the second using discrete equations. The development of a Hamiltonian based algorithm is shown and how this leads to the development of a shooting method algorithm indicated. This includes an analysis of cost function gradients and the effect these have on the algorithm design.

The necessity of correctly selecting the cost function for this application is also discussed and the test results for the algorithm presented.

Chapter 4 is a comparison of the transfer paths from the developed algorithm and the transfer paths from two similar algorithms. It builds on the work of Chapter 3 by investigating the performance improvement the developed method gives over existing solutions. The two algorithms used are the Dubin's Car approach and a receding horizon approach.

A comparison between the shooting algorithm developed in Chapter 3 and a flexible transfer using mission information for target tracking is presented in Chapter 5. Detailed is the development of a switching cost function including the switching conditions when applied to a target tracking example. The chapter details two key parameters applicable to the target tracking mission and compares the effect of the shooting and switching algorithms on these parameters.

Finally, Chapter 6 presents the conclusions of this work and includes a section discussing further work with possible algorithm adaptations.

Chapter 2

Background and Literature Review

2.1 Introduction

This chapter gives a brief history of UAV development from the original concept of flying bombs to the type of UAV drones seen today. Next a look at common UAV terminology and current examples of operational UAVs will be detailed. The chapter will culminate by looking at UAV automation including a brief look at the history of the path planning techniques central to this research with a basic formulation of the problem presented and a summary of some of the path planning methods available to solve the posed problem.

2.2 UAV History

This section provides a brief history of UAV development to the UAV platforms recognised in the present day. The majority of UAV development was driven by the need for military advancement during times of war when the necessity for finding technology that limited or removed the risk of harm to humans became paramount. The information for this section is taken from the following: [9–12]

2.2.1 Early Development

This era can be seen as the time preceding the First World War when the first recorded use of unmanned aircraft were kites and balloons dating as early as the 19th century. One of the earliest recorded uses was during the Italian Wars of Unification when, in August

1849, the Austrians used balloons to carry explosive payloads into Venice. Although these were unmanned they had no guidance control and frequently would be blown towards friendly forces. In addition the balloons had a limited range as the explosives required an attached detonator wire. This war balloon idea was later extended for use during the American Civil War when hot air balloons carried explosives with a timed release. The 'Perley Bomber' as they were known, named after inventor 'Charles Perley', had similar draw-backs to the Austrian balloons given their lack of guidance control. This again resulted in 'friendly fire' incidents caused by unfavourable wind conditions blowing the balloons back over friendly lines.

The use of unmanned aerial vehicles for reconnaissance first seemed feasible in 1883, when kites with mounted cameras were used by a photographer to take low altitude photographs. An early military application was during the Spanish-American war in 1898 where kite-cameras were employed by the US Army to survey enemy positions and fortifications.

2.2.2 World War I and II

During the First World War the attack and surveillance capabilities of UAVs were of interest especially focusing on the idea of 'glide bombs', forerunners to the German V-1. Although all sides made advances in 'glide bomb' technology both the British and American forces also made developments in remote controlled vehicles. A British radio controlled aircraft was developed in 1917 while American efforts focussed on aerial torpedoes and the 'Kettering Bug' remote controlled bi-plane. However as the First World War ended these research programmes declined and interest was lost until the late 1930s and the advent of World War II.

British and American efforts stemmed from the need for gunnery practice targets. With the invention of the bomber aircraft the Royal Air Force(RAF) and Royal Navy required a means to test effective defence strategies for Navy warships against bombing runs. This resulted in the development of the 'Fairy Queen' target drone and later the development of the first recoverable and therefore reusable UAV, the 'Queen Bee', a radio controlled Tiger Moth aircraft (see Figure 2.1)

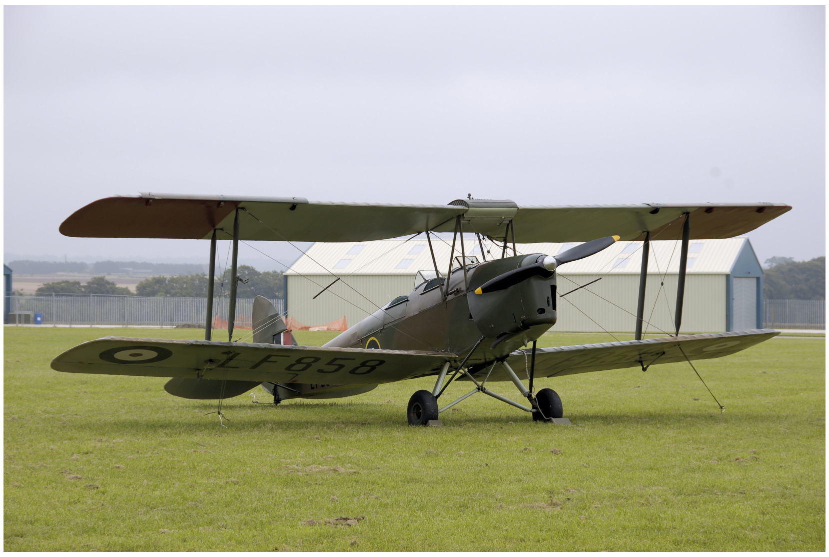

FIGURE 2.1: The first recoverable, reusable UAV the 'Queen Bee' which is a radio controlled version of the Tiger Moth aircraft

The Americans developed similar gunnery practice drones such as the 'RP-4 Radioplane' used by US services. There were also experiments with unmanned B-17 and B-24 bombers where the radio controls were set in flight before the crew would bale out from the aircraft. A suggested modification to this system was to use a television monitor attached to the bombers nose to allow better direction of the bomber onto its intended target.

German efforts between the 1930s and the end of World War II were considerably more advanced resulting in both the V-1 and the V-2 flying bombs. The advances in guidance and control arising from the German flying bombs fuelled the continued development during and after World War II when German technology transferred into Allied hands. This included American development of camera and Infrared (IR) guidance aircraft which could be seen as the start of the cruise missile and the continued UAV development into the 1950s and beyond.

2.2.3 1950s and the Vietnam War

In the 1950s the Americans developed the Teledyne Ryan Q-2 Firebee (see Figure 2.2). The adaptability and fully recoverable nature of the Firebee shows the beginning of the modern recognised concept of a UAV.

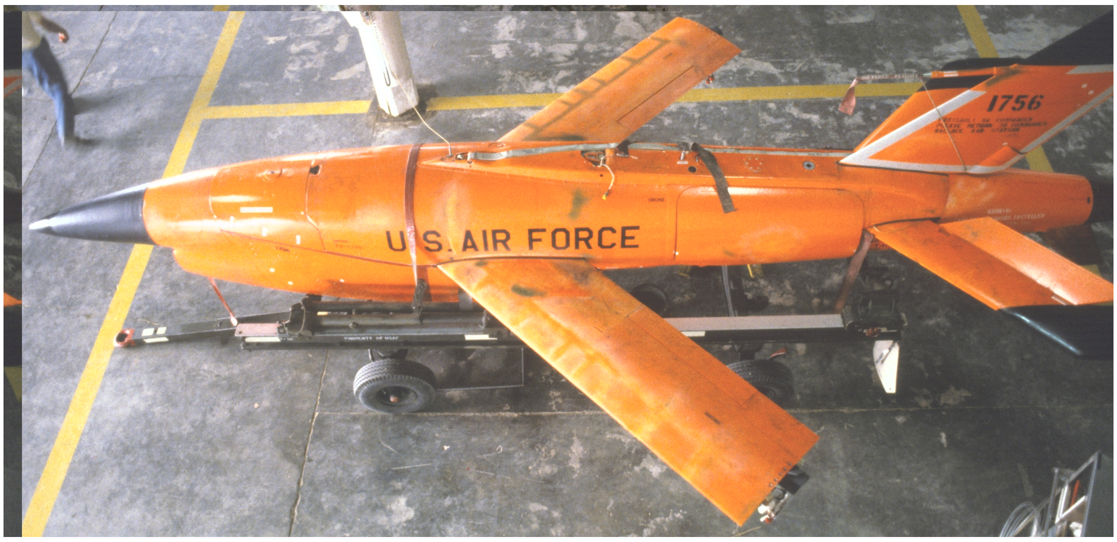

FIGURE 2.2: A Firebee drone deigned in the 1950s - 60s. This is the beginning of the concept of modern UAVs

Due to the loss of a U-2 spy plane in May 1960 and the resulting political implications with the Soviet Union along with the loss of several other manned reconnaissance aircraft the development of unmanned intelligence gathering systems became a top priority. The numerous variants of the Firebee were used heavily in the Vietnam War for many reconnaissance rolls and were also used as decoys to draw enemy fire away from manned aircraft. Some Firebees were modified to work as CHAFF dispensing drones and electronic counter measure platforms. Throughout the war the continued development of UAV systems saw the advent of drones capable of collecting and transmitting data, advancements in photographic reconnaissance including the equipping of some drones with IR to perform night operations. Near the end of the Vietnam war a few drones were adapted to carry missile payloads however the American withdrawal in 1968 saw a loss of funding for these experiments.

2.2.4 1970s to present day

The continued development of UAVs as attack platforms was carried out in Israel in the 1970s after the purchase of Firebee drones from the USA. These Firebees were primarily used for reconnaissance, decoy roles and as anti surface to air missile (SAM) platforms. In subsequent years Israel became world leaders in UAV design, with both the 'Scout' and through further development the 'Pioneer' UAVs. Both of these UAVs were either controlled via a ground station or through an autopilot system linked to a pre-defined programme. In the 1980s 20 Pioneer UAVs were bought by the US and were heavily used during Gulf War I (Figure 2.3). From the 1st Gulf War onwards research into

UAV capabilities both as attack and surveillance platforms have effectively taken off. An increasing onus on the autonomous capabilities of such platforms has been seen in recent years. Some of the modern day UAV examples will be discussed in the next section.

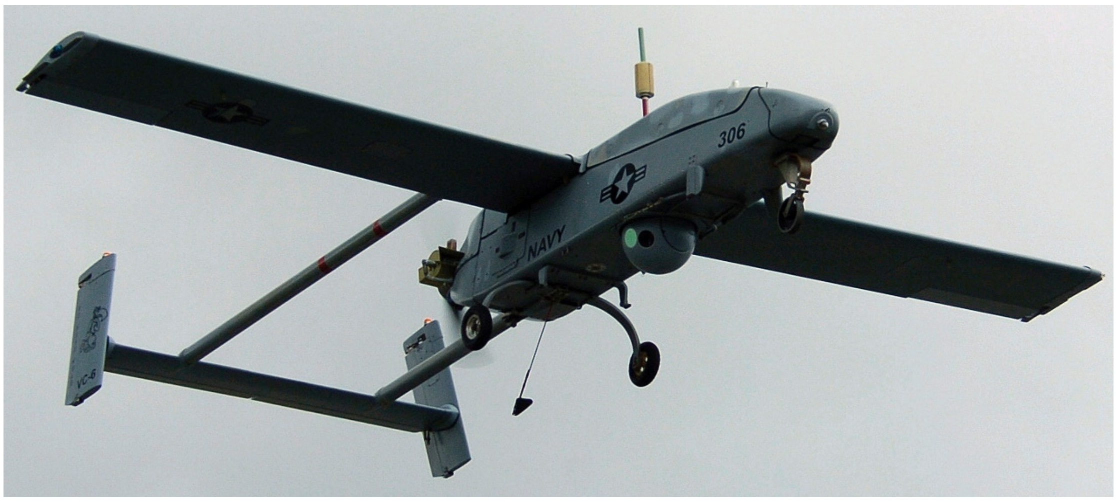

FIGURE 2.3: The Pioneer UAV, used heavily in Gulf War I

2.3 UAV terminology and examples

Over the course of UAV development many terms have arisen to describe different UAV types and configurations. Some of the more pertinent examples are detailed here:

Unmanned Aerial Vehicle (UAV): This term is used to describe any aerial vehicle that doesn't carry a human operator. This gives no indication of the intended role of the UAV, the UAV configuration or the whether the UAV is remotely controlled or autonomous in nature.

Unmanned Combat Aerial Vehicle (UCAV): This term is reserved for UAVs that are designed for combat capabilities. This type of UAV is particularly controversial especially when considering autonomous systems as there are moral and legal obligations regarding the right to fire for these types of system. Currently both the RAF and the USAAF have UCAVs in active deployment. The most recent variant in service is the 'MQ-9 Reaper' which is a remotely operated MALE UAV (see below) capable of carrying 4 Hellfire missiles and 2 laser guided bomb, Figure 2.4.

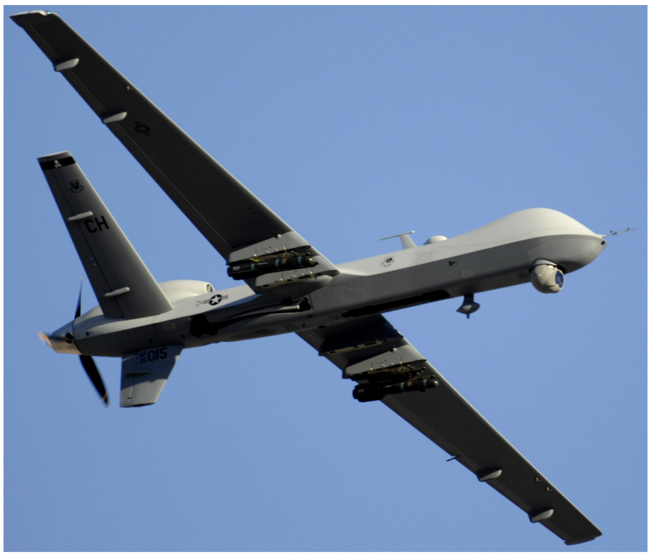

FIGURE 2.4: The MQ-9 Reaper UCAV currently in active service with both the RAF and the USAAF

Medium Altitude Long Endurance (MALE): UAVs that can operate up to 30000' with a 12-15 hour endurance.

High Altitude Long Endurance (HALE): UAVs that can operate above 30000' with an endurance of 24 hours or longer.

Unmanned Aerial System: This relatively new term refers to the entire system required to operate a UAV platform including the UAV itself, command/communication links and any associated control stations (these can either be ground, air or sea based stations). Sometimes this is also referred to as a UAVS (UAV System).

Fixed Wing UAV: This denotes the configuration of the UAV to be fixed wing such as that found on a conventional aircraft. This is the main type of UAV in use today due to the controllability and endurance of such platforms. The Watcheeeper UAV is an example of a fixed wing surveillance platform, Figure 2.5.

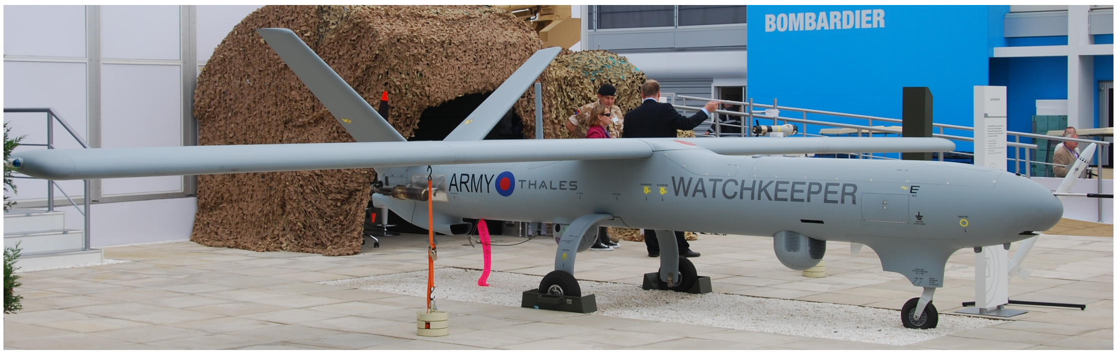

Figure 2.5: The Watchkeeper UAV, an example of a fixed wing surveillance platform

Rotary Wing UAV: This UAV is akin to vertical take off and landing platforms such as helicopters. These are subject to recent research as the ability to perform vertical take off and landings as well as the increased manoeuvrability and hover capabilities make them a more versatile UAV platform than standard fixed wing UAVs. Rotary UAVs are however harder to control especially autonomously resulting in numerous rotary wing research projects seen in academia and industry today. A typical research Quad-rotor can be seen in Figure 2.6.

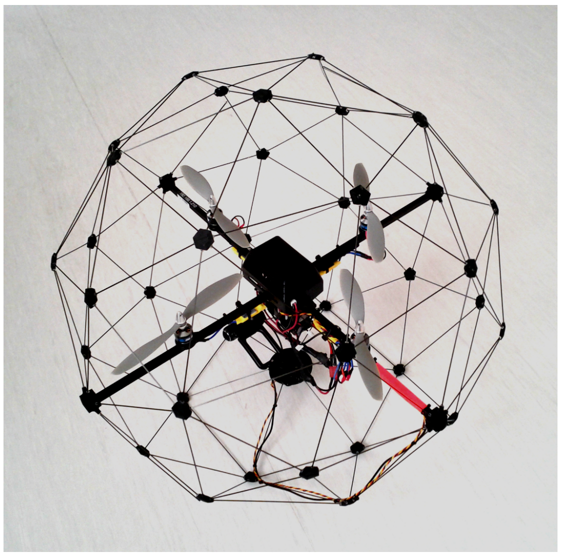

Figure 2.6: A Quad-rotor UAV as used in research at the University of Glasgow

Autonomous UAV (AUAV): This refers to any UAV this is capable of operation without the intervention of a human operator. The ability to send a UAV on a mission of up to and beyond 15 hours with minimal human intervention is desirable. Methods of autonomy are the subject of many research studies and attempts to improve and innovate in the field of UAV autonomy is a large area of UAV research today. The next section discusses the need and requirement for UAV autonomy focusing on the requirement of proper path planning and its effect on mission outcomes.

2.4 Brief History of Path Planning

In recent years the idea of path planning for UAVs has become a hot topic for research especially in areas such as target tracking. The majority of this work stems from research into path planning for small robotic vehicles and robot manipulators, initially beginning with the ability of such robots to navigate around obstacles (e.g. a robot manipulator moving an object from one area to another without hitting anything)[13–17].

Using the work carried out into path planning for ground vehicles as a basis, recent research is attempting to apply path planning techniques to aircraft [18]. The majority of this work focuses on the usage of optimisation formulations [19, 20].

An early attempt to find an optimal solution for a path connecting two points was carried out by L. E. Dubins [21]. This work was focussed on finding geodesics which are lines connecting an initial and final point that are minimum in length. Dubins restricted such paths to have a maximum curvature and found what he called sets of words that would result in geodesics. He found 6 possible words that would describe any geodesic: lrl, lsl, lsr, rlr, rsr and rsl, where l and r denote a go left or go right instruction respectively and s is a go straight command. The curvature for the left and right operations, or anticlockwise/clockwise rotation was restricted to that of a unit circle. Dubins however restricted his problem to a vehicle that could only travel forward. The word notation can be further simplified by realising that the movement can be classified as a CCC or a CSC form where C stands for curve (i.e. either a left or right rotation) and S is a straight line path.

This work was later extended in [16] when the ability to reverse was added, resulted in an extension from the original possible 6 words to 48 possible solutions that could be optimal. The work by Dubins and J. A. Reeds and L. A. Shepp has resulted in well known optimal path planning examples of Dubins Car and the Reeds Shepp Car which have been applied to numerous planning problems. They have become the basis of some of the modern day methods in an attempt to take the basic examples set out by Dubins, Reeds and Shepp and apply the work to more real life problems (i.e. adding other dimensions or removing some of the constraints set by Dubins, Reeds and Shepp). Examples of expansions of these works are [6, 18, 22, 23]

The development of path planning algorithms has driven the use of UAVs in reconnaissance roles due to their improved endurance and safety benefits compared to manned vehicles. This is no more true in the case of target tracking where a large amount of research is devoted to developing better algorithms for tracking moving targets [24]. The focus of this work aims to extend on this target tracking work by looking at new applications of path planning algorithms for transferring between target tracking scenarios.

2.5 Target Tracking Scenario in Detail

The target tracking scenarios in question come from the work carried out in [24] where a ground moving target is required to be tracked using a UAV, where the target is in a dense obstacle area. The following section is a summary of this work, with the Figures taken from [24] with the permission of the author.

The main objective was to position a UAV/UAVs in such a way that that the target was inside the field of view of the UAV's cameras, Figure 2.7.

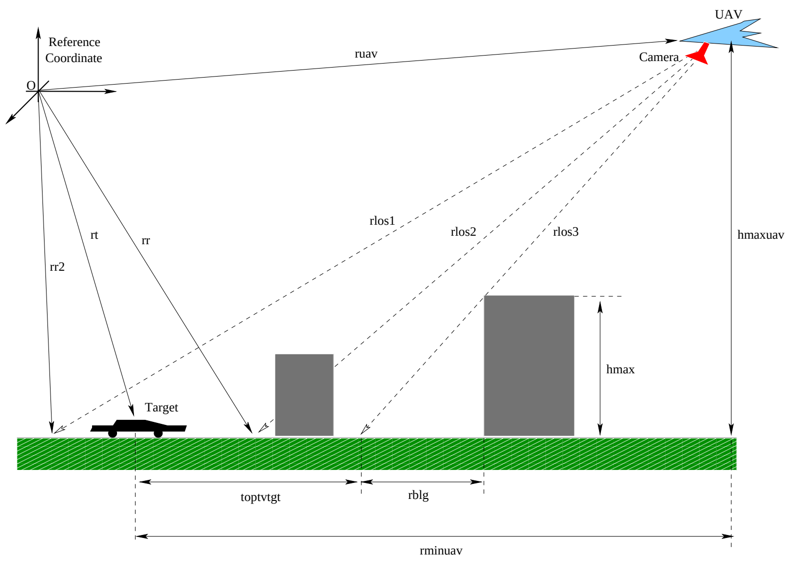

FIGURE 2.7: Moving ground target tracking using a UAV camera

Assuming the aircraft is a fixed wing UAV and that the ground obstacle locations and heights are known an example of the tracking scenario can be viewed in 3D and 2D space, Figure 2.8 and Figure 2.9, where the star indicates the target position, the circle an example of the UAVs flight path and buildings/obstacles are represented by the black boxes. If the altitude of the UAV is relatively close to the height of the highest ground obstacle it is safe to assume that at some point the line of sight (LOS) of the UAV's camera will be blocked.

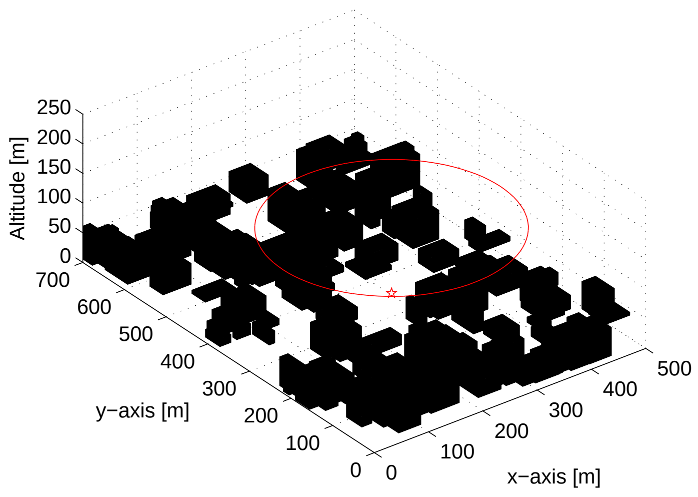

Figure 2.8: 3D view of operating area

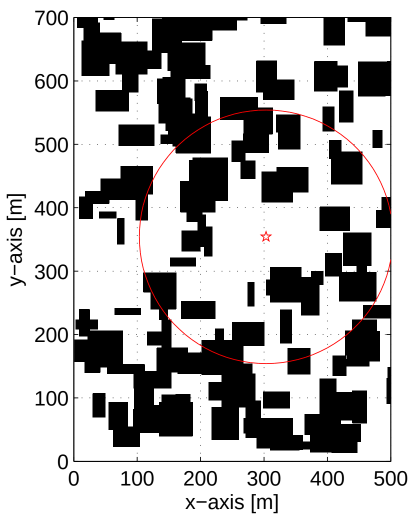

Figure 2.9: 2D view of operating area

The goal therefore is to minimise the amount of time the camera is obstructed (i.e. minimise the amount of time the UAV cannot see the target). To do this target movement needs to be taken into account. Assuming the algorithm designed to solve the problem takes a certain time to run, an area of probable target locations can be identified. Figure 2.10 shows this, with the smaller circle being the possible area in which the target can be located, the dots representing random samples of this area and the arrows from the larger circle indicating the LOS of the UAV some of which are blocked by buildings (broken arrows).

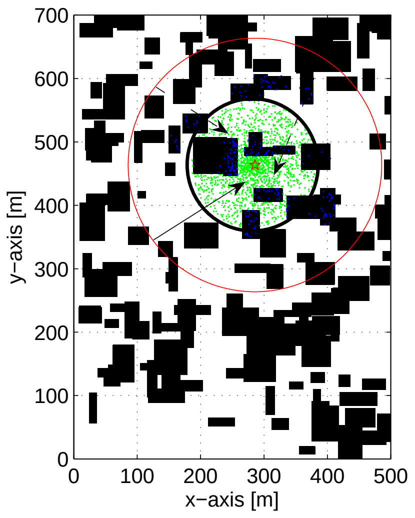

FIGURE 2.10: Possible location of the target with possible UAV flight path to track the target

Using this setup a cost function was defined that minimises the probability of the UAV LOS being blocked as a function of circle centre. An additional parameter was included to allow multiple UAVs to be optimally spaced around this optimally centred circle. Figure 2.11 shows the final result, with the solid circle representing the optimal path.

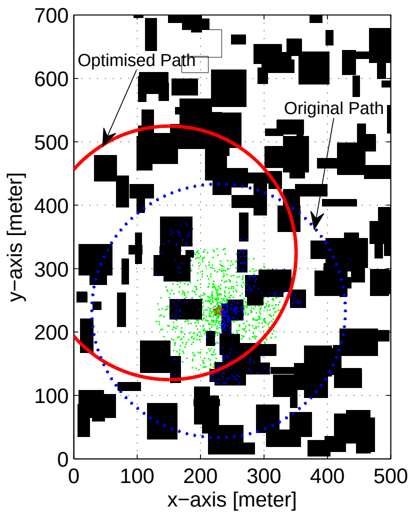

FIGURE 2.11: Flight path with optimal centre, depicted by the solid circle

Using this information as a basis the problem this work tries to solve is how to move between, or onto a circular flight path in an optimal fashion while attempting to stay as close to the final circular path as possible during the transfer. This requirement stems from the nature of the target tracking algorithm, as the closer to the final circular path the UAV is during the transfer the higher the probability of tracking the target even while carrying out the transfer.

From this work a basic formulation for the problem can be made (see 3.2). From the basic formulation it is possible to apply the cost function with constraints to numerous path planning methods. Only some of these are suitable in solving the problem and these will be discussed next.

2.6 Path Planning methods

Depending on the mission requirements paths can be pre-generated before the mission or can be calculated in real time. Paths are pre-generated for missions where path planning algorithms require considerable computational power and run times that would not be available to the UAV, hence making them unsuitable for real time applications [25, 26]. The majority of real time path planning algorithms are tailored to work in short time frames allowing a UAV to quickly adapt to changing surroundings; a good example of this would be collision avoidance where a UAV reacts to the presence of an object and calculates a path to avoid it [27–29]. This reactive path planning can clearly not be achieved in pre-generation as the presence of obstructions would not be known in advance. This is also true when considering weather avoidance as weather conditions vary in real time. However real-time path planners generally operate on a subset of the original problem to provide the increased computation speed resulting in locally optimal or ”good enough for purpose” solutions.

2.6.1 Potential Field

The Potential Field method as shown in [17] works by allocating a mathematically repulsive potential to anything considered an obstacle/obstruction and a mathematically attractive potential to anything that is a way-point/goal point. This results in a mathematical potential function which can be used to plan a path. The theory is that the path will be pushed around obstacles/obstructions and drawn towards way-points/goal points generating a path that allows the vehicle or manipulator to navigate the obstacle field.

The potential field method can be computationally very quick but does require very detailed information about the obstacle field being navigated including object locations and the shape of the obstacles. Certain obstacle shapes can cause problems with this method as they can cause vehicles to become trapped, never generating a path that achieves the mission. In addition to this the paths generated by this method are not guaranteed to be optimal. Other (more recent) examples of the potential field method can be found in [30–34].

The potential field method is also not useful for the type of path planning required to solve this problem due to its lack of constraint handling beyond physical path shape restrictions. No constraints beyond terminal location would exist as the only object in this setup would be the terminal circle which would have an attractive potential. Although a path would be generated no consideration is given for the other constraints. Curvature could be handled but it would require additional repulsive objects to be added at every time point to bound the curvature of the path.

2.6.2 Monte-Carlo Simulations

Monte-Carlo simulations are not strictly speaking a method of path planning and see use in many fields of study. They are iterative algorithm that generate results from randomly selected feasible inputs. They are used to identify trends in many problems where analytical solutions do not exist. By repeatedly running a simulation many times the resultant output can be checked to see if the solution being tested is producing results as expected. If a significant number of runs produce the desired result it proves the algorithm being simulated is operating as desired. The number of runs is important as there need to be enough so that the resultant output is statistically valid.

It is a useful approach to validate many optimisation problems such as path planners as they often do not have an analytical solution. It is therefore necessary to use numerical methods to evaluate the problem instead. Research that makes use of a Monte-Carlo method can be seen in [24, 35].

2.6.3 Branch and Bound

This method works using the principle that there exists a set of possible candidate solutions that can be split into multiple smaller subsets whose union forms the original candidate set. This branching is carried out multiple times to form a tree like search structure. The bounding section of the algorithm is so called as it involves calculating the upper and lower bound of the cost being optimized for each subset created in the branching operation. A process of pruning is then carried out where subsets are removed from the search tree depending on the calculated bound values. This can be done because some subsets have lower bounds that are greater than the upper bounds of others making them un-viable solutions. The pruning process is carried out iteratively until only one possible solution remains or the upper and lower bounds of a solution are equal, providing the optimal cost result.This method is reliant on the efficiency of the branch-and-bound calculations; good choices make for a fast pruning process and therefore a fast convergence onto the optimal path; poor choices however can cause branches to be analysed multiple times slowing the process. Examples of Branch-and-Bound optimisation are [36–40]. The best way of illustrating how this works is to look at a basic example:

Imagine there exist four UAVs and four possible paths to carry the UAVs from a start point to a mission area. Each UAV can fly any path but only one UAV can be assigned to each path. As a UAV follows a path it has a probability of being detected and subsequently shot down. What path/UAV combination should be chosen to allow the UAVs to fly to the mission point with the least chance of being detected? The detection probability for each UAV on each path is shown in Table 2.1.

		Path			
		1	2	3	4
	A	0.1	0.2	0.1	0.15
UAVs	B	0.3	0.4	0.2	0.3
	C	0.15	0.3	0.15	0.2
	D	0.25	0.3	0.25	0.3

Table 2.1: Detection Probability Table for four UAVs and four possible path choices for Branch and Bound Example

In terms of branch and bound terminology the following statements are valid:

- The bounding function in used is to assign unassigned UAV to paths based on their chance of detection assigning the least likely to be detected first even if it is assigned more than once.

- The Incumbent Solution is the best feasible solution currently found within the tree.
- A node in the branch and bound tree is a UAV path allocation either partially or fully.

- Nodes are selected based on the minimum value of the bounding function.

- Paths are selected based on their natural order, 1 to 4.

- The branch and bound process is terminated when the value of the Incumbent Solution is better than or equal to the values of the bounding function at all nodes. The solution is deemed optimal if the Incumbent solution exists when the algorithm terminates.

- A Fathomed node is one that has been expanded until each UAV is assigned to a singular unique path.

Using the above a solution tree is created starting from a Root point. As there have been no decisions made at this root point it is simplest to choose the UAV with the lowest probability to fly each path which in this case is A. The root then branches to the four possible nodes representing the four UAV that can fly path one. At each of these nodes the indicated UAV is allocated to follow path 1 and cannot be chosen again to fly any other path. The remaining UAVs are then assigned to fly the path that they have the lowest detection probability for. This yields a bounding value for each node that is the overall detection probability for the chosen path/UAV combination. For example at the Node labelled A UAV A is selected to fly path 1 the remaining paths are then assigned to UAV C as this has the lowest probability of the remaining choices and yields a bounding value of $(0.1)(0.3)(0.15)(0.2) = 0.0009$. As none of the nodes for path 1 assign a single UAV to a single path it is necessary to expand a node and branch for path 2 . The sensible choice for expansion is the node that has the lowest bounding value as this is most likely to yield an optimal solution given that our goal is to assign paths to UAVs that minimises their chance of detection. To this end the node for C at path 1 is expanded. This time as C is already assigned to path 1 only three nodes are created for UAV A, B and D. This therefore assigns these UAVs to path 2 and the remain UAVs select their paths as before. This gives one result that assigns a UAV to each path in the combination CABD for paths 1234 respectively. This has a bounding value of 0.0018 and is set as our Incumbent solution. Any node that has a value gretaer than this Incumbent solution can be trimmed as it is incapable of producing a better result. Any node that has a value of less than the Incumbent Solution is set as a new Incumbent Solution allowing further node trimming. By continuing to expand the tree along all possible options and update the incumbent solution where necessary non optimal nodes cam be trimmed resulting in a singular solution or a set of solutions that all have the same probability. The full tree can be seen in Figure 2.12 with the final solution to the problem being UAV A on path 1, D on path 2, B on path 3 and C on path 4.

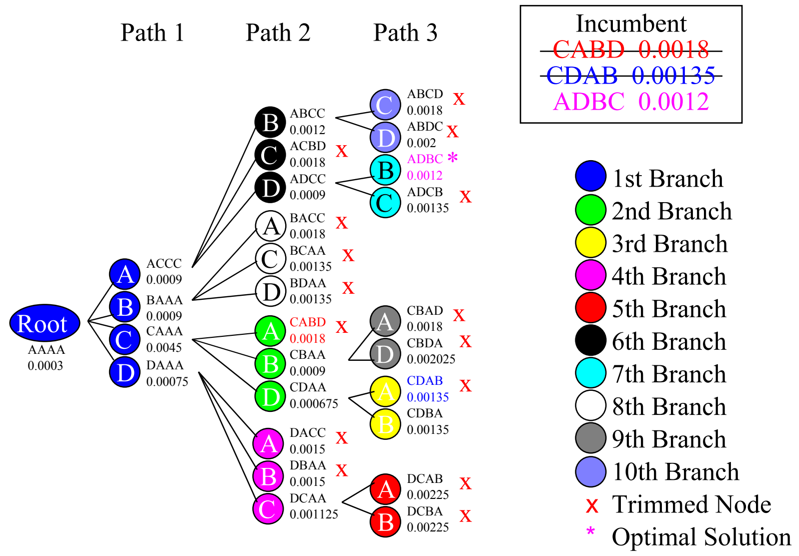

FIGURE 2.12: The solution tree for the branch and bound example of UAVs selecting the best path combination to minimise probability of detection

The above example is a good representation of the branch and bound technique in action. It can be seen however that it operates by selecting from known solutions. This makes this method unsuitable for the transfer problem as it requires knowledge of the solution space that is not available in this case. Due to there being little limitation on possible solutions beyond its constraint functions there is no way to split the problem into a form suitable for the branch and bound technique. Every time point within the solution would have to be represented within the branch and bound algorithm so that the path could be generated. As time divisions decreased to a relatively small value so as to maximise the accuracy of the generated path the number of branch and bound operations would increase exponentially making the method computationally difficult to carry out and ill suited to solve for global solutions.

2.6.4 Genetic Algorithm

This is a different approach to classical optimisation algorithms like branch-and-bound [41–46]. It works by making paths evolve towards their goal while competing for dominance. Weak paths (those considered to be less optimal than others) will 'die' out leaving only stronger and stronger paths with each passing generation. The goal is to be left with only the strongest (most optimal) path, which in effect will have 'beaten' all other paths. This type of approach can however require large time scales and computational requirements to generate an optimal path. Genetic algorithms also suffer from an issue called premature convergence where the algorithm will converge to a locally optimal solution and not the desired globally optimal one [47].

This approach is not suitable for much the same reason as Branch and Bound. An initial population is required on which to operate and as we have no information about possible solutions with which to work there is no easy start point for a genetic algorithm approach.

2.6.5 Two-Point Boundary Value Problem(TPBVP/TPBV)

As the name suggests this algorithm attempts to solve a problem between two boundary points (generally one being the start point and one being the end point). This method relies on being able to represent the system as a set of first order ordinary differential equations (ODEs) that can be evaluated at both of the two boundary points with a number of boundary conditions that must be met. The goal is then to iteratively solve the differential equations so that they conform to the set boundary conditions resulting in a solution for any unknown parameters in the original cost function. Once a solution has been found then the desired path information can be extracted from the original cost function and the ODEs. [48, 49] are examples of TPBVP being applied to path planning.

On a basic level a shooting approach, [50], is a form of boundary value problem better called an initial value problem (IVP) as it only relies on the initial values. The easiest analogy for the shooting method is to think of cannon being fired from a known start point and the goal is to get the projectile from the cannon to travel to a desired end point/plane while meeting certain conditions such as final velocity constraints. The cannon is fired and the path generated is checked against the constraints. The control parameters for the cannon are then adjusted and the cannon can be fired again generating a new path. This process can then be repeated modifying the control parameters for the problem until the path generated meets all constrains within a given tolerance including the minimisation/maximisation of the desired cost function. This method can only be used when all the initial conditions are well known which is usually not the case, as a result it is not often applied in practice with the TPBVP often utilised instead to overcome this shortcoming.

This method is a good candidate for the solving of the problem posed. The problem can easily be represented in the correct form as it trying to generate a path between a known start point and end point giving the two points required for a TPBVP. Due to the constraints that will be required based on UAV limitations and mission parameters it is necessary to form the problem using the Hamiltonian so that these are included in the optimisation. The Hamiltonian and its boundary conditions are derived as follows:

2.6.5.1 Necessary Conditions with Unconstrained Control Inputs

Assume there exists a system described by the following:

$$\dot{x}(t) = a(x(t), u(t), t) \tag{2.1}$$

where a is a function relating state and control inputs to the equivalent state derivatives.

The optimal control problem to be solved is to find control values u^* that cause the system described by Equation 2.1 to follow an optimal trajectory x^* that minimises a desired performance measure. This performance measure can be generally described as:

$$J(u) = h(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt \tag{2.2}$$

where t_0 and t_f are initial and final time respectively and h and g are scalar functions. Assuming h is differentiable,

$$h(x(t_f), t_f) = \int_{t_0}^{t_f} \frac{d}{dt}\left[h(x(t), t)\right] dt + h(x(t_0), t_0) \tag{2.3}$$

using this fact the performance measure can be rewritten

$$J(u) = \int_{t_0}^{t_f} \left\{ g(x(t), u(t), t) + \frac{d}{dt}\left[h(x(t), t)\right] \right\} dt + h(x(t_0), t_0) \tag{2.4}$$

since initial time and initial states are fixed, they are unaffected by the minimisation shortening the performance measure

$$J(u) = \int_{t_0}^{t_f} \left\{ g(x(t), u(t), t) + \frac{d}{dt}\left[h(x(t), t)\right] \right\} dt \tag{2.5}$$

Equation 2.5 can be further expanded by applying the chain rule to the derivative term involving h

$$J(u) = \int_{t_0}^{t_f} \left\{ g(x(t), u(t), t) + \left[\frac{\partial h}{\partial x}(x(t), t)\right]^T + \frac{\partial h}{\partial t}(x(t), t) \right\} dt \tag{2.6}$$

To include the constraints imposed by Equation 2.1 the performance measure given by Equation 2.6 can be augmented by introducing Lagrange multipliers $\lambda_1(t), \dots, \lambda_n(t)$.

$$\begin{aligned} J_a(u) = \int_{t_0}^{t_f} \Bigg\{ g(x(t), u(t), t) + \left[\frac{\partial h}{\partial x}(x(t), t)\right]^T + \frac{\partial h}{\partial t}(x(t), t) \\ + \lambda^T(t)\left[a(x(t), u(t), t) - \dot{x}(t)\right] \Bigg\} dt \end{aligned} \tag{2.7}$$

by further defining

$$\begin{aligned} g_a(x(t), \dot{x}(t), u(t), \lambda(t), t) \triangleq\ & g(x(t), u(t), t) \\ & + \lambda^T(t)\left[a(x(t), u(t), t) - \dot{x}(t)\right] \\ & + \left[\frac{\partial h}{\partial x}(x(t), t)\right]^T + \frac{\partial h}{\partial t}(x(t), t) \end{aligned} \tag{2.8}$$

Using the definition given by Equation 2.8 the augmented function becomes:

$$J_a(u) = \int_{t_0}^{t_f} \left\{ g_a(x(t), \dot{x}(t), u(t), \lambda(t), t) \right\} dt \tag{2.9}$$

For a minimum point to exist the variation of J_a , δJ_a must equal zero on an extremal. By introducing the variations $\delta x, \delta \dot{x}, \delta u, \delta \lambda$ and δt_f the variation δJ_a can be defined:

$$
\begin{aligned}
\delta J_a(u^*) = 0 = &\left[\frac{\partial g_a}{\partial \dot{x}}(x^*(t_f), \dot{x}^*(t_f), u^*(t_f), \lambda^*(t_f), t_f)\right]^T \delta x_f \\
&+ \Bigg[g_a(x^*(t_f), \dot{x}^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) \\
&- \left[\frac{\partial g_a}{\partial x}(x^*(t_f), \dot{x}^*(t_f), u^*(t_f), \lambda^*(t_f), t_f)\right]^T \delta \dot{x}(t_f)\Bigg] \delta t_f \\
&+ \int_{t_0}^{t_f} \Bigg\{ \Bigg[\left[\frac{\partial g_a}{\partial x}(x^*(t), \dot{x}^*(t), u^*(t), \lambda^*(t), t)\right]^T \\
&- \frac{d}{dt}\left[\frac{\partial g_a}{\partial \dot{x}}(x^*(t), \dot{x}^*(t), u^*(t), \lambda^*(t), t)\right]^T \Bigg] \delta x(t) \\
&+ \left[\frac{\partial g_a}{\partial u}(x^*(t), \dot{x}^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta u(t) \\
&+ \left[\frac{\partial g_a}{\partial \lambda}(x^*(t), \dot{x}^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta \lambda(t) \Bigg\} dt
\end{aligned}
\tag{2.10}
$$

Where x_f is the final states. Extracting the terms involving function h from Equation 2.10 yields

$$
\frac{\partial}{\partial x}\left[\left[\frac{\partial h}{\partial x}(x^*(t), t)\right]^T \dot{x}^*(t) + \frac{\partial h}{\partial t}(x^*(t), t)\right] - \frac{d}{dt}\left\{\frac{\partial}{\partial \dot{x}}\left[\left[\frac{\partial h}{\partial x}(x^*(t), t)\right]^T \dot{x}^*(t)\right]\right\} \tag{2.11}
$$

Writing Equation 2.11 out in full and applying the chain rule:

$$
\begin{aligned}
&\left[\frac{\partial^2 h}{\partial x^2}(x^*(t), t)\right] \dot{x}^*(t) + \left[\frac{\partial^2 h}{\partial t \partial x}(x^*(t), t)\right] \dot{x}^*(t) \\
&\qquad - \left[\frac{\partial^2 h}{\partial x^2}(x^*(t), t)\right] \dot{x}^*(t) - \left[\frac{\partial^2 h}{\partial x \partial t}(x^*(t), t)\right] \dot{x}^*(t)
\end{aligned}
\tag{2.12}
$$

assuming the second partial derivatives are continuous and the terms sum to zero leaves the integral terms as:

$$\begin{aligned}\int_{t_0}^{t_f} \Bigg\{ \Bigg[\left[\frac{\partial g}{\partial x}(x^*(t), u^*(t), t)\right]^T &+ \lambda^{*T}(t)\left[\frac{\partial a}{\partial x}(x^*(t), u^*(t), t)\right] \\ &- \frac{d}{dt}\Big[-\lambda^{*T}(t)\Big]\Bigg]\delta x(t) + \Bigg[\left[\frac{\partial g}{\partial u}(x^*(t), u^*(t), t)\right]^T \\ &+ \lambda^{*T}(t)\left[\frac{\partial a}{\partial u}(x^*(t), u^*(t), t)\right]\Bigg]\delta u(t) \\ &+ \Bigg[\Big[a(x^*(t), u^*(t), t) - v\dot{x}^*(t)\Big]^T\Bigg]\delta\lambda(t)\Bigg\}dt\end{aligned} \tag{2.13}$$

On an extremal Equation 2.13 must vanish. The constraints must also be satisfied on an extremal meaning that the coefficient of $\delta\lambda(t)$ is zero. Since the Lagrange multipliers are arbitrary they can be selected to make the coefficient of $\delta x(t)$ zero:

$$\dot{\lambda}^*(t) = \left[\frac{\partial a}{\partial x}(x^*(t), u^*(t), t)\right]^T \lambda^*(t) - \frac{\partial g}{\partial x}(x^*(t), u^*(t), t) \tag{2.14}$$

Equation 2.14 are called the costate equations with $\lambda(t)$ being the costate.

The final variation $\delta u(t)$ is independent making it coefficient also zero:

$$0 = \frac{\partial g}{\partial u}(x^*(t), u^*(t), t) + \left[\frac{\partial a}{\partial u}(x^*(t), u^*(t), t)\right]^T \lambda^*(t) \tag{2.15}$$

The terms outside the integral must still be delt with. given that the variation must be zero these terms must also add to zero:

$$\begin{aligned}\left[\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f)\right]^T \delta x_f + \Bigg[g(x^*(t_f), u^*(t_f), t_f) &+ \frac{\partial h}{\partial t}(x^*(t_f), t_f) \\ &+ \lambda^{*T}(t_f)\Big[a(x^*(t_f), u^*(t_f), t_f)\Big]\Bigg]\delta t_f = 0\end{aligned} \tag{2.16}$$

these are all the necessary conditions but by defining the Hamiltonian as follows:

$$\mathscr{H}(x(t), u(t), \lambda(t), t) \triangleq g(x(t), u(t), t) + \lambda^T(t)\left[a(x(t), u(t), t)\right] \tag{2.17}$$

the necessary conditions can be written as:

$$\dot{x}^*(t) = \frac{\partial \mathscr{H}}{\partial \lambda}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.18}$$

$$\dot{\lambda}^*(t) = -\frac{\partial \mathscr{H}}{\partial x}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.19}$$

$$0 = \frac{\partial \mathscr{H}}{\partial u}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.20}$$

For all $t \in [t_0, t_f]$

$$\begin{aligned}\left[\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f)\right]^T \delta x_f + \Bigg[\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) \\ + \frac{\partial h}{\partial t}(x^*(t_f), t_f)\Bigg]\delta t_f = 0\end{aligned} \tag{2.21}$$

2.6.5.2 *Necessary Conditions with Constrained Control Inputs (Pontryagin's Minimum Principle)*

In the previous section the derivation only takes into account when control values are unbounded which realistically is rarely the case. The control u^* cause the performance measure J to have a relative minimum if:

$$J(u) - J(u^*) = \Delta J \geq 0 \tag{2.22}$$

If we express $u = u^* + \delta u$ the increment in J becomes

$$\Delta J(u^*, \delta u) = \delta J(u^*, \delta u) + \text{higher-order terms}; \tag{2.23}$$

By considering admissible variations with $\|\delta u\|$ small enough so that the sign of ΔJ is determined by δJ, a necessary condition for u^* to minimise J is

$$\delta J(u^*, \delta u) \geq 0 \tag{2.24}$$

if u^* lies on a boundary during the time interval $[t_0, t_f]$ and

$$\delta J(u^*, \delta u) = 0 \tag{2.25}$$

if u^* lies within the boundary during the time interval $[t_0, t_f]$

By applying this change to the previous derivation of the necessary conditions. The increment of J is

$$\begin{aligned}\Delta J(u^*, \delta u) =& \left[\frac{\partial h}{\partial x}(x^*(t_f), t_f)) - \lambda^*(t_f)\right]^T \delta x_f \\ &+ \left[\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f)\right]\delta t_f \\ &+ \int_{t_0}^{t_f} \left\{ \left[\dot{\lambda}^*(t) + \frac{\partial \mathscr{H}}{\partial x}(x^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta x(t) \right. \\ &+ \left[\frac{\partial \mathscr{H}}{\partial u}(x^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta u(t) \\ &+ \left. \left[\frac{\partial \mathscr{H}}{\partial \lambda}(x^*(t), u^*(t), \lambda^*(t), t) - \dot{x}^*(t)\right]^T \delta\lambda(t) \right\} dt \\ &+ \text{higher-order terms}\end{aligned} \tag{2.26}$$

If $\lambda^*(t)$ is chosen so that the coefficient of $\delta x(t)$ in the integral is zero, the state equations are satisfied and the boundary condition equation given by Equation 2.21 is satisfied. The increment becomes

$$\begin{aligned}\Delta J(u^*, \delta u) = &\int_{t_0}^{t_f} \left[\frac{\partial \mathscr{H}}{\partial u}(x^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta u(t) dt \\ &+ \text{higher-order terms}\end{aligned} \tag{2.27}$$

It can be seen that the integrand is the first-order approximation of the change in $\mathscr{H}$ cause by a change in u alone, or:

$$\begin{aligned}&\left[\frac{\partial \mathscr{H}}{\partial u}(x^*(t), u^*(t), \lambda^*(t), t)\right]^T \delta u(t) \doteq \\ &\mathscr{H}(x^*(t), u^*(t) + \delta u(t), \lambda^*(t), t) - \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t)\end{aligned} \tag{2.28}$$

therefore,

$$\begin{aligned}\Delta J(u^*, \delta u) = \int_{t_0}^{t_f} \Big[& \mathscr{H}(x^*(t), u^*(t) + \delta u(t), \lambda^*(t), t) \\ & - \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \Big] dt \\ & + \text{higher-order terms}\end{aligned} \tag{2.29}$$

If $u^* + \delta u$ is in a sufficiently small neighbourhood of u^* ($\|\delta u\| < \tau$ where τ defines the maximum magnitude of this neighbourhood) the higher-order terms are small and the the integral term dominates. Therefore for u^* to be a minimizing control it is necessary that

$$\Delta J(u^*, \delta u) = \int_{t_0}^{t_f} \Big[\mathscr{H}(x^*(t), u^*(t) + \delta u(t), \lambda^*(t), t) - \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \Big] dt \geq 0 \tag{2.30}$$

for all admissible δu, such that $\|\delta u\| < \tau$

Therefore, for Equation 2.30 to be satisfied for admissible δu

$$\mathscr{H}(x^*(t), u^*(t) + \delta u(t), \lambda^*(t), t) \geq \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.31}$$

for all admissible δu and all $t \in [t_0, t_f]$

Consider the following control

$$\begin{aligned} u(t) &= u^*(t); \qquad t \notin [t_1, t_2] \\ u(t) &= u^*(t) + \delta u(t); \qquad t \in [t_1, t_2] \end{aligned} \tag{2.32}$$

where $[t_1, t_2]$ is a non zero arbitrarily small time interval and $\delta u(t)$ is an admissible control variation that satisfies $\|\delta u\| < \tau$.

Assume that the control in Equation 2.32 does not satisfy the inequality in Equation 2.31. Therefore in the time interval $[t_1, t_2]$

$$\mathscr{H}(x^*(t), u(t), \lambda^*(t), t) < \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.33}$$

Therefore,

$$\int_{t_0}^{t_f} \left[\mathscr{H}(x^*(t), u(t), \lambda^*(t), t) - \mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \right] dt < 0 \tag{2.34}$$

Since the time interval $[t_1, t_2]$ can be anywhere in the interval $[t_0, t_f]$, Equation 2.33 also exists for any $t \in [t_0, t_f]$. This means that it it is always possible to obtain and admissible control such as that given in Equation 2.32, which causes $\Delta J < 0$. Since this finding contradicts the optimality of the control u^* the necessary condition for u^* to minimise J is:

$$\mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \leq \mathscr{H}(x^*(t), u(t), \lambda^*(t), t) \tag{2.35}$$

for all $t \in [t_0, t_f]$ and all admissible controls.

Equation 2.35 indicates that an optimal control must minimise the Hamiltonian, this is Pontryagin's Minimum Principle.

the necessary conditions found previously can then be rewritten as:

$$\dot{x}^*(t) = \frac{\partial \mathscr{H}}{\partial \lambda}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.36}$$

$$\dot{\lambda}^*(t) = -\frac{\partial \mathscr{H}}{\partial x}(x^*(t), u^*(t), \lambda^*(t), t) \tag{2.37}$$

$$\mathscr{H}(x^*(t), u^*(t), \lambda^*(t), t) \leq \mathscr{H}(x^*(t), u(t), \lambda^*(t), t) \tag{2.38}$$

For all $t \in [t_0, t_f]$ and all admissible $u(t)$

$$\left[\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f)\right]^T \delta x_f + \left[\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f)\right] \delta t_f = 0 \tag{2.39}$$

2.6.5.3 Boundary Condition Derivation

Using Equation 2.21 it is possible to derive different boundary conditions by making appropriate substitutions based on the final conditions of time and states.

2.6.5.3.1 Fixed Final Time Problems Three cases exist when final time t_f is known, $x(t_f)$ can be free, fixed or be required to lie on a surface:

- Case I: Final State Fixed
  As both $x(t_f)$ and t_f are known, δx_f and δt_f both equal zero. Applying this to Equation 2.21 yields n boundary equations:

$$x^*(t_f) = X_f \tag{2.40}$$

- Case II: Final State Free
  This time only $\delta t_f = 0$ and δx_f is arbitrary. This makes Equation 2.21 become

$$\frac{\partial h}{\partial x}(x^*(t_f)) - \lambda^*(t_f) = 0 \tag{2.41}$$

  This hold as the function h is independent of t_f as t_f is fixed.

- Case III: Final state lying in a surface defined by $m(x(t)) = 0$
  Consider an example situation where the final state of a system must lie on a circle defined as follows:

$$m(x(t)) = [x_1(t) - 3]^2 + [x_2(t) - 4]^2 - 4 = 0 \tag{2.42}$$

  The system here is second-order. Admissible changes in $\delta x(t_f)$ are tangent to the circle at the point $(x^*(t_f), t_f)$. The tangent line is normal to the gradient vector

$$\frac{\partial m}{\partial x}(x^*(t_f)) = \begin{bmatrix} 2[x_1^*(t_f) - 3] \\ 2[x_2^*(t_f) - 4] \end{bmatrix} \tag{2.43}$$

  at the point $(x^*(t_f), t_f)$

  Therefore, $\delta x(t_f)$ must be normal to the gradient Equation 2.43 so that

$$\left[\frac{\partial m}{\partial x}(x^*(t_f))\right]^T \delta x(t_f) = 2[x_1^*(t_f) - 3]\delta x_1(t_f) + 2[x_2^*(t_f) - 4]\delta x_2(t_f) = 0 \tag{2.44}$$

  We can then solve for $\delta x_2(t_f)$

$$\delta x_2(t_f) = -\frac{[x_1^*(t_f) - 3]}{[x_2^*(t_f) - 4]}\delta x_1(t_f) \tag{2.45}$$

  which can then be substituted into Equation 2.21, giving

$$\left[\frac{\partial h}{\partial x}(x^*(t_f)) - \lambda^*(t_f)\right]^T \begin{bmatrix} 1 \\ -\frac{[x_1^*(t_f)-3]}{[x_2^*(t_f)-4]}\delta x_1(t_f) \end{bmatrix} = 0 \tag{2.46}$$

  since $\delta t_f = 0$ and $\delta x_1(t_f)$ is arbitrary.

The second required equation at final time is that the final optimal state must be on the surface,

$$m(x^*(t_f)) = [x_1^*(t_f) - 3]^2 + [x_2^*(t_f) - 4]^2 - 4 = 0 \tag{2.47}$$

This example can now be transferred to the general situation where there are n state variables and $1 \leq k \leq n-1$ relationships that the states must satisfy at final time. The $m(x(t))$ relationship is then

$$m(x(t)) = \begin{bmatrix} m(x_1(t)) \\ \vdots \\ m_k(x(t)) \end{bmatrix} = 0 \tag{2.48}$$

where each component of m represents a hypersurface in the n-dimensional state space.

The final state therefore lies on the intersection of these hypersurfaces with δx_f a tangent to each hypersurface at the point $(x^*(t_f), t_f)$. This means that $\delta x(t_f)$ is normal to each of the gradient vectors

$$\frac{\partial m_1}{\partial x}(x^*(t_f)), \ldots, \frac{\partial m_k}{\partial x}(x^*(t_f)) \tag{2.49}$$

These are assumed to be linearly independent.

Since $\delta t_f = 0$, Equation 2.21 becomes

$$\left[\frac{\partial h}{\partial x}(x^*(t_f)) - \lambda^*(t_f)\right]^T \delta x(t_f) \triangleq v^T \delta x(t_f) = 0 \tag{2.50}$$

where Equation 2.50 can only be satisfied if the vector v is a linear combination of the gradient vectors from Equation 2.49

$$\frac{\partial h}{\partial x}(x^*(t_f)) - \lambda^*(t_f) = d_1\left[\frac{\partial m_1}{\partial x}(x^*(t_f))\right] + \ldots + d_k\left[\frac{\partial m_k}{\partial x}(x^*(t_f))\right] \tag{2.51}$$

again the second required equation at final time is that the final optimal states must be on the hypersurface

$$m(x^*(t_f)) = 0 \tag{2.52}$$

Applying Equation 2.51 and 2.52 to the original example can be shown to yield the same results as originally obtained.

2.6.5.3.2 Free Final Time Problems Similarly to fixed final time problems several case exist.

- Case I: Final State Fixed
  Since final time is free δt_f is arbitrary and $\delta x_f = 0$ Equation 2.21 becomes

$$\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f) = 0 \tag{2.53}$$

- Case II: Final State Free
  In this case both δt_f and δx_f are arbitrary and independent. Therefore for Equation 2.21 to equal zero the coefficients of δt_f and δx_f must be zero

$$\lambda^*(t_f) = \frac{\partial h}{\partial x}(x^*(t_f), t_f) \tag{2.54}$$

$$\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f) = 0 \tag{2.55}$$

  Note that if $h = 0$

$$\lambda^*(t_f) = 0 \tag{2.56}$$

$$\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) = 0 \tag{2.57}$$

- Case III: $x(t_f)$ lies on a moving point $\theta(t)$
  In this case δx_f and δt_f are related as follows:

$$\delta x_f \doteq \left[\frac{d\theta}{dt}(t_f)\right] \delta t_f \tag{2.58}$$

  substituting this into Equation 2.21 gives

$$\begin{gathered}\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f) + \left[\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f)\right]^T \\ \times \left[\frac{d\theta}{dt}(t_f)\right] = 0\end{gathered} \tag{2.59}$$

  and the remaining equation that the optimal final states must lie on the point

$$x^*(t_f) = \theta(t_f) \tag{2.60}$$

- Case IV: Final state lying on a surface defined by $m(x(t)) = 0$. This case is similar to Case III when final time was fixed. Since δx_f is independent of δt_f, the coefficient of δt_f must be zero. This gives four conditions:

$$x^*(t_0) = x_0 \tag{2.61}$$

$$\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f) = d_1\left[\frac{\partial m_1}{\partial x}(x^*(t_f))\right] + \ldots + d_k\left[\frac{\partial m_k}{\partial x}(x^*(t_f))\right] \tag{2.62}$$

$$m(x^*(t_f)) = 0 \tag{2.63}$$

$$\mathscr{H}(x^*(t_f), u^*(t_f), \lambda^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f) = 0 \tag{2.64}$$

(Refer to Case III for fixed time for more detailed derivation)

- Case V: Final state lying on a moving surface defined by $m(x(t), t) = 0$
This case is very similar to Case IV except that the surface is itself moving.

$$\frac{\partial h}{\partial x}(x^*(t_f), t_f) - \lambda^*(t_f) = d_1\left[\frac{\partial m_1}{\partial x}(x^*(t_f), t_f)\right] + \ldots + d_k\left[\frac{\partial m_k}{\partial x}(x^*(t_f), t_f)\right] \tag{2.65}$$

$$m(x^*(t_f), t_f) = 0 \tag{2.66}$$

2.6.6 Receding Horizon

A receding horizon approach is a form of online path planning that attempts to approximate a globally optimal solution using a cut down and simplified version of the mathematics used to solve the global problem. The receding horizon approach breaks the problem down into a horizon of known end time with detailed information about the solution space within the horizon. The goal is to find the point within the horizon that minimises or maximises the desired cost function, with the path needed to move optimally through the horizon being the path between the horizon start point and the found optimal end point. This process can then be repeated with the previous end point becoming the start point for the next horizon. A path can therefore be constructed from a collection of these reduced horizons. [51–53] show examples of receding horizon approaches applied to UAV tracking type algorithms.

The algorithm applicable to the transfer problem in this case is based on the work shown in [54].

The main simplification that directly impacts the run time of the algorithm is the reduction of the solution space. One of the complexities of the global algorithm is the curvature range allowing paths to take any curvature between the upper and lower limits. This means that numerous paths exist within the solution space causing the iterative process to run for longer time scales as the algorithm is trying numerous curvatures while searching for a solution. The receding horizon approach reduces this large set of solutions to a small set of known curvature values. In this case three different path shapes are considered:

1. Straight - A straight path with zero curvature

2. Left Curve - A turn to the left with maximum positive curvature

3. Right Curve - A turn to the right with maximum negative curvature

Therefore at each time point, or node, only three possible solutions exist. Since computational power is no longer required to identify the solution it is simply the case of calculating the cost value for every possible node within the horizon using the desired cost function and then selecting the optimal point. Within the horizon the possible paths are sampled from the feasible curvature area as shown on Figure 2.13.

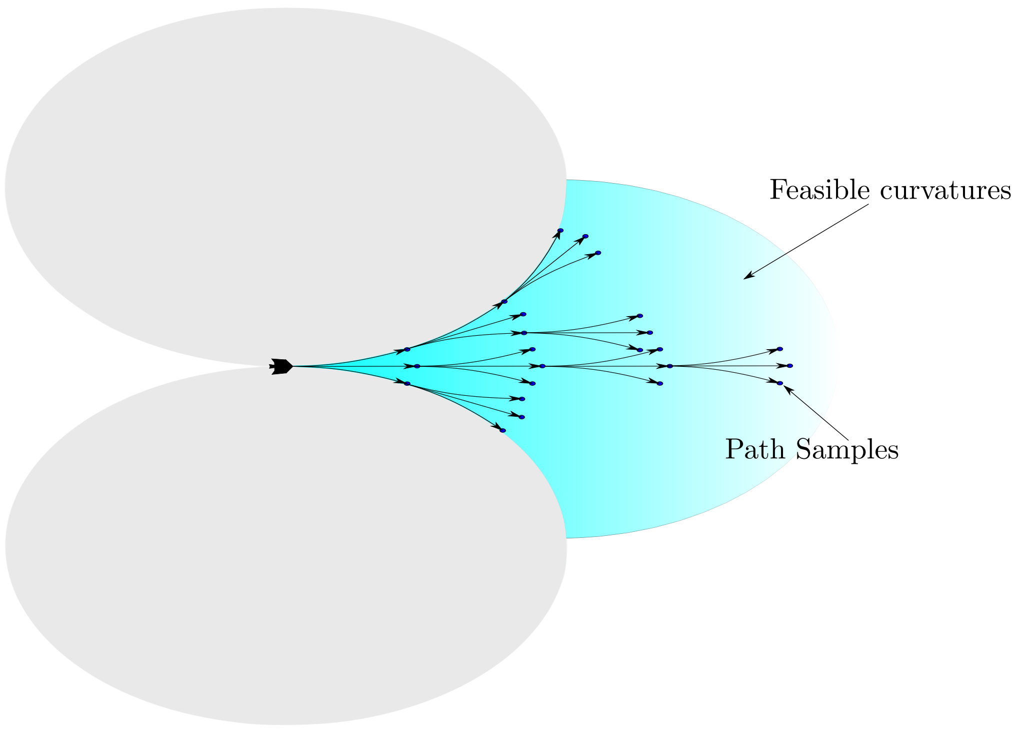

FIGURE 2.13: Receding horizon path sampling from feasible curvatures

The number of points sampled in each horizon is given by the following equation:

$$n_s^{total} = \sum_{k=1}^{N} n_s^k \tag{2.67}$$

where n_s is the number of paths available at each time interval (3 in this case) and N comes from the equation:

$$T = N\Delta t \tag{2.68}$$

where T is the horizon length and Δt is the sampling interval. For a horizon where $T = 10$, $\Delta t = 2$ and $n_s = 3$ the number of nodes that need to be sampled for each horizon is 363, however if $\Delta t = 5$ then the number of nodes is reduced to only 39. Clearly as T and Δt are adjusted the number of nodes can be altered to achieve the best trade off between number of calculations (and therefore algorithm run time) and algorithm accuracy (the more samples taken the more accurate the path representation). Once the appropriate sample space is decided a cost can be allocated to each node. The node with the lowest cost gives the end point of the optimal path within the horizon and is in turn the start point of the next horizon. To obtain the full path from horizon start to this optimal end point it is simply a case of working back along the preceding nodes. Figure 2.14 shows an example of this for the case where $n_s = N = 3$.

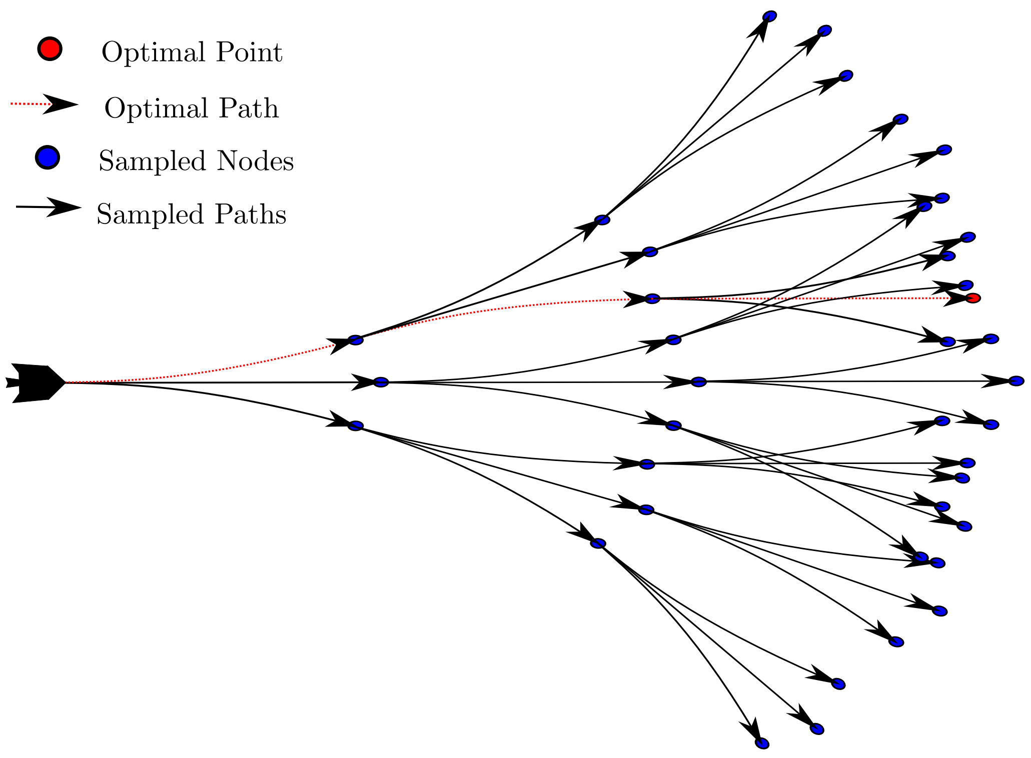

Figure 2.14: Receding horizon path selection from sampled space

The process is then repeated over multiple horizons until the cost value of the optimal point is within the desired threshold (i.e. the horizon has arrived at the destination within a set tolerance). An example of this can be seen in Figure 2.15.

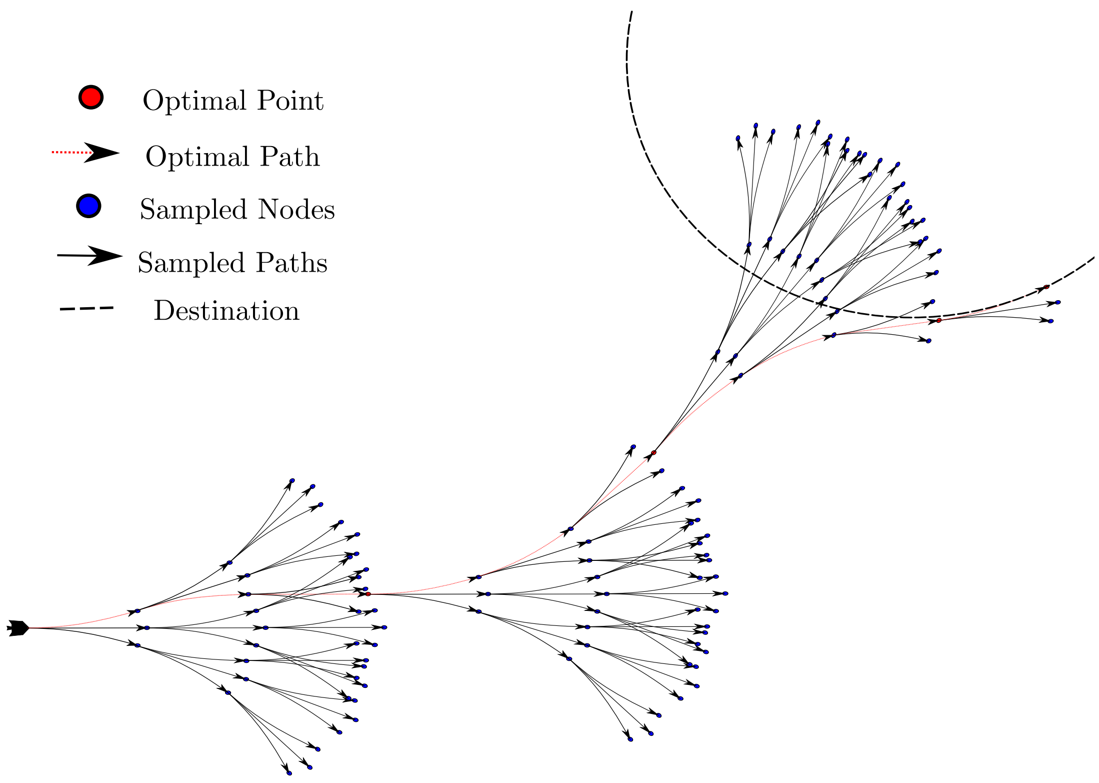

Figure 2.15: Receding horizon example

Although not applicable to solve the main problem of this work due to it not producing a global solution the receding horizon approach gives a reasonable approximation of results within a fast time scale, making this method an appropriate choice when comparing the results of different algorithms used to solve this problem.

2.7 Importance of the work in this thesis

The majority of the path planning methods mentioned above carefully choose their solution spaces so that they become easier to solve.

Starting with Dubin's where only two possible path choices exist this has been expanded as computational methods have improved that are better equipped to handle larger solution spaces. Simple additions have been made to expand on the capability of such path planners such as adding the ability to reverse to Dubin's results or adding a third dimensions to handle altitude.

Even more complex algorithm such as Genetic algorithms, branch and bound and Receding Horizon work using reduced or carefully selected solution spaces to facilitate their application to problems and faster runtimes. Indeed branch and bound algorithms can not work unless provided with known solutions on which to operate as too large or an unknown solution space would result in the algorithm having a long runtime or failing completely.

This work relaxes the constraint on the available solution space seen in similar methods by allowing paths to take any value as long as they conform to certain constraints such as velocity magnitude and maximum curvature of turn. This is a closer approximation to how a human operator would control a UAV and facilitates the goal of producing globally optimal solutions for this problem rather than the locally solutions of other methods arising from their use of a tightly constrained solution space.

The relaxing of the curvature constraint is far removed from works like Dubin's or even Receding Horizon where curvature is fixed to a set value or limited to only a few cases. This will allow the available choice of paths to greatly increase.

The work in this thesis looks at the development of an algorithm that utilises a more realistic curvature constraint and investigates its performance in producing global solutions for the transfer problem in question.

Chapter 3

Algorithm Construction and Testing

3.1 Introduction

Before any path planning algorithm can be implemented the problem has to first be defined. This includes UAV dynamics plus any constraints that apply. In addition this also includes deciding on the desired form of the cost function to meet the requirements of transfer paths. From this initial formulation it is possible to construct an algorithm using a Two Point Boundary Value(TPBV) method to solve the problem. The initial formulation is detailed and then followed by the formulation of the TPBV problem including the identification of the necessary boundary conditions . The resulting issues of this TPBV problem are discussed giving motivation for the implementation of a simpler shooting approach. The formulation of the shooting algorithm is discussed including a discussion on cost function validity. Algorithm results are presented showing the functionality of the developed algorithm.

3.2 Initial Problem Formulation

3.2.1 UAV Equations

As with most Engineering problems it is possible to apply some simplifications to the transfer problem. Firstly the altitude can be assumed constant therefore reducing the problem to two dimensions. Secondly as the goal is to produce a guidance law that gives path information, rather than a control law that allows a specific UAV to follow the path information, it is acceptable to use simplified UAV dynamics when solving the problem. The UAV can be considered as a point mass and its dynamics in state-space are given by.

$$\dot{\mathbf{x}} = \begin{bmatrix} 0_2 & I_2 \\ 0_2 & 0_2 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0_2 \\ I_2 \end{bmatrix} \mathbf{u} = A\mathbf{x} + B\mathbf{u} \tag{3.1}$$

where $\mathbf{x} = [x,\ y,\ v_x,\ v_y]^T$, x and y are the coordinates of aircraft, v_x and v_y are the aircraft velocity in x and y direction, respectively, $[\cdot]^T$ is the transpose, 0_2 is 2×2 zero matrix, I_2 is 2×2 identity matrix, $\mathbf{u} = [u_x,\ u_y]^T$, u_x and u_y are the control input in the x and y direction, respectively.

From the results shown in [24] the nominal path for an aircraft tracking a target in a dense obstacle area is circular. The problem can therefore be formed as a transfer from one circular path to another as shown in Figure 3.1. The initial and final UAV position must be as follows:

$$x^2(t_0) + y^2(t_0) = r^2 \tag{3.2a}$$

$$x(t_0)\dot{x}(t_0) + y(t_0)\dot{y}(t_0) = 0 \tag{3.2b}$$

$$[x(t_f) - \alpha]^2 + [y^2(t_f) - \beta] = r^2 \tag{3.2c}$$

$$[x(t_f) - \alpha]\dot{x}(t_f) + [y(t_f) - \beta]\dot{y}(t_f) = 0 \tag{3.2d}$$

where α is the distance offset between the centres of the two circles in x, β is the distance offset between the centres of the two circles in y, initial time t_0 is fixed, initial location $x(t_0)$ and $y(t_0)$ are given, final time t_f is free, and final location $x(t_f)$ and $y(t_f)$ lie on the circle defined by Equation 3.2c. All quantities are expressed in the global coordinates shown in Figure 3.2.

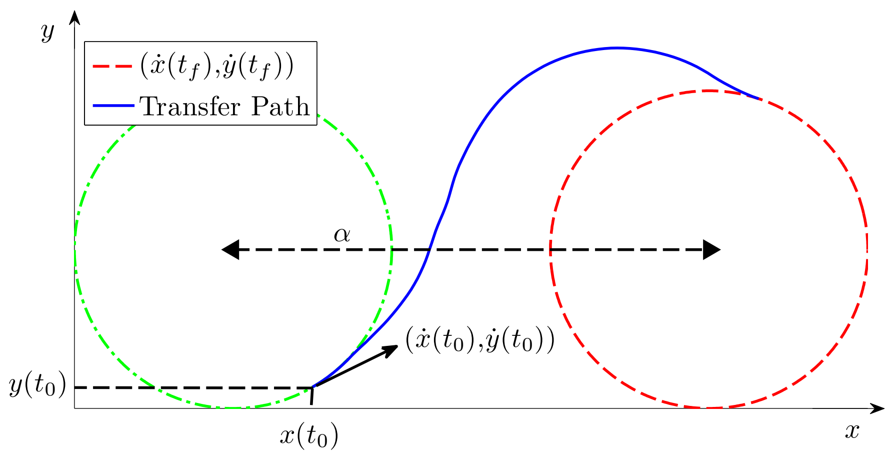

FIGURE 3.1: An example of the transfer between two circular tracking paths

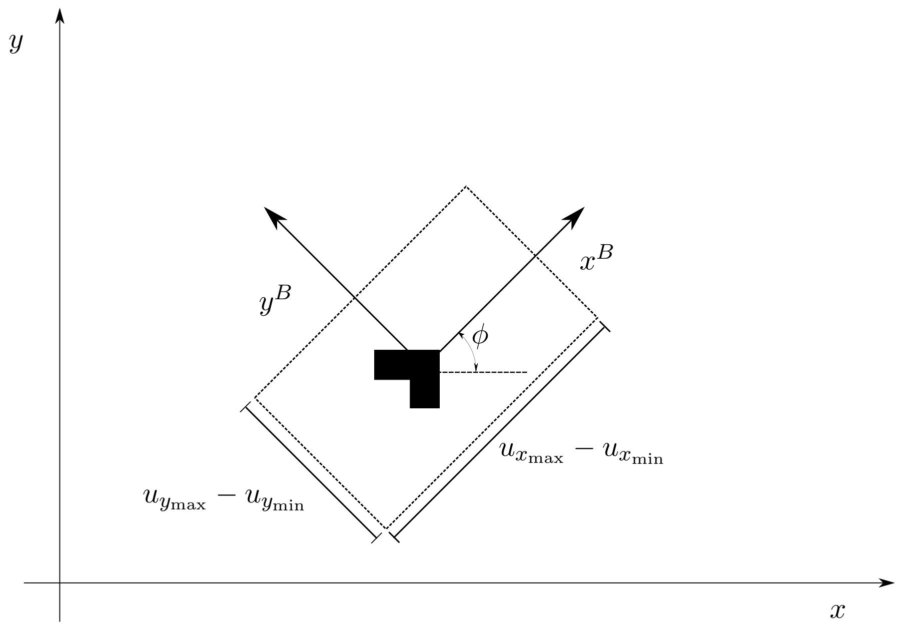

FIGURE 3.2: Global (X–Y) and UAV local ($x_B - y_B$) coordinates with the control input magnitude constraint, which is indicated by the dotted box.

3.2.2 UAV Constraints

The following critical constraints for the UAV must be considered to design an optimal guidance algorithm:

- Velocity for x-direction in the body coordinates must satisfy the following:

$$0 < v_{\text{min}} \leq v_x^B \leq v_{\text{max}} \tag{3.3}$$

where v_x^B is the aircraft velocity in the body coordinates. The aircraft attitude is assumed to be aligned with the velocity vector, hence, v_y^B is always equal to zero. Equation 3.3 can be expressed in the global coordinates as follows:

$$v_{\text{min}}^2 \leq v_x^2 + v_y^2 \leq v_{\text{max}}^2 \tag{3.4}$$

where

$$v_x = v_x^B \cos\phi \tag{3.5a}$$

$$v_y = v_x^B \sin\phi \tag{3.5b}$$

where ϕ is equal to $\tan^{-1}(v_y/v_x)$.

- Control input magnitudes are constrained as follows:

$$u_{x_{\text{min}}} \leq u_x^B \leq u_{x_{\text{max}}} \tag{3.6a}$$

$$u_{y_{\text{min}}} \leq u_y^B \leq u_{y_{\text{max}}} \tag{3.6b}$$

where u_x^B and u_y^B are the control input expressed in the aircraft body coordinates. Note that the ranges of control input for x and y are not the same as each other, in general. Equation 3.6 can be expressed in the global coordinates as follows:

$$u_{x_{\text{min}}} \leq u_x \cos\phi + u_y \sin\phi \leq u_{x_{\text{max}}} \tag{3.7a}$$

$$u_{y_{\text{min}}} \leq -u_x \sin\phi + u_y \cos\phi \leq u_{x_{\text{max}}} \tag{3.7b}$$

- The turn radius of aircraft must be larger than the given minimum radius turn of aircraft. In other words, the radius of curvature of the flight path must be smaller than the inverse of the minimum radius.

$$\frac{|v_x u_y - v_y u_x|}{\left(v_x^2 + v_y^2\right)^{3/2}} \leq \frac{1}{r_{\text{min}}} \tag{3.8}$$

where r_{min} is the radius of the circle corresponding to the minimum radius turn. The above equation can be written as follows:

$$-\frac{1}{r_{\text{min}}}\left(v_x^2 + v_y^2\right)^{3/2} \leq v_x u_y - v_y u_x \leq \frac{1}{r_{\text{min}}}\left(v_x^2 + v_y^2\right)^{3/2} \tag{3.9}$$

where the region in the control input space satisfying the above inequalities is the enclosed region by two straight lines which are parallel to the x_B-axis.

3.2.3 Choosing the basic form of the cost function

To successfully carry out the path planning task the form of the cost function must first be decided. Two sensible measures exist that can be applied to this problem:

1. Minimum Time
2. Minimum Distance

To choose which of these measure is most appropriate it is first required to look at the type of path desired for the transfer. In this case the most direct path is desirable as it is the most useful when considering the target tracking mission. If the most direct path is used then the UAV would always be travelling towards the tracking region defined by the circle. By using minimum time no control is given over the directionality of the path which could result in paths that are shortest in time but may not be the most direct. Minimum distance on the other hand is directly trying to control the length of the generated path, with the minimum distance equating to the most direct path. A cost function involving minimum distance is therefore the most appropriate for this application. The path length parameter will form the basis of the minimum distance cost function and can be derived as follows:

3.2.3.1 Path Length Derivation

For UAV path planning problems, one of the natural choices for the minimising cost function is the path length. The equation for path length can be derived from the equation of arc length as follows:

For a straight line the arc length can be simply be represented as the square root of the square sum of the difference in x and y coordinates

$$L = \sqrt{(\Delta x)^2 + (\Delta y)^2} \tag{3.10}$$

An arbitrary path can be split into numerous small straight line segments (polygonal arc) each having a length of:

$$L_i = \sqrt{[f(t_i) - f(t_{i-1})]^2 + [g(t_i) - g(t_{i-1})]^2} \tag{3.11}$$

where $x = f(x)$, $y = g(x)$ and $t_0 < t < t_f$ is the time interval over which the path exists.

However if

$$\Delta t = t_i - t_{i-1} = (t_0 - t_f)/n \tag{3.12}$$

then from the mean value theorem

$$f(t_i) - f(t_{i-1}) = f'(t_i^*)\Delta t \tag{3.13a}$$

$$g(t_i) - g(t_{i-1}) = g'(t_i^*)\Delta t \tag{3.13b}$$

where t_i^* is some time in $[t_{i-1}, t_i]$

Substituting this back

$$L_i = \sqrt{[f'(t_i^*)]^2 + [g'(t_i^*)]^2}\Delta t \tag{3.14}$$

The total path length is therefore the sum of all the L_i elements

$$s = \sum_{i=1}^{n} L_i \tag{3.15a}$$

$$s = \sum_{i=1}^{n} \sqrt{[f'(t_i^*)]^2 + [g'(t_i^*)]^2}\Delta t \tag{3.15b}$$

as $n \rightarrow \infty$

$$s = \int_{t_0}^{t_f} \sqrt{[f'(t_i^*)]^2 + [g'(t_i^*)]^2} dt \tag{3.16a}$$

$$s = \int_{t_0}^{t_f} \sqrt{\dot{x}^2 + \dot{y}^2} dt \tag{3.16b}$$

as, integrating over an infinitesimally small arc length:

$$\delta s = \sqrt{\delta x^2 + \delta y^2} = \delta t \sqrt{\dot{x}^2 + \dot{y}^2} \tag{3.17}$$

3.2.4 Form of the Optimal Control Problem

From equations Equation 3.4, Equation 3.7 and Equation 3.9 the form of optimal control problem can be defined as:

$$\underset{t_f \in [0,\infty), \mathbf{u}(t) \in \mathbb{U}}{\text{Minimise}} \; J = \int_{t_0}^{t_f} f(t) dt \tag{3.18}$$

where $\mathbb{U}$ is a compact set defined by Equation 3.7, t_0 is the initial time, t_f is the final time, which is free, and $f(t)$ is a function to be chosen based on Equation 3.16b.

subject to

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$

and

$$\begin{gathered} v_{\min}^2 \leq v_x^2 + v_y^2 \leq v_{\max}^2 \\ u_{x_{\min}} \leq u_x \cos\phi + u_y \sin\phi \leq u_{x_{\max}} \\ u_{y_{\min}} \leq -u_x \sin\phi + u_y \cos\phi \leq u_{x_{\max}} \\ -\frac{1}{r_{\min}}\left(v_x^2 + v_y^2\right)^{3/2} \leq v_x u_y - v_y u_x \leq \frac{1}{r_{\min}}\left(v_x^2 + v_y^2\right)^{3/2} \end{gathered}$$

3.3 Two Point Boundary Value(TPBV) Problem

From the initial formulation it is required to obtain a set of ordinary differential equations (ODE) that will be used to solve the TPBV problem. For problems with constraints such as this an augmented cost function called the Hamiltonian is produced that gives a set of ODE and associated boundary conditions that should be sufficient to solve the problem.

3.3.1 Hamiltonian Generation

Firstly an augmented cost function must be formed to create the Hamiltonian $\mathscr{H}$.

The cost function chosen for this is the square of path length from Equation 3.16b:

$$J = \int_{t_0}^{t_f} \dot{x}^2 + \dot{y}^2 dt \tag{3.19}$$

This choice is due to the removal of the square root as it provides a simpler equation to augment. The Hamiltonian is formed by augmenting the original cost function by adding the constraint equations in Equation 3.18, weighted using time varying Lagrange multipliers:

$$\begin{aligned} \mathscr{H} =& x_3^2 + x_4^2 + \lambda_1 x_3 + \lambda_2 x_4 + \lambda_3 u_x + \lambda_4 u_y \\ &+ \lambda_5[(x_3^2 + x_4^2 - V_{\max}^2)^2 \mathbb{1}(V_{\max}^2 - x_3^2 - x_4^2) \\ &+ (V_{\min}^2 - x_3^2 - x_4^2)^2 \mathbb{1}(x_3^2 + x_4^2 - V_{\min}^2)] \end{aligned} \tag{3.20}$$

where $x_3 = \dot{x}_1 = \dot{x}$, $x_4 = \dot{x}_2 = \dot{y}$, $u_x = \dot{x}_3 = \ddot{x}$ and $u_y = \dot{x}_4 = \ddot{y}$. $\lambda_{1,2,...5}$ are time varying Lagrange multipliers. The terms involving $\mathbb{1}$ are defined as follows:

$$\mathbb{1}(-f_i) = \begin{cases} 1 \text{ for } f_i \leq 0 \\ 0 \text{ for } f_i > 0 \end{cases} \tag{3.21}$$

Taking the derivative of $\mathscr{H}$ and setting it to zero gives one of the necessary conditions for a minimum point to exist as follows:

$$\frac{\delta\mathscr{H}}{\delta x_i} = -\dot{\lambda}_i = \mathscr{H}_{x_i}$$

where $i = 1, 2, 3.....,n$.
Therefore

$$\mathscr{H}_{x_1} = -\dot{\lambda}_1 = 0 \tag{3.22}$$

$$\mathscr{H}_{x_2} = -\dot{\lambda}_2 = 0 \tag{3.23}$$

$$\begin{aligned}\mathscr{H}_{x_3} &= -\dot{\lambda}_3 \\ &= -2x_3 - \lambda_1 - \lambda_5[4x_3(x_3^2 + x_4^2 - V_{\text{max}}^2)\mathbb{1}(V_{\text{max}}^2 - x_3^2 - x_4^2) \\ &- 4x_3(V_{\text{min}}^2 - x_3^2 - x_4^2)\mathbb{1}(x_3^2 + x_4^2 - V_{\text{min}}^2)]\end{aligned} \tag{3.24}$$

$$\begin{aligned}\mathscr{H}_{x_4} &= -\dot{\lambda}_4 \\ &= -2x_4 - \lambda_2 - \lambda_5[4x_4(x_3^2 + x_4^2 - V_{\text{max}}^2)\mathbb{1}(V_{\text{max}}^2 - x_3^2 - x_4^2) \\ &- 4x_4(V_{\text{min}}^2 - x_3^2 - x_4^2)\mathbb{1}(x_3^2 + x_4^2 - V_{\text{min}}^2)]\end{aligned} \tag{3.25}$$

$$\mathscr{H}_{x_5} = \dot{\lambda}_5 = 0 \tag{3.26}$$

In addition to these equations the derivatives of the states can also be defined as:

$$\frac{\delta\mathscr{H}}{\delta\lambda_i} = \dot{x}_i$$

where $i = 1, 2, 3.....,n$.

Therefore

$$\begin{aligned}
\dot{x_1} =& x_3 \\
\dot{x_2} =& x_4 \\
\dot{x_3} =& u_x \\
\dot{x_4} =& u_y \\
\dot{x_5} =& (x_3^2 + x_4^2 - V_{\max}^2)^2 \mathbb{1}(V_{\max}^2 - x_3^2 - x_4^2) \\
& + (V_{\min}^2 - x_3^2 - x_4^2)^2 \mathbb{1}(x_3^2 + x_4^2 - V_{\min}^2)
\end{aligned}$$

where x_5 is a dummy state to allow for the addition of the fifth Lagrange multiplier λ_5.

This gives ten differential equations that describe the problem. It is shown that λ_1, λ_2 and λ_5 are constant values.

3.3.2 Hamiltonian Control Considerations

Due to the existence of a constraint on the control inputs the problem must be solved using Pontryagin's Minimum Principle. This involves finding a control input that minimises the Hamiltonian $\mathscr{H}$. Firstly the Hamiltonian is rewritten by extracting the terms involving u_x and u_y and then forming the minimisation problem using the control constraint equation (3.7)

$$\underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}}\, \mathscr{H}_{control} = \lambda_3 u_x + \lambda_4 u_y \tag{3.27}$$

subject to

$$\begin{gathered}
u_{x_{\min}} \leq u_x \cos\phi + u_y \sin\phi \leq u_{x_{\max}} \\
u_{y_{\min}} \leq -u_x \sin\phi + u_y \cos\phi \leq u_{x_{\max}} \\
-\frac{1}{r_{\min}} \left(v_x^2 + v_y^2\right)^{3/2} \leq v_x u_y - v_y u_x \leq \frac{1}{r_{\min}} \left(v_x^2 + v_y^2\right)^{3/2}
\end{gathered}$$

Due to the condition imposed by Equation 3.7 u_x and u_y form a box in the u_x and u_y plane as shown in Figure 3.3.

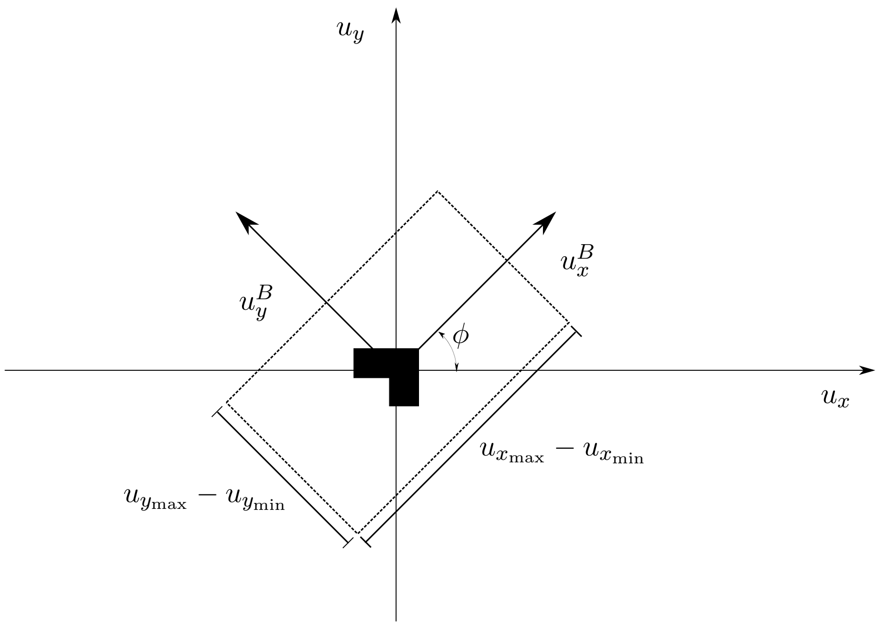

FIGURE 3.3: Graphical Representation of the UAV Control Box

By performing a 2D rotation on the u_x^B and u_y^B body coordinates the u_x and u_y values for any ϕ can be found. The transformation is as follows:

$$\begin{bmatrix} u_x \\ u_y \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} u_x^B \\ u_y^B \end{bmatrix} \tag{3.28}$$

The available control values exist within this box and will dictate the value of $\mathscr{H}_{control}$. This is because the minimum value of $\mathscr{H}_{control}$ is a function of the maximum magnitude of u_x and u_y. This can more clearly be seen by noting that, from Equation 3.27, the cost function describes lines in the u_x and u_y plane given by Equation 3.29 as shown in Figure 3.4.

$$u_y = -\frac{\lambda_3}{\lambda_4} u_x + \frac{\mathscr{H}_{control}}{\lambda_4} \tag{3.29}$$

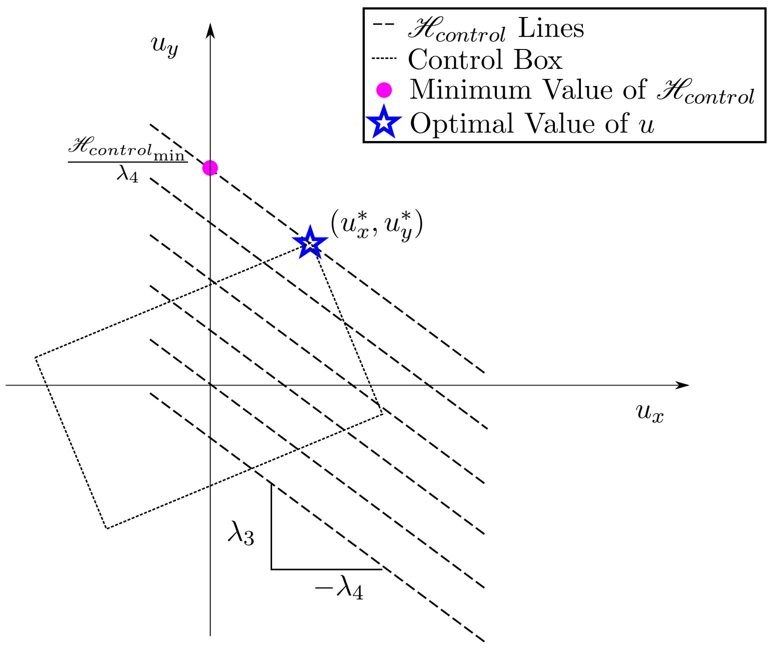
$\mathscr{H}_{control}$ Lines
Control Box
Minimum Value of $\mathscr{H}_{control}$
Optimal Value of u
u_y
u_x
$\frac{\mathscr{H}_{control_{\min}}}{\lambda_4}$
(u_x^*, u_y^*)
λ_3
$-\lambda_4$

FIGURE 3.4: Graphical Representation of the Control Cost

The optimal control values of u_x^* and u_y^* will therefore exist when the line corresponding to the minimum $\mathscr{H}_{control}$ value is within the bounding box. However the curvature constraint can also affect the size of this box and must first be taken into account before the optimal control values can be selected.

The curvature constraint is two lines in the u_x and u_y plane that will either be large enough that they encompass the whole u_x^B and u_y^B box or are small enough that they shrink the u_x^B and u_y^B box as shown in Figure 3.5.

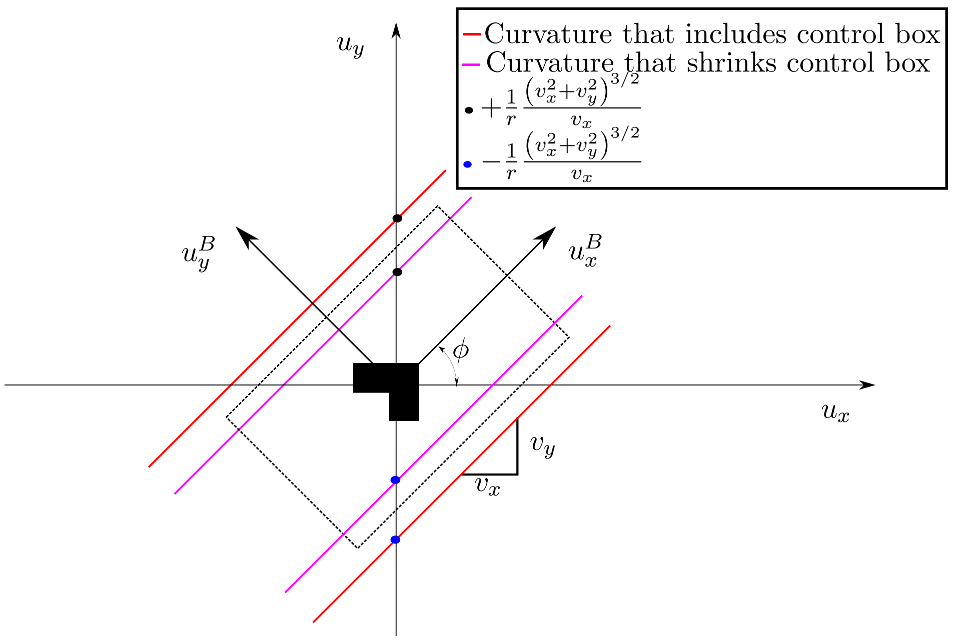
Curvature that includes control box
Curvature that shrinks control box
$+\frac{1}{r}\frac{(v_x^2+v_y^2)^{3/2}}{v_x}$
$-\frac{1}{r}\frac{(v_x^2+v_y^2)^{3/2}}{v_x}$
u_y
u_y^B
u_x^B
ϕ
u_x
v_y
v_x

FIGURE 3.5: Graphical Representation of the Curvature Constraint

The control box will therefore change in size depending on the velocity of the aircraft and how this affects the curvature constraint. If the control box shrinks the maximum available control will be less and therefore impact the value of $\mathscr{H}_{control}$. The curvature lines are always parallel with the u_x^B axis and can effectively reduce the control magnitude available in u_y^B. This is the case as the angle of the curvature lines to the u_x axis is the same as the heading angle, ϕ, of the UAV. The Figure 3.5 shows the generic case for this curvature bounding however when the body axes are parallel to the inertial axes two special cases exist for the equations of the bounding lines:

- Case $\sharp$ 1, $v_x = 0$ and $v_y \neq 0$

$$u_x = \pm\frac{1}{r}v_y^2$$
$$u_y = 0$$

- Case $\sharp$ 2, $v_x \neq 0$ and $v_y = 0$

$$u_x = 0$$
$$u_y = \pm\frac{1}{r}v_x^2$$

Using the above the u_x^B and u_y^B control box can be defined for any velocity allowing for the corresponding u_x and u_y values to be found that minimise equation Equation 3.27.

Knowing that the minimum $\mathscr{H}_{control}$ occurs at the boundary of the control box, where control magnitude is at its maximum, we only need consider the u_x and u_y values at the four corners of the box. The minimum value of $\mathscr{H}_{control}$ will be achieved at one of these points allowing u_x^* and u_y^* to be selected. The coordinates of the four corners as shown in Figure 3.6, in the u_x and u_y plane are:

$$\begin{aligned}
u_{x_{\text{right upper}}} &= u_{x_{\max}} \cos\phi - u_{y_{\max}} \sin\phi \\
u_{y_{\text{right upper}}} &= u_{x_{\max}} \sin\phi + u_{y_{\max}} \cos\phi \\
u_{x_{\text{right lower}}} &= u_{x_{\max}} \cos\phi - u_{y_{\min}} \sin\phi \\
u_{y_{\text{right lower}}} &= u_{x_{\max}} \sin\phi + u_{y_{\min}} \cos\phi \\
u_{x_{\text{left upper}}} &= u_{x_{\min}} \cos\phi - u_{y_{\max}} \sin\phi \\
u_{y_{\text{left upper}}} &= u_{x_{\min}} \sin\phi + u_{y_{\max}} \cos\phi \\
u_{x_{\text{left lower}}} &= u_{x_{\min}} \cos\phi + u_{y_{\min}} \sin\phi \\
u_{y_{\text{left lower}}} &= u_{x_{\min}} \sin\phi + u_{y_{\min}} \cos\phi
\end{aligned}$$

where the value of $u_{y_{\max}}$ and $u_{y_{\min}}$ varies depending on the curvature constraints.

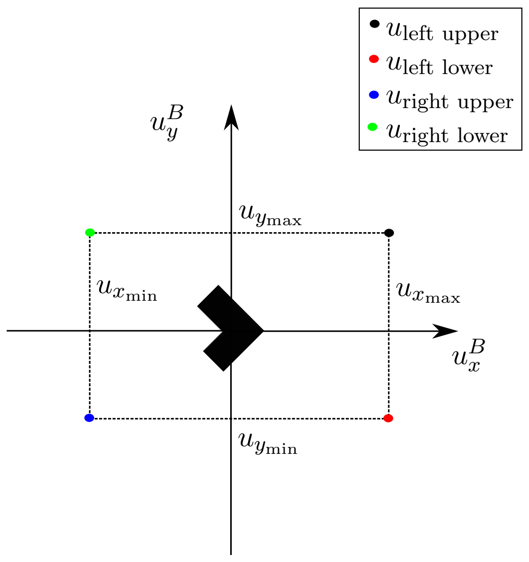
● $u_{\text{left upper}}$
● $u_{\text{left lower}}$
● $u_{\text{right upper}}$
● $u_{\text{right lower}}$
u_y^B
$u_{y_{\max}}$
$u_{x_{\min}}$
$u_{x_{\max}}$
u_x^B
$u_{y_{\min}}$

FIGURE 3.6: Reference for the corners of the control box including maximum and minimum control indications

3.3.3 TPBV Problem Boundary Conditions

To complete the formulation of the TPBV problem the remaining boundary conditions must be derived. To do this the following sufficient condition must be met:

$$[\frac{\delta h}{\delta x}(x^*(t_f),t_f) - \lambda^*(t_f)]^T \delta x_{t_f} + [\mathscr{H}(t_f) + \frac{\delta h}{\delta t}(x^*(t_f),t_f)]\delta t_f = 0 \tag{3.30}$$

where x^* and λ^* are optimal x and λ, the function h is a function of the final time and final x and $\mathscr{H}(t_f)$ is the Hamiltonian at final time.

For Equation 3.30 to equal zero, both parts of the sum must equal zero. Since final time is free, $\delta(t_f)$ is greater than zero,therefore the coefficient of $\delta(t_f)$ must equal zero:

$$\mathscr{H}(t_f) + \frac{\delta h}{\delta t}(x^*(t_f),t_f) = 0 \tag{3.31}$$

In addition since final position lies on a fixed surface:

$$[\frac{\delta h}{\delta x}(x^*(t_f),t_f) - \lambda^*(t_f)]^T \delta x_{t_f} = 0 \tag{3.32}$$

Since h is on a fixed surface it does not vary with final time, hence $\frac{\delta h}{\delta t}(x^*(t_f),t_f) = 0$, giving one boundary condition:

$$\mathscr{H}(t_f) = 0 \tag{3.33}$$

$\frac{\delta h}{\delta x}(x^*(t_f),t_f) = 0$ is the gradient of the surfaces that x must be on at final time:

$$\frac{\delta h}{\delta x}(x^*(t_f),t_f) = \lambda^*(t_f) \tag{3.34}$$

There are three surfaces that x must lie on, one defining final position, one defining final velocity and the third defining the surface for the dummy state x_5 at final time:

$$m_1[x(t)] = [x_1(t_f) - \alpha]^2 + [x_2(t_f) - \beta]^2 - r^2 = 0 \tag{3.35a}$$

$$m_2[x(t)] = 2[x_1(t_f) - \alpha]x_3(t_f) + 2[x_2(t_f) - \beta]x_4(t_f) = 0 \tag{3.35b}$$

$$m_3[x(t)] = x_5 = 0 \tag{3.35c}$$

$\frac{\delta h}{\delta x}(x^*(t_f), t_f)$ will be a weighted sum of the gradients from both of these equations.

$$\frac{\delta m_1}{\delta x} = \begin{bmatrix} 2[x_1(t_f) - \alpha] \\ 2[x_2(t_f) - \beta] \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.36}$$

$$\frac{\delta m_2}{\delta x} = \begin{bmatrix} x_3(t_f) \\ x_4(t_f) \\ 2(x_1(t_f) - \alpha) \\ 2(x_2(t_f) - \beta) \\ 0 \end{bmatrix} \tag{3.37}$$

$$\frac{\delta m_3}{\delta x} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.38}$$

it follows that

$$\frac{\delta h}{\delta x}[x^*(t_f), t_f] = d_1 \begin{bmatrix} 2[x_1(t_f) - \alpha] \\ 2[x_2(t_f) - \beta] \\ 0 \\ 0 \\ 0 \end{bmatrix} + d_2 \begin{bmatrix} x_3(t_f) \\ x_4(t_f) \\ 2(x_1(t_f) - \alpha) \\ 2(x_2(t_f) - \beta) \\ 0 \end{bmatrix} + d_3 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.39}$$

where d_1, d_2 and d_3 are weighting factors, substituting back

$$\begin{bmatrix} \lambda_1(t_f) \\ \lambda_2(t_f) \\ \lambda_3(t_f) \\ \lambda_4(t_f) \\ \lambda_5(t_f) \end{bmatrix} = d_1 \begin{bmatrix} 2(x_1(t_f) - \alpha) \\ 2(x_2(t_f) - \beta) \\ 0 \\ 0 \\ 0 \end{bmatrix} + d_2 \begin{bmatrix} x_3(t_f) \\ x_4(t_f) \\ 2(x_1(t_f) - \alpha) \\ 2(x_2(t_f) - \beta) \\ 0 \end{bmatrix} + d_3 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.40}$$

Using these equations it is possible to obtain values for d_1 and d_2. Firstly note that four equations exist involving d_1 and d_2:

$$\lambda_1(t_f) = 2d_1(x_1(t_f) - \alpha) + d_2 x_3(t_f) \tag{3.41}$$

$$\lambda_2(t_f) = 2d_1(x_2(t_f) - \beta) + d_2 x_3(t_f) \tag{3.42}$$

$$\lambda_3(t_f) = 2d_2(x_1(t_f) - \alpha) \tag{3.43}$$

$$\lambda_4(t_f) = 2d_2(x_2(t_f) - \beta) \tag{3.44}$$

Using the last of these two equations it is possible to solve for d_2:

$$d_2 = \frac{\lambda_3(t_f)}{2(x_1(t_f) - \alpha)} = \frac{\lambda_4(t_f)}{2(x_2(t_f) - \beta)} \tag{3.45}$$

Substituting back into the equations for λ_1 and λ_2:

$$\lambda_1(t_f) = 2d_1(x_1(t_f) - \alpha) + \frac{\lambda_3(t_f)x_3(t_f)}{2(x_1(t_f) - \alpha)} \tag{3.46}$$

$$\lambda_2(t_f) = 2d_1(x_2(t_f) - \beta) + \frac{\lambda_4(t_f)x_4(t_f)}{2(x_2(t_f) - \beta)} \tag{3.47}$$

These equations can then be solved for d_1:

$$d_1 = \frac{\lambda_1(t_f)}{2(x_1(t_f) - \alpha)} - \frac{\lambda_3(t_f)x_3(t_f)}{4(x_1(t_f) - \alpha)^2} = \frac{\lambda_2(t_f)}{2(x_2(t_f) - \beta)} - \frac{\lambda_4(t_f)x_4(t_f)}{4(x_2(t_f) - \beta)^2} \tag{3.48}$$

If Equation 3.48 and Equation 3.45 are rearranged to equal zero a further two boundary conditions are obtained:

$$\frac{\lambda_3(t_f)}{2(x_1(t_f) - \alpha)} = \frac{\lambda_4(t_f)}{2(x_2(t_f) - \beta)} \tag{3.49}$$

$$2\lambda_3(t_f)(x_2(t_f) - \beta) = 2\lambda_4(t_f)(x_1(t_f) - \alpha) \tag{3.50}$$

$$\lambda_3(t_f)(x_2(t_f) - \beta) - \lambda_4(t_f)(x_1(t_f) - \alpha) = 0 \tag{3.51}$$

and

$$\frac{\lambda_1(t_f)}{2(x_1(t_f)-\alpha)} - \frac{\lambda_3(t_f)x_3(t_f)}{4(x_1(t_f)-\alpha)^2} = \frac{\lambda_2(t_f)}{2(x_2(t_f)-\beta)} - \frac{\lambda_4(t_f)x_4(t_f)}{4(x_2(t_f)-\beta)^2} \tag{3.52}$$

$$\begin{aligned}\frac{1}{2(x_1(t_f)-\alpha)}&\left[\lambda_1(t_f) - \frac{\lambda_3(t_f)x_3(t_f)}{2(x_1(t_f)-\alpha)}\right]\\ &= \frac{1}{2(x_2(t_f)-\beta)}\left[\lambda_2(t_f) - \frac{\lambda_4(t_f)x_4(t_f)}{2(x_2(t_f)-\beta)}\right]\end{aligned} \tag{3.53}$$

$$\begin{aligned}2\lambda_1(t_f)(x_2(t_f)-\beta) &- \frac{\lambda_3(t_f)x_3(t_f)(x_2(t_f)-\beta)}{(x_1(t_f)-\alpha)}\\ &= 2\lambda_2(t_f)(x_1(t_f)-\beta) - \frac{\lambda_4(t_f)x_4(t_f)(x_1(t_f)-\alpha)}{(x_2(t_f)-\beta)}\end{aligned} \tag{3.54}$$

$$\begin{aligned}&\frac{1}{(x_1(t_f)-\alpha)}\left[2\lambda_1(t_f)(x_1(t_f)-\alpha)(x_2(t_f)-\beta) - \lambda_3(t_f)x_3(t_f)(x_2(t_f)-\beta)\right]\\ &= \frac{1}{(x_2(t_f)-\beta)}\left[2\lambda_2(t_f)(x_1(t_f)-\alpha)(x_2(t_f)-\beta) - \lambda_4(t_f)x_4(t_f)(x_1(t_f)-\alpha)\right]\end{aligned} \tag{3.55}$$

$$\begin{aligned}(x_2(t_f)-\beta)^2&\left[2\lambda_1(t_f)(x_1(t_f)-\alpha) - \lambda_3(t_f)x_3(t_f)\right]\\ &= (x_1(t_f)-\alpha)^2\left[2\lambda_2(t_f)(x_2(t_f)-\beta) - \lambda_4(t_f)x_4(t_f)\right]\end{aligned} \tag{3.56}$$

$$\begin{aligned}(x_2(t_f)-\beta)^2&\left[2\lambda_1(t_f)(x_1(t_f)-\alpha) - \lambda_3(t_f)x_3(t_f)\right]\\ &- (x_1(t_f)-\alpha)^2\left[2\lambda_2(t_f)(x_2(t_f)-\beta) - \lambda_4(t_f)x_4(t_f)\right] = 0\end{aligned} \tag{3.57}$$

Finally the equation for$\lambda_5(t_f)$ can is simply:

$$\lambda_5(t_f) - d_3 = 0; \tag{3.58}$$

However as the weighting factor d_3 is undetermined and can take any valueEquation 3.3.3 is not a suitable boundary condition for $\lambda_5(t_f)$. To solve this it can be noted that $\dot{\lambda_5}$ was found to be zero in Equation 3.26 meaning that λ_5 is constant. At final time λ_5 must equal the same value as at initial time therefore:

$$\lambda_5(t_f) - \lambda_5(t_0) = 0; \tag{3.59}$$

This gives three boundary conditions involving $\lambda_{1,2,\ldots,5}$ at final time.

These equations in addition to the equations of the non-dummy surfaces, the initial condition and the final condition on the Hamiltonian give a total of 10 boundary conditions:

Four for the initial values:

$$\begin{aligned}
x_1(t_0) - r\cos(\theta) &= 0 \\
x_2(t_0) - r\sin(\theta) &= 0 \\
x_3(t_0) + V_{\min}\sin(\theta) &= 0 \\
x_4(t_0) - V_{\min}\cos(\theta) &= 0
\end{aligned}$$

where θ denotes the angle around the initial circular path from which the UAV departs,

Five for the terminal values:

$$\begin{aligned}
(x_1(t_f) - \alpha)^2 + (x_2(t_f) - \beta)^2 - r^2 &= 0 \\
2(x_1(t_f) - \alpha)x_3(t_f) + 2(x_2(t_f) - \beta)x_4(t_f) &= 0 \\
(x_2(t_f) - \beta)^2 \left[2\lambda_1(t_f)(x_1(t_f) - \alpha) - \lambda_3(t_f)x_3(t_f)\right] & \\
-(x_1(t_f) - \alpha)^2 \left[2\lambda_2(t_f)(x_2(t_f) - \beta) - \lambda_4(t_f)x_4(t_f)\right] &= 0 \\
\lambda_3(t_f)(x_2(t_f) - \beta) - \lambda_4(t_f)(x_1(t_f) - \alpha) &= 0 \\
\lambda_5(t_f) - \lambda_5(t_0) &= 0
\end{aligned}$$

And one for the Hamiltonian at the terminal point

$$\begin{aligned}
x_3^2(t_f) + x_4^2(t_f) + \lambda_1(t_f)x_3(t_f) + \lambda_2(t_f)x_4(t_f) + \lambda_3(t_f)u_x(t_f) + \lambda_4(t_f)u_y(t_f) & \\
+ \lambda_5(t_f)[(x_3^2(t_f) + x_4^2(t_f) - V_{\max}^2)^2 \mathbb{1}(V_{\max}^2 - x_3^2(t_f) - x_4^2(t_f)) & \\
+ (V_{\min}^2 - x_3^2(t_f) - x_4^2(t_f))^2 \mathbb{1}(x_3^2(t_f) + x_4^2(t_f) - V_{\min}^2)] &= 0
\end{aligned}$$

3.3.4 Problems Coding the TPBV problem

MATLABs bvp4c TPBV solver was used to code this problem. After numerous attempts at testing and algorithm rewrites, this method proved to yield no results. The algorithm resulted in the generation of a singular Jacobian while attempting to solve the collocation equations. This was traced to a problem with the initial guess of the solution. Given the limited information as to how the states vary over the solution space the initial guess provided to the solver was arbitrary. The initial guess is used to give the solver a start point and can effect the resulting solution so much so that a different solution can be found simply by varying the initial guess. An inappropriate guess can go as far as causing the solving algorithm to fail completely yielding no result as in this case. As it is infeasible to obtain a more accurate guess for an unknown solution space another method must be employed to solve this path planning task.

As all transfers start form a known point in the solution space the initial values of the problem are well defined. As this is the case it is possible to apply an initial value approach such as a shooting method to solve this problem. In addition it is also possible to further simplify the approach by changing from continuous to discrete time removing the need to handle differential equations when solving the problem and instead being able to work in purely algebraic terms. As a discrete shooting method relies only on the initial conditions the unknown nature of the solution space that restricts the use of a TPBV method will have little impact making it a better choice to solve the posed transfer problem.

3.4 Shooting Method Algorithm

MATLAB solver fmincon was selected as the optimiser to be used for the shooting method problem. From Equation 3.16b two cost functions were considered as viable options to generate distance optimal paths:

$$J = \int_{t_0}^{t_f} \sqrt{\dot{x}^2 + \dot{y}^2} dt \tag{3.60a}$$

$$J = \int_{t_0}^{t_f} \dot{x}^2 + \dot{y} dt \tag{3.60b}$$

3.4.1 Preparing the cost function for algorithm use

As mentioned previously, for numerical optimisation to be applied it is necessary to discretise the cost function so that it can be solved as an algebraic equation rather than a differential equation.

The equations of motion can be discretised as follows:

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k) \tag{3.61}$$

Where A_d and B_d are discrete forms of the A and B matrices respectively and $\mathbf{x}(k+1)$ denotes the next discrete step and $\mathbf{x}(k)$ denotes the previous discrete step.

In turn the cost functions can also be written in a discrete form:

$$J = \sum_{k=0}^{N-1} \sqrt{\dot{x}^2(k) + \dot{y}^2(k)} h \tag{3.62}$$

$$J = \sum_{k=0}^{N-1} \dot{x}^2(k) + \dot{y}^2(k)h \tag{3.63}$$

where h is the discretisation step size.

To create these functions from the original equations of motion we must first select only the x and y velocity components required in the cost functions. This will give us a matrix used for the selection ($\mathbf{C}$) and the output of the selection ($\mathbf{Y}$):

$$\mathbf{C} = \begin{bmatrix} 0_2 & I_2 \end{bmatrix} \tag{3.64}$$

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \tag{3.65}$$

Therefore

$$\mathbf{Y} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \mathbf{CX} \tag{3.66}$$

The costs can then be defined in terms of the matrix $\mathbf{Y}$:

$$J = \sum_{k=0}^{N-1} \sqrt{\mathbf{Y}^T(k)\mathbf{Y}(k)}h \tag{3.67}$$

$$J = \sum_{k=0}^{N-1} \mathbf{Y}^T(k)\mathbf{Y}(k)h \tag{3.68}$$

where $\mathbf{Y}^T(k)$ is the matrix transpose of $\mathbf{Y}(k)$.

The next step is to find the generic equation for $Y(k)$ for any given time step k. This can be done by looking at the first few terms in the expansion of $Y(k)$ and identifying the generic equation.

$$\begin{aligned}
\mathbf{Y}(1) &= \mathbf{C}A_d\mathbf{x}(0) + \mathbf{C}B_d\mathbf{u}(0) \\
\mathbf{Y}(2) &= \mathbf{C}A_d\mathbf{x}(1) + \mathbf{C}B_d\mathbf{u}(1) \\
&= \mathbf{C}A_d(A_d\mathbf{x}(0) + B_d\mathbf{u}(0)) + \mathbf{C}B_d\mathbf{u}(1) \\
&= \mathbf{C}{A_d}^2\mathbf{x}(0) + \mathbf{C}A_dB_d\mathbf{u}(0)(1) + \mathbf{C}B_d\mathbf{u}(1) \\
\mathbf{Y}(3) &= \mathbf{C}A_d\mathbf{x}(2) + \mathbf{C}B_d\mathbf{u}(2) \\
&= \mathbf{C}A_d(A_d\mathbf{x}(1) + B_d\mathbf{u}(1)) + \mathbf{C}B_d\mathbf{u}(2) \\
&= \mathbf{C}A_d(A_d(A_d\mathbf{x}(0) + B_d\mathbf{u}(0)) + B_d\mathbf{u}(1)) + \mathbf{C}B_d\mathbf{u}(2) \\
&= \mathbf{C}A_d({A_d}^2\mathbf{x}(0) + A_dB_d\mathbf{u}(0) + B_d\mathbf{u}(1)) + \mathbf{C}B_d\mathbf{u}(2) \\
&= \mathbf{C}{A_d}^3\mathbf{x}(0) + \mathbf{C}A_d^2B_d\mathbf{u}(0) + \mathbf{C}A_dB_d\mathbf{u}(1) + \mathbf{C}B_d\mathbf{u}(2)
\end{aligned}$$

Therefore

$$\mathbf{Y}(n) = \mathbf{C}{A_d}^n\mathbf{x}_0 + \mathbf{C}\sum_{i=0}^{n-1} {A_d}^i B_d\mathbf{u}(n-1-i) \tag{3.69}$$

where $\mathbf{x}_0$ is the initial conditions of $\mathbf{x}$.

The summing term in Equation 3.69 can be further simplified:

$$\sum_{i=0}^{n-1} {A_d}^i B_d\mathbf{u}(n-1-i) = \begin{bmatrix} {A_d}^{n-1} & {A_d}^{n-2} & ... & {A_d}^2 & A_d & I \end{bmatrix} B_d \begin{bmatrix} \mathbf{u}(0) \\ \mathbf{u}(1) \\ \mathbf{u}(2) \\ . \\ . \\ . \\ \mathbf{u}(n-1) \end{bmatrix} \tag{3.70}$$

From this we can define two matrices as follows:

$$\begin{bmatrix} {A_d}^{n-1} & {A_d}^{n-2} & ... & {A_d}^2 & A_d & I \end{bmatrix} B_d = \mathbb{F}$$

$$\begin{bmatrix} \mathbf{u}(0) \\ \mathbf{u}(1) \\ \mathbf{u}(2) \\ . \\ . \\ . \\ \mathbf{u}(n-1) \end{bmatrix} = \mathbb{X}$$

Substituting back into Equation 3.69:

$$\mathbf{Y}(n) = \mathbf{C}{A_d}^n\mathbf{x}_0 + \mathbf{C}\mathbb{F}\mathbb{X} \tag{3.71}$$

Finally the equation for $Y^T(n)Y(n)$ can be found in terms of control inputs and initial values:

$$\begin{aligned} Y^T(n)Y(n) &= [{\mathbf{x}_0}^T({A_d}^n)^T\mathbf{C}^T + \mathbb{X}^T\mathbb{F}^T\mathbf{C}^T][\mathbf{C}{A_d}^n\mathbf{x}_0 + \mathbf{C}\mathbb{F}\mathbb{X}] \\ &= {\mathbf{x}_0}^T({A_d}^n)^T\mathbf{C}^T\mathbf{C}{A_d}^n\mathbf{x}_0 \\ &+ 2{\mathbf{x}_0}^T({A_d}^n)^T\mathbf{C}^T\mathbf{C}\mathbb{F}\mathbb{X} \\ &+ \mathbb{X}^T\mathbb{F}^T\mathbf{C}^T\mathbf{C}\mathbb{F}\mathbb{X} \end{aligned} \tag{3.72}$$

Using Equation 3.72, Equation 3.67 and Equation 3.68 can be constructed by summing the value of the equation over all time points. In addition Equation 3.61 can be used to generate x, y, $\dot{x}$ and $\dot{y}$ values over time which can be used to calculate constraints and plot generated paths.

3.4.2 Problems with a single optimising action

The algorithm using the cost function of the form shown in Equation 3.18 was unable to produce paths as desired. Figure 3.7 shows this clearly as the path generated fails to perform the transfer. This issue presents itself for all path exit angles greater than 15°. To analyse this problem the gradient of the cost function with respect to the control inputs u_x, u_y and the control for final time, t_f, step size h was calculated.

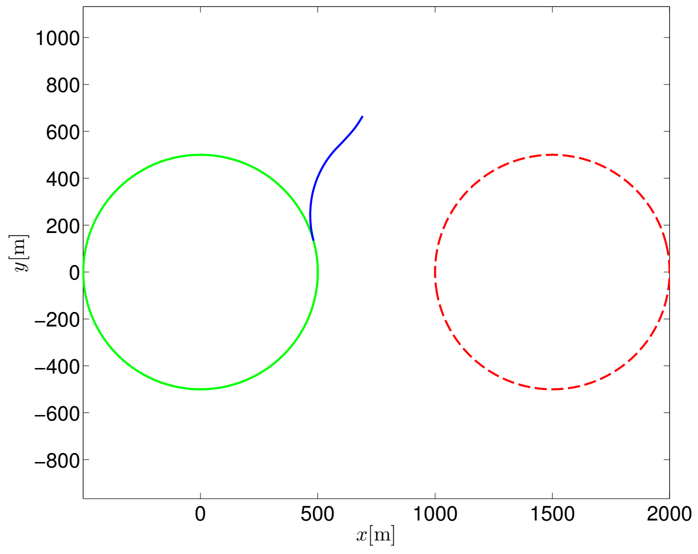

FIGURE 3.7: The path generated by a single optimising action from 15°exit angle

Due to the algorithm producing discrete results the norm of gradient values wrt u_x, u_y was taken with infinite gradient values being replaced with the maximum real value of the norm. This was plotted with the absolute values wrt h to allow the comparison of the largest gradient values. Figure 3.8 shows this for an exit angle of -90°. Gradient figures for other exit angles can be seen in Appendix A.

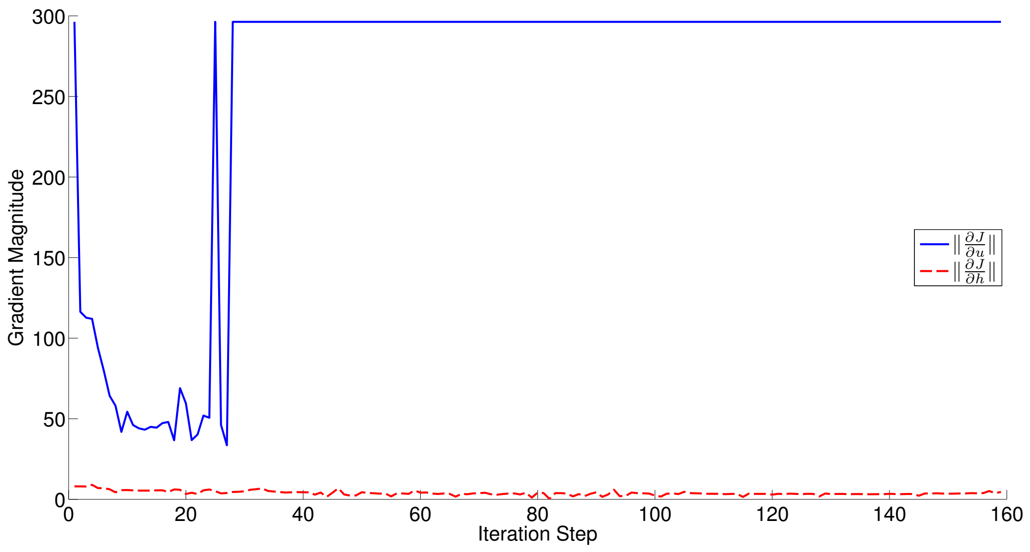

FIGURE 3.8: Gradient comparison for -90°

It is clear from these figures that the gradient wrt to u_x and u_y shown by the solid blue line is much larger than the gradient wrt h shown by the dashed red line:

$$\frac{\partial J}{\partial u} \gg \frac{\partial J}{\partial h} \tag{3.73}$$

where $h = t_f/N$, causing this issue.

This gradient stiffness results in the single optimising action terminating at local solutions instead of the desired global solutions. Due to this the form of the optimiser needs to be changed to overcome this limitation.

3.4.3 Using a Split Optimiser

It is possible to take the single optimising action shown in Equation 3.18 and split it into two distinct actions:

$$\begin{aligned}&\underset{t_f}{\text{Minimise}}\,(J_{\text{inner}})\\ :=\ &\underset{t_f\in[0,\infty)}{\text{Minimise}}\left(\underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}}\, J=\int_{t_0}^{t_f} f(u)dt\right)\end{aligned} \tag{3.74}$$

The inner optimiser generates the control values to follow a path at a given final time. It results in a valid or invalid path at the set final time, and is the optimiser responsible for generating the path shape while ensuring that valid paths meet the set constraints. The outer optimiser adjusts the final time of the path. In essence this optimiser attempts to minimise the time taken to fly a distance optimal transfer path. However it was not known, if the action of splitting the optimisation process into two steps, could yield different results depending on the choice of cost function.

All costs generated from either Equation 3.60a or Equation 3.60b will lie between limits generated from Equation 3.4. The limits are generated from the accelerated and minimum speed profiles shown in Figure 3.9. Using the chosen values of maximum speed V_{max}, minimum speed V_{min} and maximum control magnitude u, the values of t'_f, the final time for accelerated transfer, t_f, the final time for minimum speed transfer, and t_1, the time that the UAV reaches the maximum velocity, can be represented:

$$t_f=\frac{d}{V_{\text{min}}},\quad t_1=\frac{\Delta V}{u},\quad t'_f=\frac{d}{V_{\text{max}}}+\frac{\Delta V^2}{2uV_{\text{max}}} \tag{3.75}$$

where d is the path length, $u=\sqrt{u_{x\text{max}}^2+u_{y\text{max}}^2}$ and $\Delta V=V_{\text{max}}-V_{\text{min}}$.

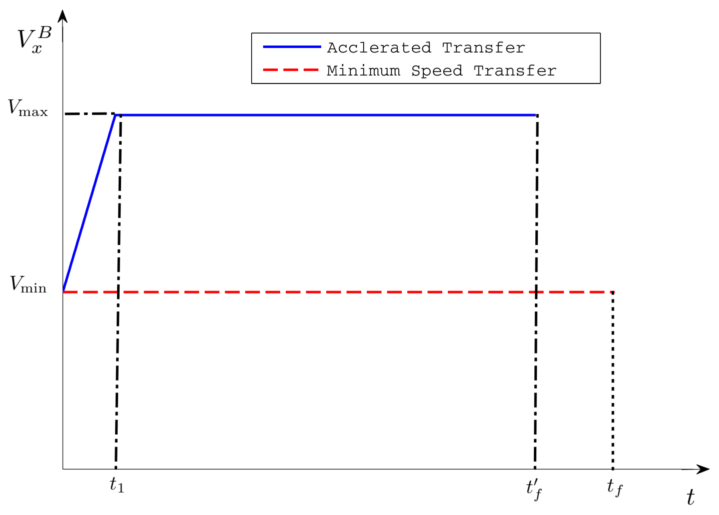

FIGURE 3.9: Speed Transfer Profiles

The limits on optimal cost, J^* can be calculated for Equation 3.60a and Equation 3.60b.

3.4.3.1 Cost function with $f(t) = \dot{x}^2 + \dot{y}^2$

The cost function, J_{inner}, with $f(t) = \dot{x}^2 + \dot{y}^2$ for the minimum speed profile, becomes

$$\begin{aligned} J_{\text{inner}} &= \underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}} \int_0^{t_f} \left(\dot{x}^2 + \dot{y}^2\right) dt \\ &= \int_0^{t_f} V_{\min}^2 dt = V_{\min}^2 t_f = V_{\min} d \qquad (3.76) \end{aligned}$$

This can be repeated for the accelerated speed profile:

$$\begin{aligned} J_{\text{inner}} &= \underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}} \int_0^{t_1} \left(\dot{x}^2 + \dot{y}^2\right) dt + \int_{t_1}^{t'_f} \left(\dot{x}^2 + \dot{y}^2\right) dt \\ &= \int_0^{t_1} (V_{\min} + ut)^2 dt + \int_{t_1}^{{t_f}'} {V_{\max}}^2 dt \qquad (3.77) \\ &= {V_{min}}^2 t_f + V_{min} u {t_1}^2 + \frac{u{t_1}^2}{2} + {V_{max}}^2 \left({t_f}' - t_1\right) \qquad (3.78) \\ &= \frac{{V_{min}}^2 \Delta V}{u} + \frac{V_{min} \Delta V^2}{u} + \frac{\Delta V^3}{3u} + V_{max} d + \frac{V_{max} \Delta V^2}{2u} - \frac{{V_{max}}^2 \Delta V}{u} \qquad (3.79) \\ &= V_{max} d - \frac{\Delta V^2 \left(V_{max} - 2V_{min}\right)}{6u} \qquad (3.80) \end{aligned}$$

Therefore the limits on J^* when $f(t) = \dot{x}^2 + \dot{y}^2$ are:

$$V_{\text{max}}d - \frac{\Delta V^2 \left(V_{\text{max}} - 2V_{\text{min}}\right)}{6u} \leq J^* \leq V_{\text{min}}d \tag{3.81}$$

3.4.3.2 Cost function with $f(t) = \sqrt{\dot{x}^2 + \dot{y}^2}$

The cost function, J_{inner}, with $f(t) = \sqrt{\dot{x}^2 + \dot{y}^2}$ for the minimum speed profile, becomes

$$\begin{aligned} J_{\text{inner}} &= \underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}} \int_0^{t_f} \sqrt{\dot{x}^2 + \dot{y}^2} dt \\ &= \int_0^{t_f} V_{\text{min}} dt = V_{\text{min}} t_f = d \end{aligned} \tag{3.82}$$

And, for the accelerated speed profile:

$$\begin{aligned} J_{\text{inner}} &= \underset{\mathbf{u}(t)\in\mathbb{U}}{\text{Minimise}} \int_0^{t_1} \sqrt{\dot{x}^2 + \dot{y}^2} dt + \int_{t_1}^{t'_f} \sqrt{\dot{x}^2 + \dot{y}^2} dt \\ &= \int_0^{t_1} (V_{\text{min}} + ut) dt + \int_{t_1}^{t_f{}'} V_{\text{max}} dt && (3.83) \\ &= V_{min} t_1 + \frac{u{t_1}^2}{2} + V_{max}\left(t_f{}' - t_1\right) && (3.84) \\ &= \frac{V_{min}\Delta V}{u} + \frac{\Delta V^2}{2u} + d + \frac{\Delta V^2}{2u} - \frac{V_{max}\Delta V}{u} && (3.85) \\ &= d + \frac{\Delta V}{u}\left(V_{min} - V_{max}\right) + \frac{\Delta V^2}{u} && (3.86) \\ &= d + \frac{\Delta V^2}{u} - \frac{\Delta V^2}{u} && (3.87) \\ &= d && (3.88) \end{aligned}$$

Since the costs are independent of speed profile, $J^* = d^*$ where d^* is the distance corresponding to the optimal value of final time t_f^* when $f(t) = \sqrt{\dot{x}^2 + \dot{y}^2}$

3.4.3.3 Applying the outer cost function

The effect of the outer cost function, $\text{Minimise}_{t_f\in[0,\infty)}$, is to find the cost, J_{inner}, that corresponds to the optimal time i.e when final time is minimum for the minimum distance path. This occurs when the accelerated speed profile is used.

For $f(t) = \dot{x}^2 + \dot{y}^2$, the accelerated speed profile will only be used if the cost of the accelerated speed profile is less than the cost of the minimum speed profile. Or if:

$$V_{\max}d - \frac{\Delta V^2 \left(V_{\max} - 2V_{\min}\right)}{6u} < V_{\min}d \tag{3.89}$$

or

$$V_{\max}d - V_{\min}d < \frac{\Delta V^2 \left(V_{\max} - 2V_{\min}\right)}{6u} \tag{3.90}$$

$$d < \frac{\Delta V \left(V_{\max} - 2V_{\min}\right)}{6u} \tag{3.91}$$

This new constraint causes the cost function, Equation 3.60b, to become unfeasible as inputting sensible values for $V_{\max} = 30m/s$,$V_{\min} = 20m/s$ and $u = 4.2426m/s^2$ gives $d < -3.928$. This is clearly impossible as distance cannot be negative. Therefore using Equation 3.60b in the split optimiser will not generate correct time optimal results. This effect can be further shown by plotting the value of d for varying values of $V_{\max}$, $V_{\min}$ and u. Figure 3.10 shows the case mentioned above. It is clear that for d to be a usable value, V would need to take unrealistic values. Figure 3.11 shows the case when $u = 1.5426$. This still has similar issues although indicates that lower values of u could allow Equation 3.60b to be used. Figure 3.12 shows the case where $u = 0.14264$. Although this value for u is small and unrealistic it could be used to allow Equation 3.60b as a cost function as d is now a usable value for realistic V. Since $V_{\min} \not> V_{\max}$ these figures do not show information for this region.

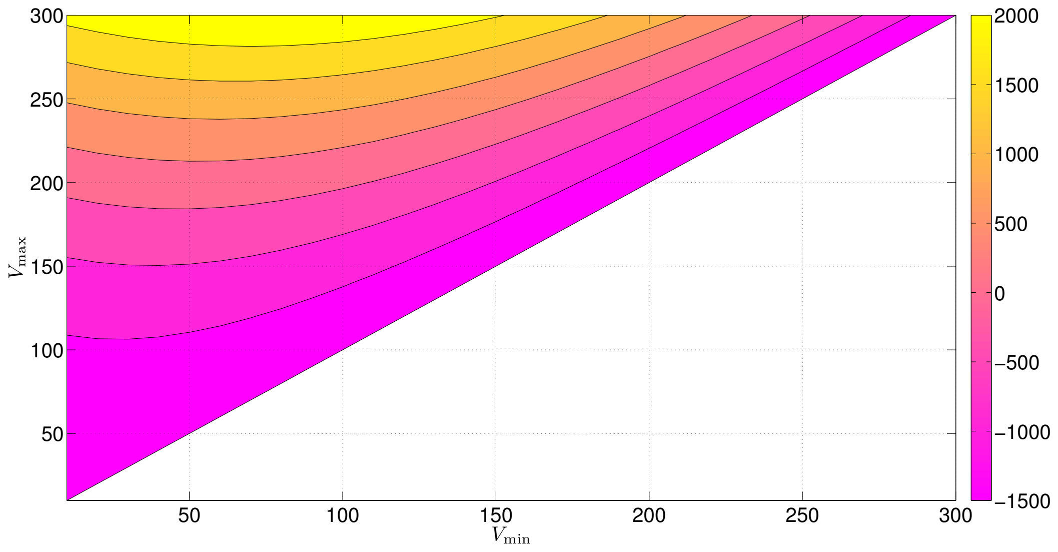

FIGURE 3.10: Speed Distance Contour for u = 4.2426

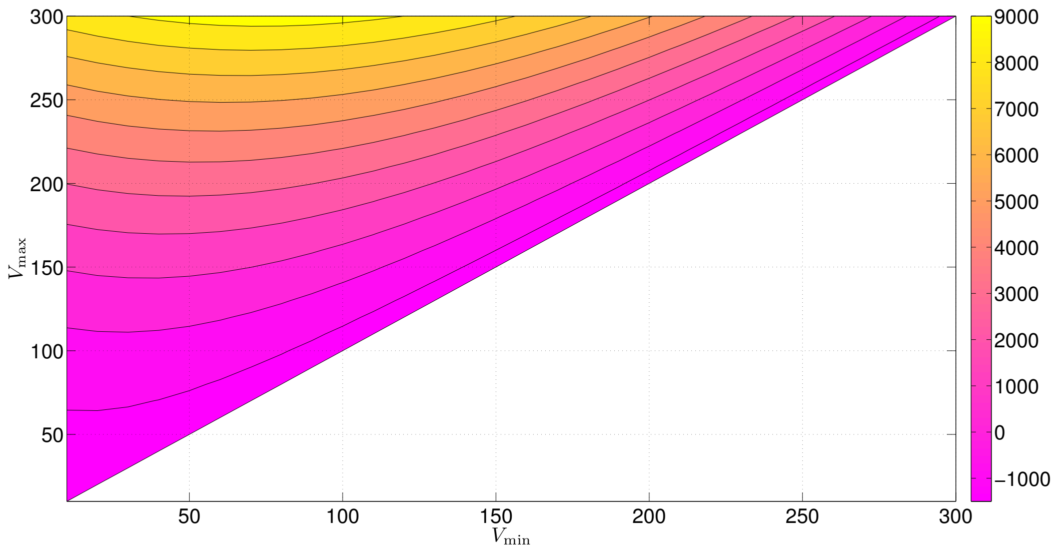

FIGURE 3.11: Speed Distance Contour for u = 1.5426

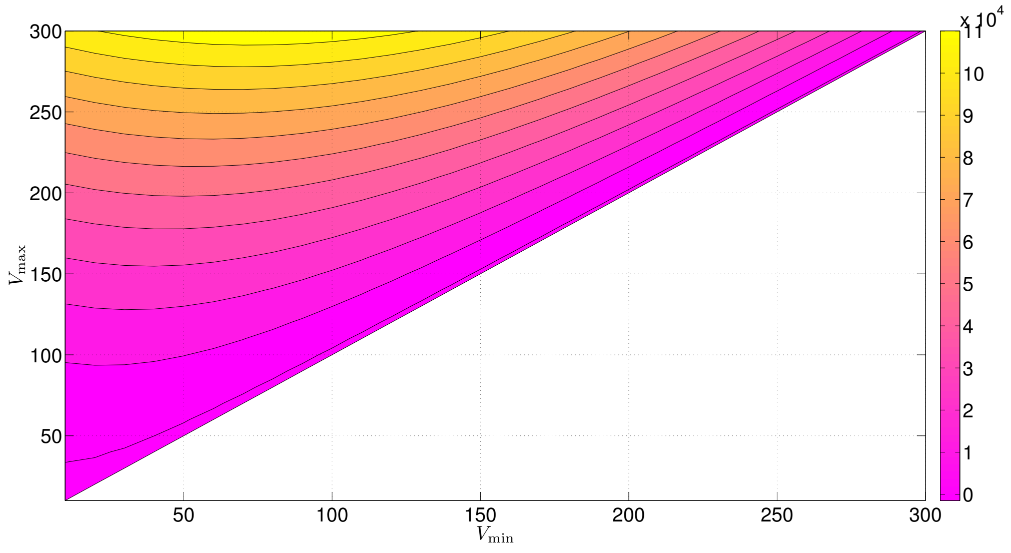

FIGURE 3.12: Speed Distance Contour for u = 0.14264

In a similar way when, $f(t) = \sqrt{\dot{x}^2 + \dot{y}^2}$, we are free to choose the cost value corresponding to any final time. Therefore the path that uses the optimal value of time, t_f^*, i.e. the time for the accelerated speed profile, is always available resulting in Equation 3.60a always generating correct time optimal results if applied correctly.

The generated paths for both cost functions can be plotted as can the velocity profiles generated for each. Figure 3.13 shows the two paths plotted together with the path for Equation 3.60b shown by the solid blue line and the path for Equation 3.60a shown by the dotted purple line.

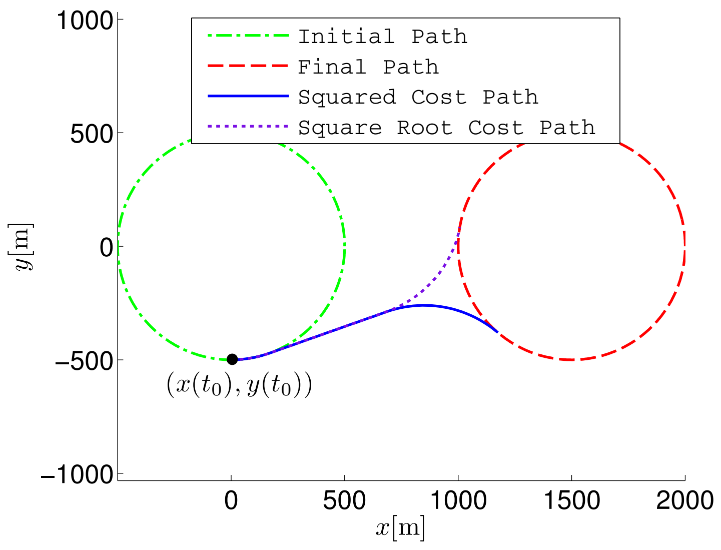

FIGURE 3.13: The path comparison for an exit angle of -90°for the cost functions Equation 3.60b and Equation 3.60a

It can be seen from Figure 3.13 that the generated path for each of the cost functions is significantly different due to the velocity transfer profile issue described. Figure 3.14 shows the velocity profile for each path and the issue can be clearly seen. The profile for(3.60b) can be seen to always sit on the $V_{\min}$ boundary line causing the difference in path shape and final time between Equation 3.60a which is describing an accelerated transfer.

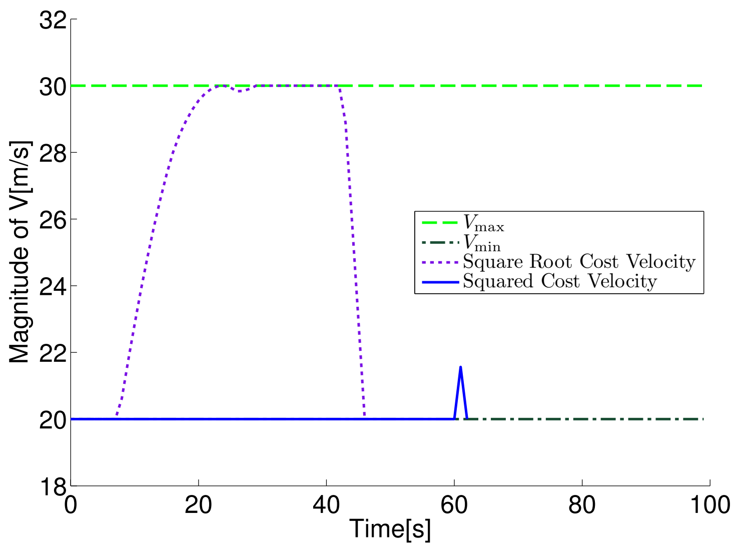

FIGURE 3.14: The velocity profile comparison for an exit angle of -90°for the cost functions Equation 3.60b and Equation 3.60a

3.4.4 Testing the Algorithm

Once a viable form of the algorithm was produced it was necessary to test the generated paths to ensure that the algorithm was working as expected. The tests mainly involve checking that the generated paths correctly meet all constraints verifying that they are correct optimal paths. The algorithm generates multiple paths at numerous different final times and then categorises them as either valid or non-valid paths based on the values of the constraint function for the path. The algorithm uses bisection to carry out the operation of the outer optimiser by first searching between a lower time bound of zero and a set upper time bound, that is set large enough to ensure all paths can be found, this operation aims to find the valid path with the shortest final time. Once this is found the bounds are adjusted so that the lower bound is the time for the shortest time valid path and the upper bound is the time for the next largest valid path. Paths are again searched for this time looking for the path with the lowest cost. This can be done as the cost function curve is assumed smooth and continuous. A flow diagram for this algorithm can be found in Appendix D

To test the algorithm four parameters must be checked:

- Velocity Magnitude:
  The magnitude of the UAV velocity can not be larger than the set value of $V_{\max}$ or smaller than the value of $V_{\min}$. All valid paths' velocity profiles should lie between these bounds whereas non-valid paths could break this constraint.
- Control Input:
  The control input values u_x and u_y are bound in a similar way to velocity and must lie within the boundaries set by $u_{x_{\max}}$, $u_{x_{\min}}$ $u_{y_{\max}}$ and $u_{y_{\min}}$. All valid paths should follow this rule while non-valid paths may break the boundary.
- Curvature:
  The UAV is constrained to turns no smaller than $r_{\min}$ or curvature values that lie between $\pm\frac{1}{r_{\min}}$
- The path that is selected as optimal has the minimum cost value of all the valid paths found.

Shown are figures for the an exit angle of -90°. The figures for all other exit angles are shown in Appendix B for completeness.

The validity of the algorithm can be investigated using these figures to check that the parameters mentioned above are adhered to. Firstly it can be seen that the paths that indicate as valid all lie within or on the boundary lines for velocity on Figure 3.15 showing that the velocity constraints are working correctly.

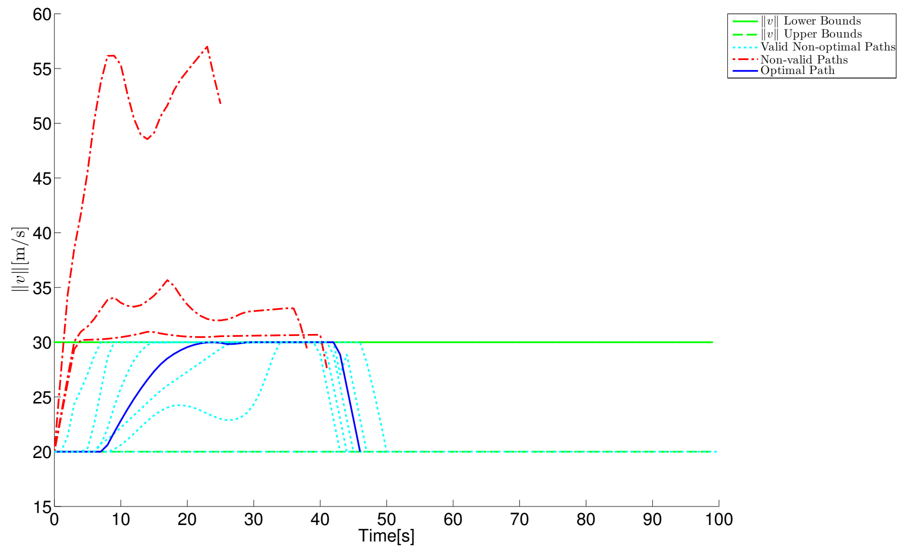

FIGURE 3.15: The velocity profiles for all paths found for an exit angle of -90°

Looking at the graph for control, Figure 3.16, a similar situation is shown where valid paths correctly lie between the boundaries set on the body control values.

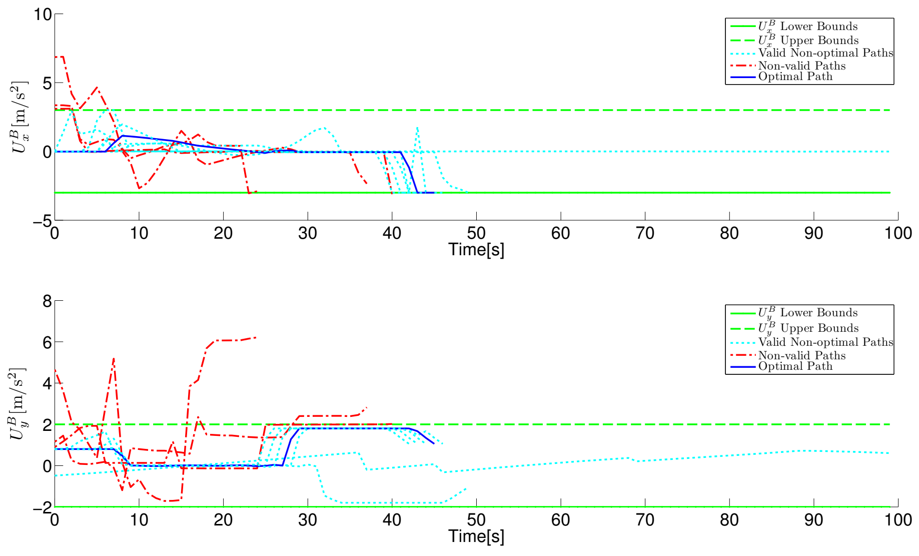

FIGURE 3.16: The U_x and U_y control history for all paths found for an exit angle of -90°

The final constraint of curvature, Figures 3.17, again shows this pattern with valid paths correctly lying between the set curvature bounds.

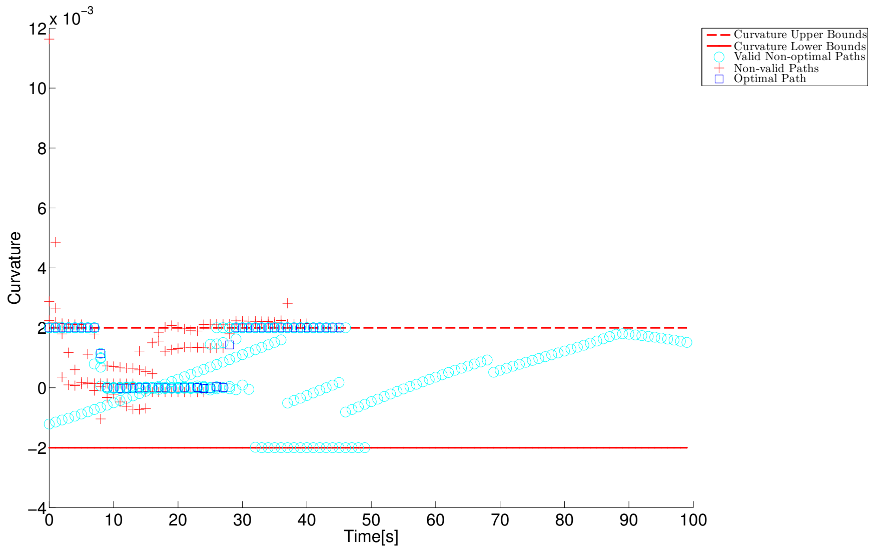

FIGURE 3.17: Path curvatures for all paths found for an exit angle of -90°

The actual path shape can be seen on Figure 3.18 and can be used to check the constraints on final position as paths that meet the final circle and don't violate any other constraints are valid.

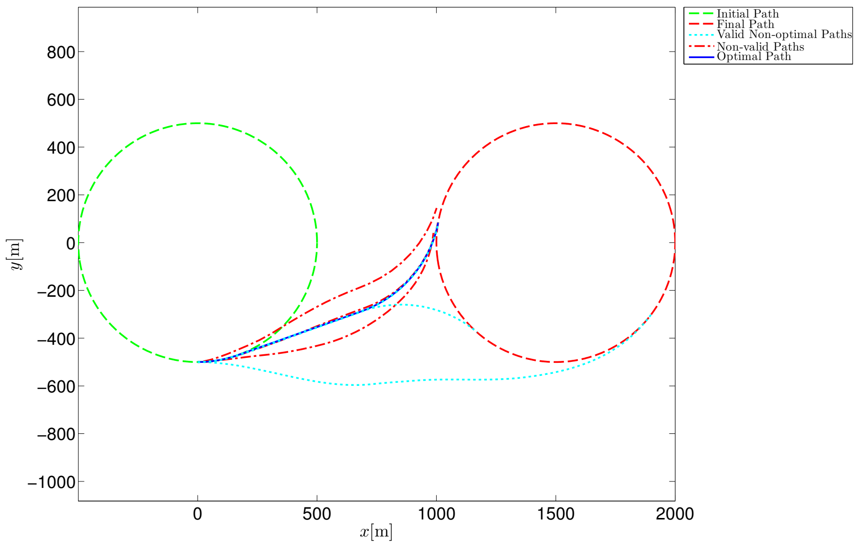

FIGURE 3.18: All paths found for an exit angle of -90°

The results shown by these figures verify that the constraint equations are being correctly activated allowing for the correct classification of paths as valid or non-valid, with those that are non-valid breaking at least one of the constraint equations outlined in Section 3.2.2. The correct classification of paths as valid or non-valid was essential to finding the optimal path as it was the key method used for managing the bounds of the bisection search use to handle the time minimisation operation. To ensure the algorithm is working as desired it is required to investigate the selection of the optimal path from all paths that are found to be valid. This path should have the minimum cost value out of all the valid paths. This can be investigated by looking at the figure of cost value at each value of final time. This is shown in Figure 3.19. It can be seen that the cost functions are continuous with the minimum point being correctly selected as the optimal path. From these results it is therefore sufficient to say that the shooting method algorithm performs as expected producing globally optimal paths that meet the set constraints.

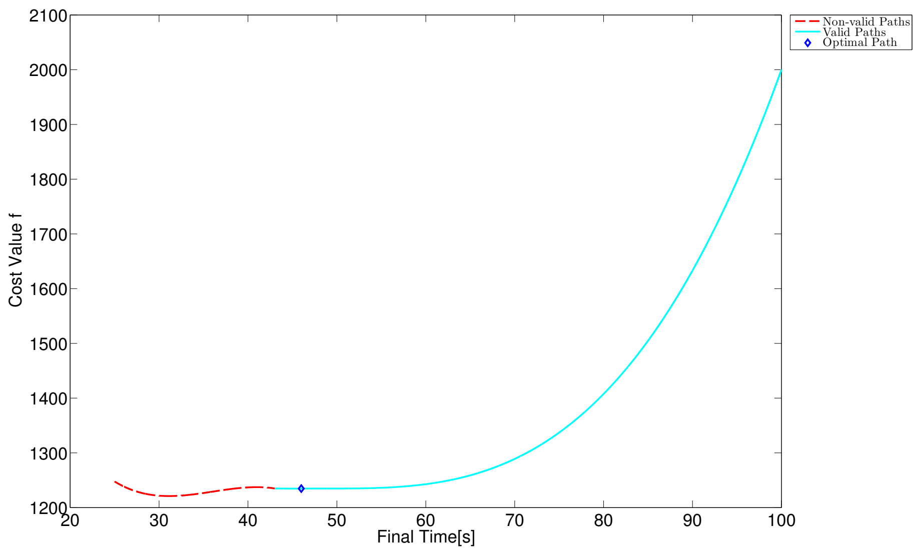

FIGURE 3.19: Cost curve for an exit angle of -90°

3.5 Summary

From the initial formulations the problem had been presented as a Two Point Boundary Value problem using a Hamiltonian formulation with Pontryagin's Minimum Principle. This method was subsequently shown not to yield usable results due to the inability to select an appropriate initial guess caused by the unknown nature of the solution space. This led to a move to represent the problem as a discrete shooting method algorithm as it only relied on the initial values which were well known. As part of the discrete formulation the effect of gradient on the resulting cost function was shown. Due to the gradient on the control values being much larger than that of time the algorithm would suffer from premature convergence indicating that the problem could not be solved as a single optimiser. To this end the problem was re-worked as a split optimiser with a discussion on the importance of cost function choice when using the split optimiser.

The effect of splitting the optimising action in effect added an unknown extra constraint regarding separation distance between the initial and final path and therefore reduced the viable cost functions that could successfully be used to solve the problem. Once an appropriate cost function was selected a series of tests were carried out to check the developed algorithm was correctly activating the constants and correctly selecting the path with minimum cost. These tests validated the output of the algorithm in terms of functional correctness. An investigation into the efficiency of the developed algorithm compared to existing methods is discussed in Chapter 4

Chapter 4

Comparing Shooting Results to Similar Path Planning Methods

4.1 Introduction

Up to this point the benefit of the path planning algorithm developed in Chapter 3 has not been investigated. The purpose of this chapter is to investigate the efficiency gain of the new algorithm when compared with other similar path planning methods.

The downside of the shooting method algorithm is the algorithm's long run time. Due to the complexity of the constraints the computational requirements of the algorithm are high making run times to find individual paths unsuitable for most real time applications unless significant hardware is available to process the algorithm in a shorter time-scale. Most current applications however do not have access to such hardware. The performance improvement of the developed path planning method can be compared to a similar real time path planner to further understand the benefit of the new method.

The trade-off between a real time algorithm and one that is offline is the trade-off between run time and accuracy. Real time algorithms use simplified constraints and system models in an attempt to perform the path planning operation in a run time that is suitable for the application. Due to these cut down systems the accuracy of the generated paths often suffer converging to local minimums instead of optimal solutions. Essentially this is the difference between methods that perform exhaustive searches and those that perform a 'good enough for practice' approach.

The two methods chosen for comparison are the well known Dubin's car experiment discussed in Chapter 2 and a receding horizon approach commonly used for path planning tasks such as this.

4.2 Viable Paths for Comparison

The first stage in the comparison process is to identify the paths from the shooting algorithm that provide unique results. It is apparent that only a select number of paths are unique with the initial exit angle being the differentiator between unique and non-unique paths. Figure 4.1 is an example of a unique path for an exit angle of 60°.

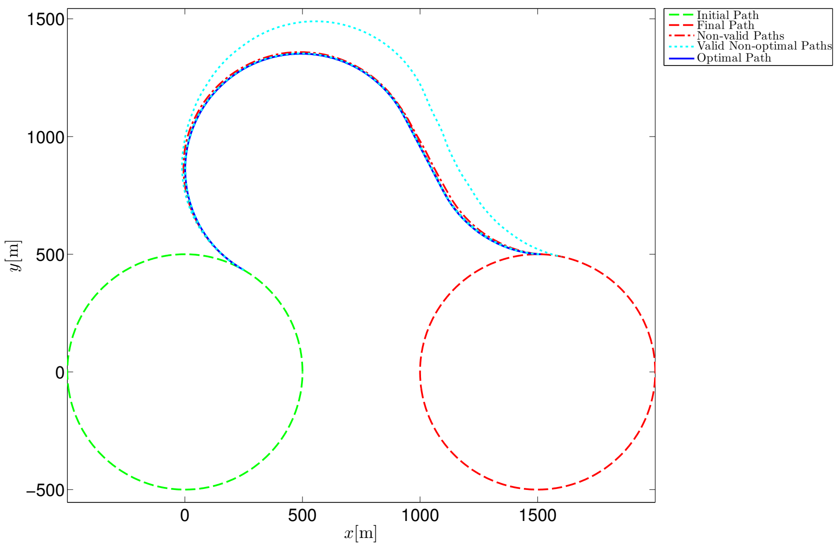

FIGURE 4.1: All paths found for an exit angle of 60°

Figure 4.2 however is not unique as the path from an exit angle of 75°actually exits the initial circular flight path between -90°and -75°for which a solution already exits.

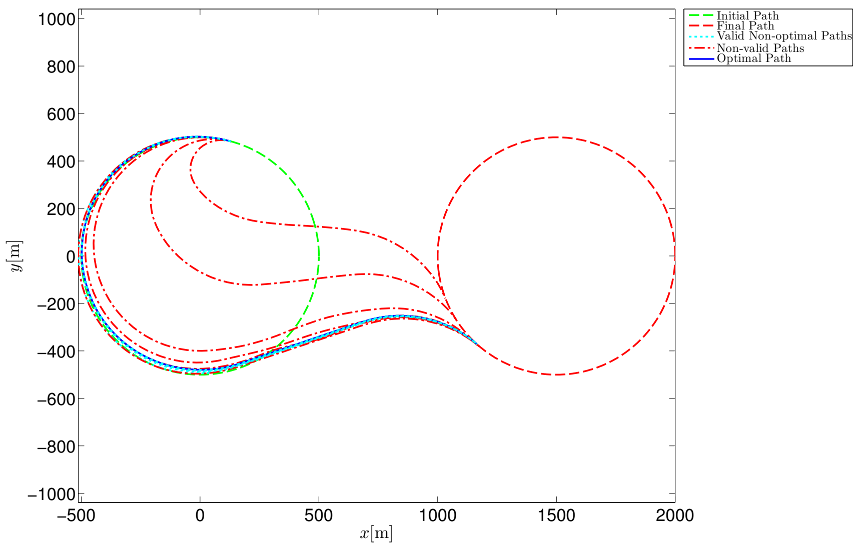

FIGURE 4.2: All paths found for an exit angle of 75°

This fact holds for all paths with an exit angle equal to and greater than 75°. Therefore it is only necessary to consider paths with exit angles between -90°and 60°as any path outside this range will simply follow its initial circular flight path until exiting within this unique transfer region. Since this is the case, only paths from exit angles within the unique transfer region need to be considered for comparison. This result is in itself a further validation of the shooting method output as it is producing paths that are intuitive when compared to the actions a human operator would perform. In this case it would be expected that after a certain point turning the UAV off its initial flight path to make the transfer is not sensible as it is flying away for the desired transfer destination. In these cases it would be sensible to fly the UAV on its original circular path and exit when closer to the transfer destination.

4.3 Dubin's Comparison

Dubin's results have been a long standing way to solve problems such as this as it simplifies the required paths to a set of curved and straight sections with the curvature of the curved sections maintained at the minimum turn radius. The Dubin's result that mimics the result of the shooting algorithm can be seen in Figure 4.3 and is simply the tangent lines between the two circular flight paths from an exit angle of -90°and -48.19°. For the UAV to transfer between the two circular path using the Dubin's result it must fly around its original circular flight path until it reaches one of the two tangent lines to make the transfer. The tangent line that provides the shortest transfer will depend on where on the initial flight path the UAV decides to exit, with the tangent lines having a set length as shown:

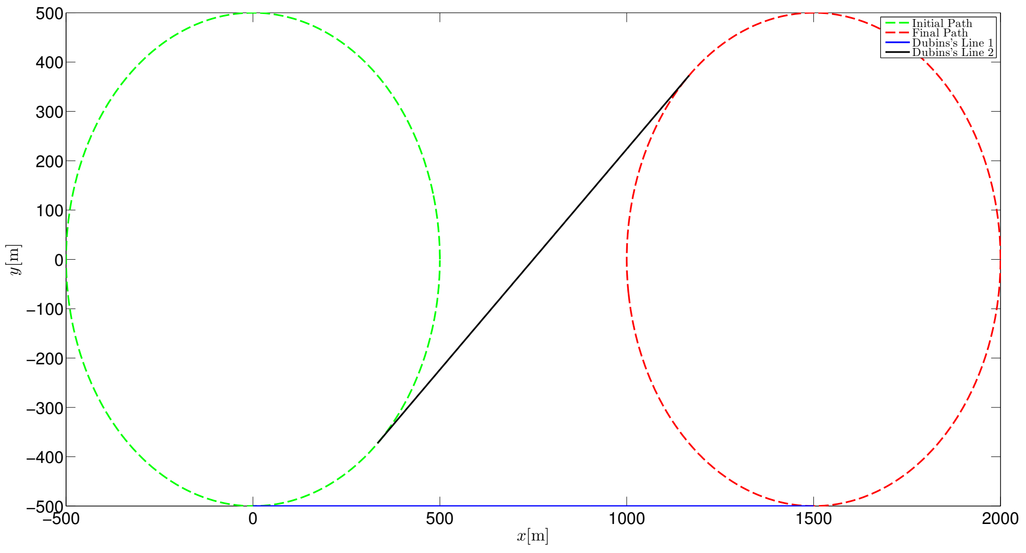

FIGURE 4.3: Dubin's straight line path from-90°

$$-90^\circ < \theta \leq -48.19^\circ \xrightarrow[exitpath]{-48.19^\circ} \text{Length} = 1118.04\text{m}$$

$$\text{otherwise} \xrightarrow[exitpath]{-90^\circ} \text{Length} = 1500\text{m}$$

Adding the relevant tangent line length to the distance the UAV must fly around its initial circular path to reach the exit angle will give the total cost of the Dubin's transfer for comparison against the shooting results. Table 4.1 shows the resulting values:

Exit angle(°)	Dubin's Cost (m)	Shooting Cost (m)	Efficiency vs Dubin's(%)
-90	1,500.00	1,234.78	17.68
-75	1,352.00	1,103.87	18.35
-60	1,221.10	974.12	20.23
-45	1,892.70	861.61	54.48
-30	2,023.60	806.13	60.16
-15	2,154.50	872.90	59.48
0	2,285.40	1,150.76	49.65
15	2,416.30	1,571.87	34.95
30	2,547.20	1,992.87	21.76
45	2,678.10	2,356.50	12.01
60	2,809.00	2,706.29	3.66

Table 4.1: Cost comparison table for Dubin's cost for different exit angles

The paths found by the shooting algorithm can be seen to be more efficient than those produced by Dubin's. The ability to generate paths from the full curvature range instead of being restricted to only the minimum radius turn can yield a path length improvement of up to $\sim 60\%$ which is a significant improvement over Dubin's method.

4.4 Receding Horizon Comparison

From the algorithm described in Section 2.6.6 it necessary to obtain a data set to which the shooting algorithm results can be compared. Repeated runs of the receding horizon algorithm are carried out for each exit angle varying the values of T and Δt each time. A sensible range for these values can be decided noting that $\Delta t \ngtr T$. For each run the path length from horizon start to optimal end is calculated. Using this information a cost contour plot for each exit angle is created. Figure 4.4 shows an examples of these plots for an exit angle of -90°. The contour plots for the remaining exit angles can be found in Appendix B.

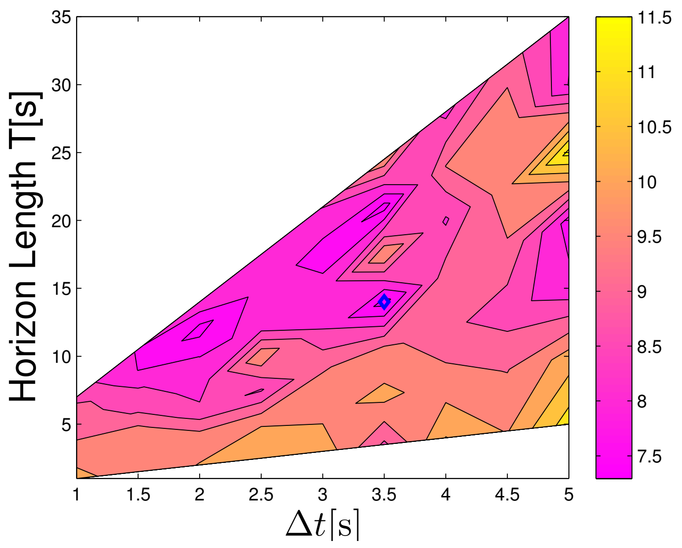

FIGURE 4.4: Cost contours for a Receding Horizon from -90°over various horizon lengths and Δt values

To handle issues of large scale values, the plotted points are the natural log of actual cost values. The blue diamonds indicate the minimum cost values. To compare the cost values from the receding horizon algorithm to the shooting method it was first required to take the many receding horizon results that exist due to the varying T and Δt combinations and obtain a single cost value for comparison for each exit angle. A sensible measure for this was the average value of the cost, however it quickly became clear that this value was not a suitable measure. Due to the large variation in cost values over the range of T and Δt, the average cost is significantly large in comparison to the shooting cost giving what would appear to be large and unrealistic efficiency gains between the two algorithms. Although an efficiency gain is hypothesised due to the larger range of path curvatures available to the shooting algorithm when compared to the receding horizon method the large efficiency gains as seen in Table 4.2 are not true gains but are instead artifacts of improper T and Δt choices resulting in skewed average cost values. Therefore the cost value best used for the comparison is the minimum produced by the receding horizon algorithm. This value is the result of the best combination of T and Δt and will be the closest to the shooting method cost. Using the minimum value, Table 4.3 can be produced, this shows the efficiency gain provided by the shooting algorithm for each exit angle.

Exit angle(°)	Average Cost (m)	Shooting Cost (m)	Efficiency vs Average(%)
-90	14672.70	1234.78	91.58
-75	18013.81	1103.87	93.87
-60	9677.30	974.12	89.93
-45	12505.56	861.61	93.11
-30	9617.94	806.13	91.62
-15	9754.44	872.90	91.05
0	10477.14	1150.76	89.02
15	9386.67	1571.87	83.25
30	10588.73	1992.87	81.18
45	10104.60	2356.50	76.68
60	10735.56	2706.29	74.79

Table 4.2: Cost comparison table for average cost for different Receding Horizon exit angles

Exit angle(°)	Minimum Cost (m)	Shooting Cost (m)	Efficiency vs Minimum(%)	Horizon Length T(s)	Δt(s)
-90	1470	1234.78	16	14	3.5
-75	1240	1103.87	10.98	10	2
-60	1150	974.12	15.29	10	2.5
-45	1000	861.61	13.84	15	2.5
-30	900	806.13	10.43	17.5	2.5
-15	1280	872.90	30.80	14	2
0	1540	1150.76	25.28	14	3.5
15	1750	1571.87	10.18	15 & 17.5	2.5
30	2100	1992.87	5.10	17.5	2.5
45	2430	2356.50	3.02	31.5	4.5
60	2820	2706.29	4.03	12 & 15	3

Table 4.3: Cost comparison table including T and Δt values for minimum cost for different Receding Horizon exit angles

In addition Table 4.3 shows the T and Δt values that result in the minimum receding horizon cost for each start point. This lookup table provides a better starting guess for the T and Δt values for a receding horizon algorithm performing this type of path planning. This results in an algorithm that can work in real time that also attempts to minimise the efficiency loss when compared to an offline equivalent through appropriate T and Δt selection.

4.5 Summary

By comparing the cost values from the shooting algorithm against the equivalent cost values from both Dubin's results and an equivalent receding horizon approach it has been possible to gauge the performance improvement that the shooting algorithm developed in Chapter 3 gives over similar methods. The algorithm is shown to gives significant performance improvements in terms of generated path length against both of the comparison algorithms. This result was expected due to the shooting method algorithm's greater choice in selecting available paths. The simple addition of the curvature constants has allowed a larger degree of flexibility in the generated transfer paths resulting in the increased efficiency (shorter transfer distance) over other methods. It can be noted that by increasing the number of nodes within a horizon and/or by allowing more than three available paths per node for the receding horizon approach could more closely approximate the shooting method results, however this would in turn impact the run time of a receding horizon approach.

Chapter 5

Comparison of the Shooting Transfer Against a Flexible Transfer Case

5.1 Introduction

One of the questions that arises when applying the shooting transfer developed in Chapter 3 is "Could better performance be achieved if a more flexible transfer algorithm was used?"

Up to this point the developed algorithm could be considered relatively generic. Although the fact that it minimises distance is beneficial when considering the target tracking mission (the UAV will always be flying towards its target if distance is minimised) to be carried out once on the final circular path the algorithm itself could in essence be applied to any application when moving between a known initial point and final circular path is required. Although this makes the algorithm more widely applicable than just to a single mission its suitability for the transfer to a target tracking mission could be called into question.

This chapter looks at this situation by applying a more flexible algorithm to the target tracking mission through the development of a new switching method. The switching method incorporates information about target movement to select the best cost function depending on the situation. By investigating the effect of both the switching and direct method on two visibility indices, that are key to the performance of the target tracking mission, a comparison can be carried out. This allows the performance of the shooting transfer and switching transfer to be compared based on their influence on these key parameters.

5.2 Switcher Design

From the results shown in 3.4.3 it became clear that by simply altering the power of the standard cost function:

$$J = \int_{t_0}^{t_f} s \tag{5.1}$$

three cost functions could be obtained each performing a different minimising action based on the power of the original function.

$$J = \int_{t_0}^{t_f} s^0 \tag{5.2}$$

$$J = \int_{t_0}^{t_f} s \tag{5.3}$$

$$J = \int_{t_0}^{t_f} s^2 \tag{5.4}$$

With Equation 5.2 providing a time optimal transfer, Equation 5.3 providing a distance optimal transfer and Equation 5.4 providing a minimum velocity transfer. These three cost functions could then be used to create a switching algorithm where the most appropriate cost function for the current operating environment is used to solve the path planning problem. Consider the following environments attaching an appropriate cost function to each:

Figure 5.1: A UAV transferring to track a target where the centre of the tracking circle is at a distance of less than r away. This relates to Equation 5.2 as the UAV is now effectively too close to the target. The goal is to move onto a tracking trajectory as quickly as possible so that the probability of keeping the target in track is maximised. In addition the UAV is attempting to maintain a minimum distance from the target which has been broken in this environment so it is desirable that it re-establishes its normal flying state as quickly as possible hence why the cost function selected is that which minimises time.

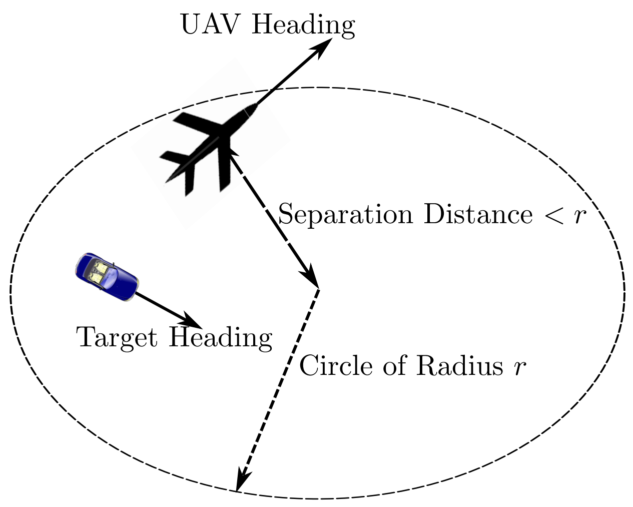

Figure 5.1: Transfer environment for a UAV transferring to track a target where the centre of the tracking circle is at a distance of less than r away

Figure 5.2: A UAV transferring to track a target where the centre of the tracking circle is at a distance of greater than r away and the target is not travelling towards the UAV. This relates to Equation 5.3 as this is the environment of operation equivalent to the shooting method algorithm with one exception, that the target and UAV cannot be moving towards each other within a set threshold. It is therefore desirable to transfer in the fastest time with the shortest distance to ensure that the transfer onto the desired tracking path is carried out as efficiently as possible, hence the use of the distance optimal cost.

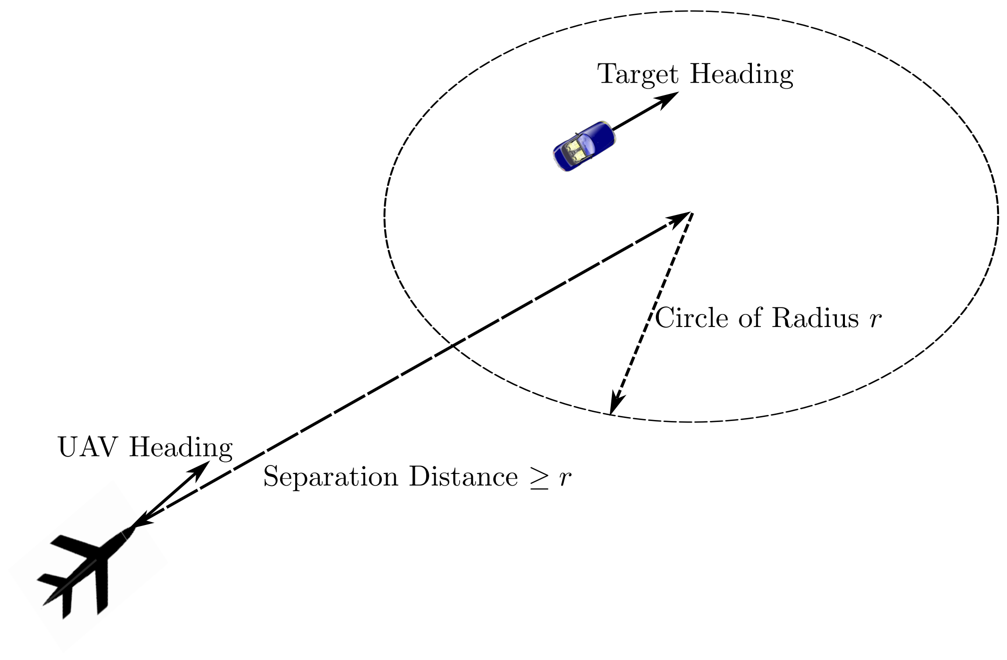

FIGURE 5.2: Transfer environment for a UAV transferring to track a target where the centre of the tracking circle is at a distance of greater than r away and the target is not travelling towards the UAV

Figure 5.3: A UAV transferring to track a target where the centre of the tracking circle is at a distance of greater than r away and the target is travelling towards the UAV. This relates to Equation 5.4. This cost function is the same as that originally rejected in Section 3.4.3.3 as it produced transfer paths that were always at the minimum velocity. Although not useful for performing distance optimal transfers it can be used for cases where the environment requires a slower transfer in reaction to target movement. In this case the target heading is towards the UAV meaning that it is possible that the UAV could pass over the target which is undesirable as it could cause a loss of line of sight (LOS) between the UAV cameras and the target. In this situation it would be beneficial for the aircraft to perform a minimum speed transfer to reduce the possibility of the target being lost, hence this cost function becomes useful.

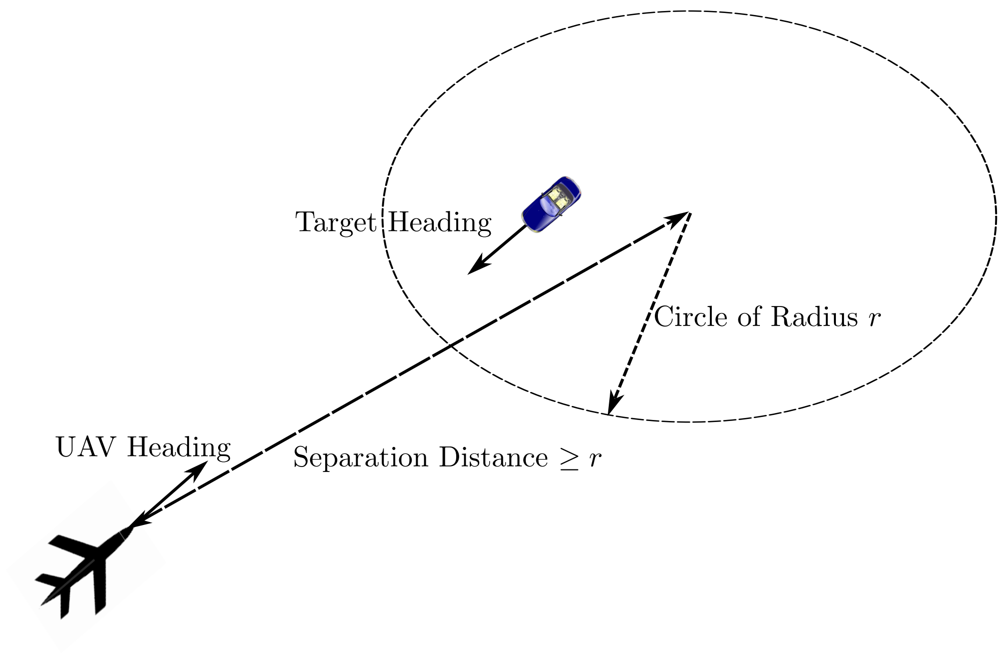

FIGURE 5.3: Transfer environment for a UAV transferring to track a target where the centre of the tracking circle is at a distance of greater than r away and the target is travelling towards the UAV

The final stage is to decide upon the switching conditions that cause the change between the three cost functions. If we consider the generic case shown in Figure 5.4

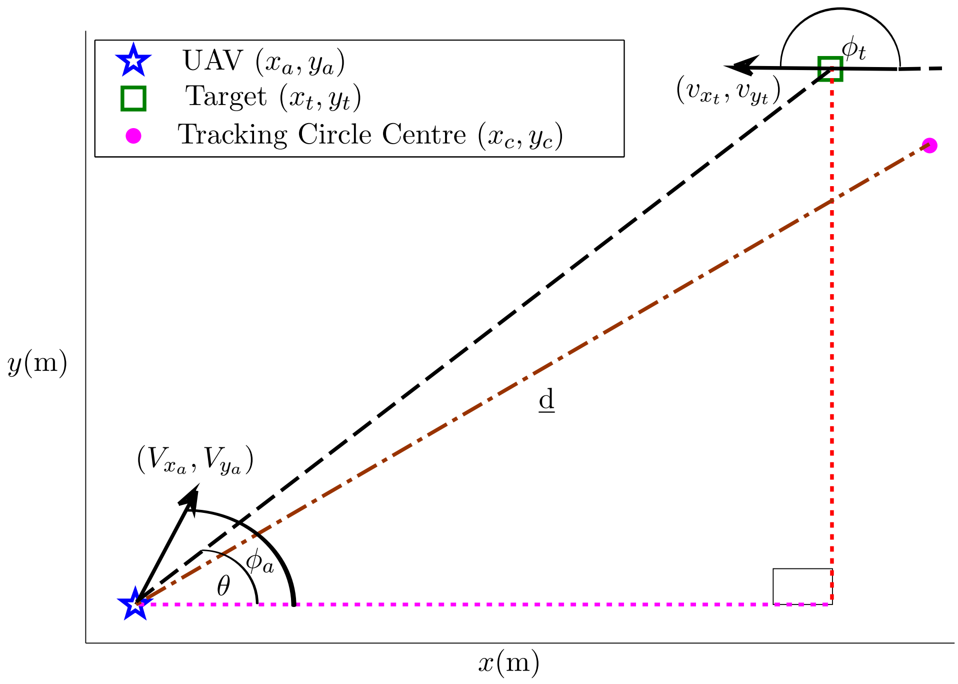

FIGURE 5.4: Generic example of UAV and target separation

where,

$$\phi_a = tan^{-1}\left(\frac{v_{y_a}}{v_{x_a}}\right) \tag{5.5}$$

$$\phi_t = tan^{-1}\left(\frac{v_{y_t}}{v_{x_t}}\right) \tag{5.6}$$

$$\theta = tan^{-1}\left(\frac{(y_t - y_a)}{(x_t - x_a)}\right) \tag{5.7}$$

$$\underline{\mathrm{d}} = \sqrt{(x_c - x_a)^2 + (y_c - y_a)^2} \tag{5.8}$$

it is possible to generate a number of switching conditions. Firstly the conditions based on separation distance:

$$\underline{\mathrm{d}} < r_{\text{min}} \tag{5.9}$$

$$\underline{\mathrm{d}} \geq r_{\text{min}} \tag{5.10}$$

where $r_{\text{min}} = r$ the minimum turning radius.

Lastly the condition based on heading angle needs to be identified to allow proper switching if the UAV and target are moving towards each other within a given threshold.

$$|[\phi_a - \theta] - [\phi_t - (\frac{\pi}{2} - \theta)] + \pi| > \psi \tag{5.11}$$

$$|[\phi_a - \theta] - [\phi_t - (\frac{\pi}{2} - \theta)] + \pi| \leq \psi \tag{5.12}$$

where, ψ is the threshold on the difference in resulting heading angle between the UAV and the target. This value should be relatively small as it is desirable only to enter the minimum speed operating mode only for resulting angles around 0 radians so as to operate in the more efficient distance or time optimal modes for the widest range of resulting headings possible. In this case a value of $\pm\frac{\pi}{8}$ radians or $\pm 22.5°$was chosen as acceptable.

The full switching cost function is therefore,

$$\underset{t_f \in [0,\infty)}{\text{Minimise}} \left(\underset{\mathbf{u}(t) \in \mathbb{U}}{\text{Minimise}}\, J \right) \tag{5.13}$$

where $\mathbb{U}$ is a compact set defined by Equation 3.7, t_0 is the initial time and t_f is the final time, which is free.

where,

$$J = \begin{cases} \int_{t_0}^{t_f} dt & \text{for Equation 5.9 and Equation 5.11} \\ \int_{t_0}^{t_f} \sqrt{\dot{x_a}^2 + \dot{y_a}^2} dt & \text{for Equation 5.10 and Equation 5.11} \\ \int_{t_0}^{t_f} (\dot{x_a}^2 + \dot{y_a}^2) dt & \text{for Equation 5.10 and Equation 5.12} \end{cases}$$

subject to

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$

and

$$\begin{aligned} v_{\text{min}}^2 \leq v_x^2 + v_y^2 \leq v_{\text{max}}^2 \\ u_{x_{\text{min}}} \leq u_x \cos\phi + u_y \sin\phi \leq u_{x_{\text{max}}} \\ u_{y_{\text{min}}} \leq -u_x \sin\phi + u_y \cos\phi \leq u_{x_{\text{max}}} \\ -\frac{1}{r_{\text{min}}} \left(v_x^2 + v_y^2\right)^{3/2} \leq v_x u_y - v_y u_x \leq \frac{1}{r_{\text{min}}} \left(v_x^2 + v_y^2\right)^{3/2} \end{aligned}$$

5.3 Defining the Performance Measures and Comparing the Transfer Algorithms

It is desired to test the effect of both the switching method and the shooting method when applied to the target tracking example described in 2.5, where a UAV is transferring to an optimal circular tracking path. To do this two performance measures are introduced denoting the probability of the target tracking the UAV, Equation 5.14, and the probability of the UAV tracking the target, Equation 5.15.

$$\rho_{\text{vis}}^{\text{target}} = \frac{\Sigma \Delta t_{\text{target}}^{\text{see}}}{T} \tag{5.14}$$

$$\rho_{\text{vis}}^{\text{UAV}} = \frac{\Sigma \Delta t_{\text{UAV}}^{\text{see}}}{T} \tag{5.15}$$

The target tracking mission in question is already designed to optimise these visibility indices through its placement of the tracking circle and therefore these are not directly controlled by the transferring path planner. These indices however are only at their optimal state when the UAV is actually flying the final circular path. During the transfer there is no direct control of the visibility indexes but depending on where the UAV is located and how it is flying the values of the indices will change. The desired outcome is to maximise Equation 5.15 while minimising Equation 5.14 even while transferring so that the UAV can potentially begin tracking the target before it reaches the tracking path. As the UAV gets closer to the tracking circle these indices will approach their optimal values. It is also at this point where the switching method could have most effect as being able to react to target motion during a path update could provide a beneficial change in these indices over the shooting method.

Using these visibility indexes a simulation was run to analyse the probability differences between the switching and shooting methods to test if providing mission information had an effect. The algorithm is set up so that the UAV calculates a path update every 10 seconds assuming that the target has randomly moved and the target tracking circle has therefore been updated. Each run lasts a total of 100 seconds or 10 path updates with visibility information for each second stored for use in calculating the visibility indices. One thousand runs for each method was completed with each method utilising the same random path information to provide a direct comparison. Figure 5.5 shows the comparison between the switching and shooting methods, it can be seen that there is no significant difference in the visibility indexes.

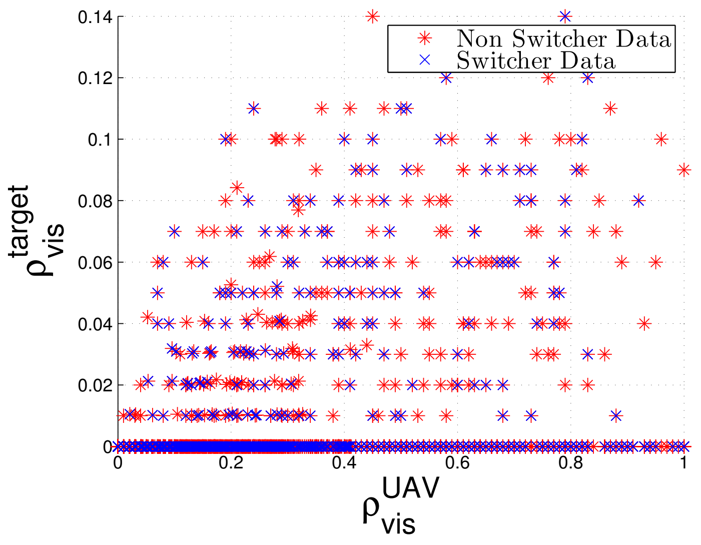

FIGURE 5.5: Comparison of Visibility Index of the Switching and Non Switching Cost Functions

This result indicates that the switching method is for the majority of the time activating the minimum distance cost which is the same cost utilised by the shooting method. There is some discrepancy between results indicating that the remaining switching states are activated on occasion but not enough to impact the visibility indexes significantly. When investigating the histogram plots. Figure 5.6 and Figure 5.7, for $\rho_{\text{vis}}^{\text{UAV}}$ for both methods it can also be seen that the number of paths that have greater than 60% probability of the UAV viewing the target during the transfer is higher for the non switching method.

This indicates that the non switching method is able to get the UAV closer to the final circular path than the Switching method. Therefore not only is the Shooting method the algorithm that is most used, it is also capable of manoeuvring the UAV closer to its intended destination.

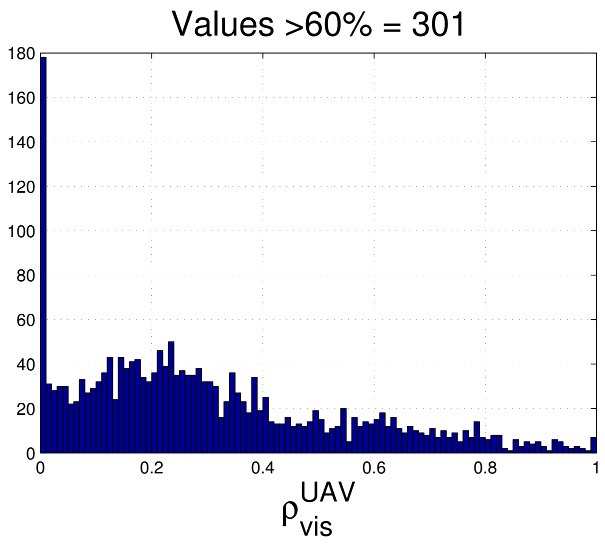

FIGURE 5.6: Histogram plot for the value of $\rho_{\text{vis}}^{\text{UAV}}$ for the non switching method

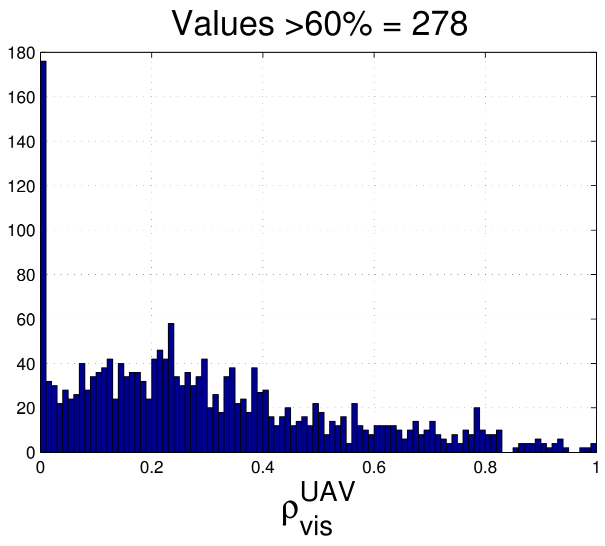

FIGURE 5.7: Histogram plot for the value of $\rho_{\text{vis}}^{\text{UAV}}$ for the switching method

5.4 Summary

A switching algorithm was developed to provide more flexibility in the transfer paths for a target tracking mission. Using key visibility indexes the transfer paths for the switcher were compared against the singular cost function transfer paths. The results indicated no significant difference in the visibility indexes for either method. Given that one of the switching modes was the singular cost function it is clear that during the transfer operation the UAV spends the majority of its time utilising the singular cost function to produce paths, and that adding flexibility to the path planning effort by providing more cost function choice yields no significant effect on the target tracking mission being carried out. In addition it was also seen that the single cost function path produced a higher number of results with greater than 60% visibility index indicating that this method was also able to move the UAV closer to the desired target tracking path that the switching method. These result further validate the single cost functions developed in Chapter 3 and its suitability for the target tracking mission posed.

Chapter 6

Conclusions and Further Work

6.1 Conclusions

Over the past three decades the use of unmanned aerial vehicles has been on the increase. Primarily driven by military applications, UAVs are being increasingly used in fields of operation where detrimental human factors (e.g. long operational time scales, risk of injury etc.) dictates the use of technologies that remove the human element from the immediate operational vicinity. The prevalence of information warfare has seen the role of reconnaissance take the forefront in UAV missions and over recent years has lead to a push in autonomous aircraft that can operate for many hours without the need for human control or intervention. This drive for UAV autonomy has seen the need for accurate and efficient path planning algorithms designed to generate the flight paths required for UAVs to carry out missions. The majority of these algorithms have been focused on specific mission parameters such as collision avoidance and target tracking resulting in a lack of algorithms for simple tasks such as transfer algorithms required for a UAV to move between its mission phases or to simply manoeuvre between way-points.

From the work on target tracking carried out by Dr Jongrae Kim, where a UAV was found to be capable of carrying out a target tracking mission by flying an optimal circular flight path, it became clear that the development of an algorithm capable of transferring a UAV onto or between these target tracking circles was required. The path type best suited for such a transfer was the most direct path resulting in the requirement that the algorithm worked to minimise distance leading to the derivation of the equation for path length from which the desired cost function could be constructed. A simple UAV model was used to develop the guidance law including the derivation of all constraints leading to an applicable minimisation equation for the transfer.

The first aim of this work was to look into the development of an algorithm capable of producing optimal solutions for this transfer task.

This work relaxed the constraint on the available solution space seen in similar methods by allowing paths to take any value as long as they conformed to certain constraints such as velocity magnitude and maximum curvature of turn. This is a closer approximation to how a human operator would control a UAV and facilitated the goal of producing globally optimal solutions for this problem rather than the local solutions of other methods arising from their use of a tightly constrained solution space

The work in this thesis looked primarily at the development of an algorithm that utilised a more realistic curvature constraint and provided an investigation into its performance in producing global solutions for the transfer problem in question.

Firstly an algorithm method had to be selected that was most appropriate for the problem. Due to the known information about the initial and terminal conditions a two point boundary value algorithm (TPBVA) was originally though to be most appropriate.

A Hamiltonian was formed for this problem by augmenting the cost function to provide the set of continuous differential equations required by the TPBVA which included the necessary constraint information with Pontryagin's Minimum Principle applied to handle the control constraints. Boundary condition were generated from the information known about initial and terminal points resulting in all the required equations for the TPBVA.

This type of approach is unfortunately highly susceptible to errors arising from improper initial guess values. This results in paths being generated that are not optimal or no paths being generated at all. In this case the unknown nature of the solution space meant that the chosen initial guess for the algorithm yielded no useful results and without more information being available this could not be corrected. As such a simpler approach was adopted using an initial value shooting method.

Initial value methods are usually not applied to many problems as they are highly reliant on knowing the initial values for them to work and in most cases all this information is not known. However for this transfer case the initial values were well defined as each path started from a known point within the solution space making this an acceptable approach. To simplify the algorithm further it was transferred to discrete time rather than continuous time.

The development of the shooting method approach called for an investigation into the gradient of the cost function in reference to control input and time to see how the parameters key to the minimisation effected the convergence of the algorithm to the desired solution. It was observed that the gradient of the cost function relating to control inputs was significantly greater than that for time which resulted in the algorithm converging to locally optimal solutions instead of the globally optimal solution as desired. This discovery meant that the algorithm could not solve for both time and control inputs at the same time as this would result in local solutions.

To overcome this issue the cost function was split into two minimising actions; an inner minimisation relating to the control inputs for a fixed time and an outer minimisation relating to time. Using this split optimiser the gradient disparity was removed, however the change to a split optimiser made some natural cost function choices invalid for this application. Splitting the minimising action introduced a new constraint on separation distance between initial and final flight path; dependant on the scale of the cost function; significantly changing algorithm results with correct solutions being disregarded if the wrong form of the cost function was used. Different forms of the cost function are considered showing the creation of this new constraint for certain forms, allowing for a valid cost function form to be selected.

The final stage of the development of this algorithm was validation. Constraint conditions were checked against simulation values showing that the algorithm correctly activated the constraints and could select between valid and non valid paths. A check on cost function value was also performed identifying the shape of the cost function and that the minimum value was being correctly selected within the set tolerances.

The second aim of this work was to investigate the benefit of the developed algorithm by comparing its results against the those from other equivalent path planners. Two algorithms were identified for this purpose, the Dubin's Car approach and a receding horizon approach. These were selected as they are commonly used to solve similar path planning problems.

Due to the simplicity of the Dubin's approach a performance increase was expected, however the observed increase was significantly higher than anticipated with some shooting paths being almost half the length of the equivalent Dubin's path. The increase in available path curvatures in the shooting algorithm yields the extra path flexibility needed to generate paths that have significant performance improvements over the Dubin's method.

The receding horizon approach had two parameters that were key to the performance and accuracy of this algorithm, horizon length and number of horizon nodes. It was shown that there exists a balance between these two parameters giving a trade off in runtime and accuracy with the goal of keeping the number of nodes sufficiently large within a long enough horizon length that path accuracy is maximised while the runtime is minimised. There exists many combinations of horizon length, parametrised by horizon step size, and number of horizon nodes that yield suitable paths.

To perform the comparison cost data for numerous parameter combinations was obtained and the minimum value compared against the shooting results. Again in this case the shooting method results were expected to show a performance increase as the shooting algorithm had greater path flexibility. As expected this was the case showing the benefit of the new method in generating optimal paths.

In addition to this analysis the parameter combination that yielded the minimum cost values were noted. As the accuracy of the algorithm is highly susceptible to these values they must be carefully chosen or the generated paths could be significantly different to optimal solutions. Using the values a better first guess for these parameters is provided which would result in a receding horizon algorithm for this type of transfer to be more accurate.

The final aim of this work was to investigate the performance of the algorithm for the target tracking mission against an algorithm that provides greater path flexibility tailored to target tracking. To this end a Switching Cost function was created. The switching function was designed for the target tracking example to select the most appropriate cost function depending on target location and velocity. The switching function resulted in three cost functions; minimum time, minimum distance and minimum speed; with switching conditions to select the appropriate function based on separation distance between UAV and tracking path centre and relative UAV/target flight/movement path angle. The development of the switcher provides an algorithm that uses mission information to intelligently produce its paths and is a good comparison against the shooting algorithm.

To identify the performance of both algorithms the important mission parameters were extracted. For the case of target tracking these parameters were key visibility indices. The first was that of target to UAV visibility or the probability of the UAV being able to view the target. The second was that of UAV to target visibility or the probability of the target viewing the UAV. The desired goal of any target tracking mission is to track the target so the visibility of target to UAV should be high, conversely it is undesirable for the UAV to be visible to the target so this index should be low. To investigate the effect of the path planning methods on these indices a Monte Carlo simulation was performed for randomly generated initial and terminal conditions, calculating the indices for each run. Using this method visibility data was obtained for both the shooting and switching algorithm, a comparison of the data showed no marked difference between the indices for both the path planning methods. Given that one of the switching modes was the singular cost function it is clear that during the transfer operation the UAV spends the majority of its time utilising the singular cost function to produce paths and that adding flexibility to the path planning effort by providing more cost function choice yields no significant effect on the target tracking mission being carried out further validating the developed algorithm. In addition to this it can also be seen that more paths using the single cost function are closer to the desired final tracking circle (within the same time-scale) resulting in higher visibility indexes.

6.2 Further Work

Due to the complexity and runtime of the algorithms developed throughout this work a concious effort was made to maximise the work carried out in the time available. To this end some avenues of algorithm development were left unexplored to prioritise the main themes of the research. This section details potential areas of further work using the algorithms developed as part of this thesis.

6.2.1 Hamiltonian

The work using the two point boundary value approach with the Hamiltonian could not be revisited. The better understanding of the solution space provided by the shooting method algorithm could be used to rework the TPBV problem so that it yields usable results. The benefit of doing so would be the provision of a second algorithm that could carry out transfer path planning for target tracking. A TPBV approach could have a shorter and more efficient runtime than the shooting method algorithm if the initial guess was better defined. A comparison between the two algorithms could be carried out to further validate the results in this work.

6.2.2 Shooting Method Algorithm

The shooting method algorithm was developed using a 2D solution space. This was done to reduce the complexity of the problem by setting altitude as constant with a value large enough to avoid collision with ground obstacles. A potential extension to the algorithm would be to allow a varying altitude based on external information about ground obstacle height. The effect of changing to a 3D solution space is currently unknown but could allow for paths that are shorter than their 2D counterparts.

The goal of the shooting method algorithm was to design a guidance law, not to apply a specific guidance law to a particular model, hence providing path information that could be used by any UAV. As a result it was designed using a very simple model representing the UAV as a point mass. A valid and necessary extension of this work is the design of a control law for specific UAVs. This extension is essential if the path information generated in this work is to be put to practical use. The constraint equations would need to be tweaked for specific UAVs so that the correct velocity and control constraints are used with the resulting path information forming the basis of a control law for a specific and detailed UAV model. The coupling of path planner and controller would allow a specific UAV to generate then follow the generated transfer path.

6.2.3 Result Comparison

The shooting method has so far been compared to only two other path planners, which were chosen due to their similarity to the shooting method. Other applicable path planning algorithms could be investigated to further investigate the benefit of the developed shooting algorithm. In addition to this extra validation exercise comparing the results of this optimal algorithm against others could lead to improved algorithm development as a baseline result for this type of transfer is now available.

6.2.4 Switching Cost Function Applications

The switching cost function method was created to provide a comparison between the shooting algorithm and one which is capable of using mission information to select the applicable transfer cost function. The switching method requires further investigation to analyse the potential benefits (if any) over other optimal control techniques. Although showing no significant improvement over a single cost function example in this case, further tests are required. Testing in this work applied the switching method to only one mission and as such it is not fair to dismiss the technique as a viable path planning method. Further work in this could be therefore wide ranging.

The switching conditions alone could be a large area of study as they are a tunable parameter. An interesting extension of this work would be to develop an algorithm that could alter the switching conditions potentially providing control over the effect on the mission parameters. In effect this is adding sensory intelligence to the UAV as it monitors external conditions and adapts its path planning methods accordingly.

Another area of study would be to the application of this method to other algorithms and missions. Only a specific target tracking mission was investigated and simply because no improvement between path planning methods was shown here it does not mean that this would be the case for other mission types. To further work in this area, application of the switching method to other missions must be carried out. This analysis would provide a better understanding of the effects of utilising a switching method approach.

Appendix A

Plots of the Single Optimiser Gradient Issue

This appendix contains extra figures showing the single optimiser gradient issue shown in 3.4.2. This appendix does not include any figure already shown in the main work and only shows exit angles up to 15°as this is where this particular algorithm fails to produce valid results.

A.1 Gradient comparison for -75°

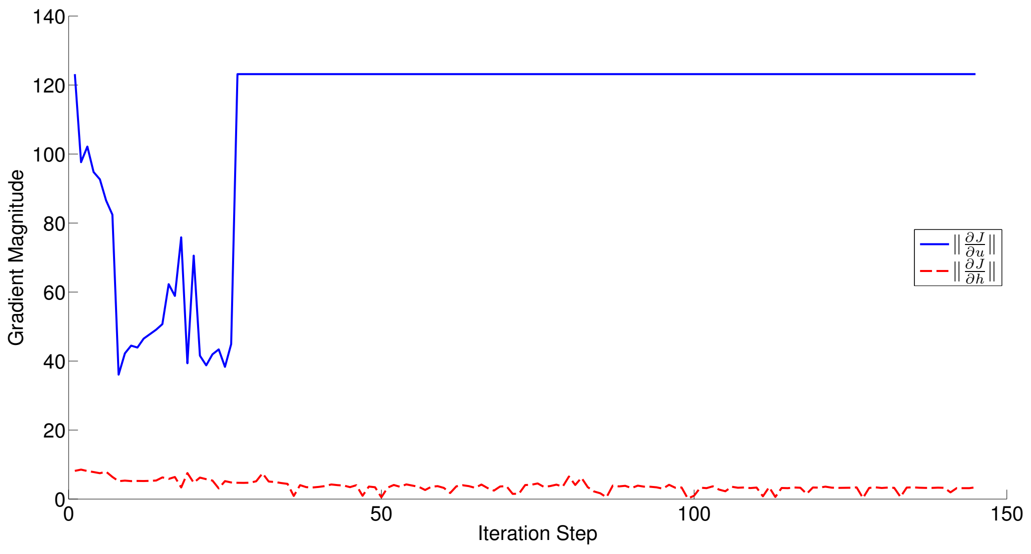

FIGURE A.1: Gradient comparison for -75°

A.2 Gradient comparison for -60°

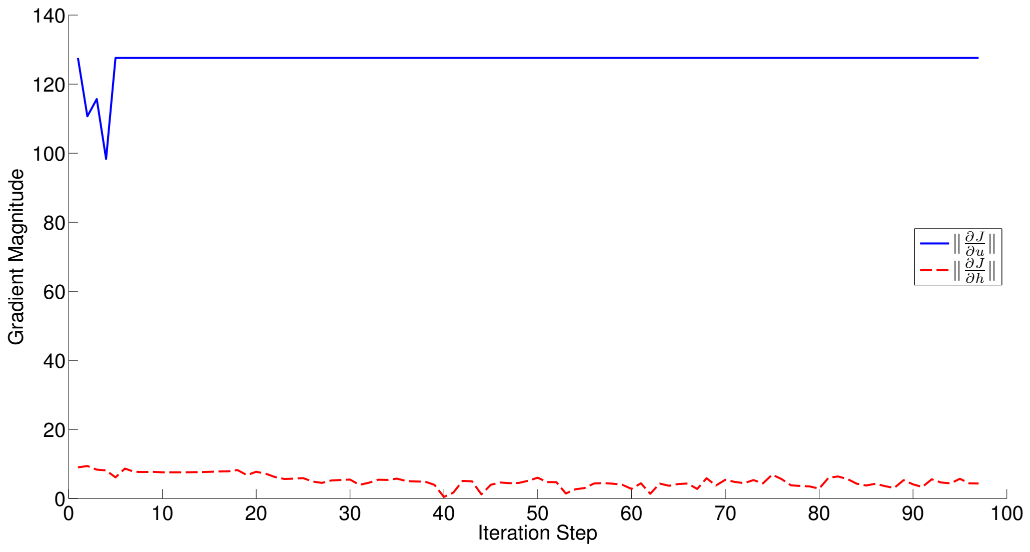

Figure A.2: Gradient comparison for -60°

A.3 Gradient comparison for -45°

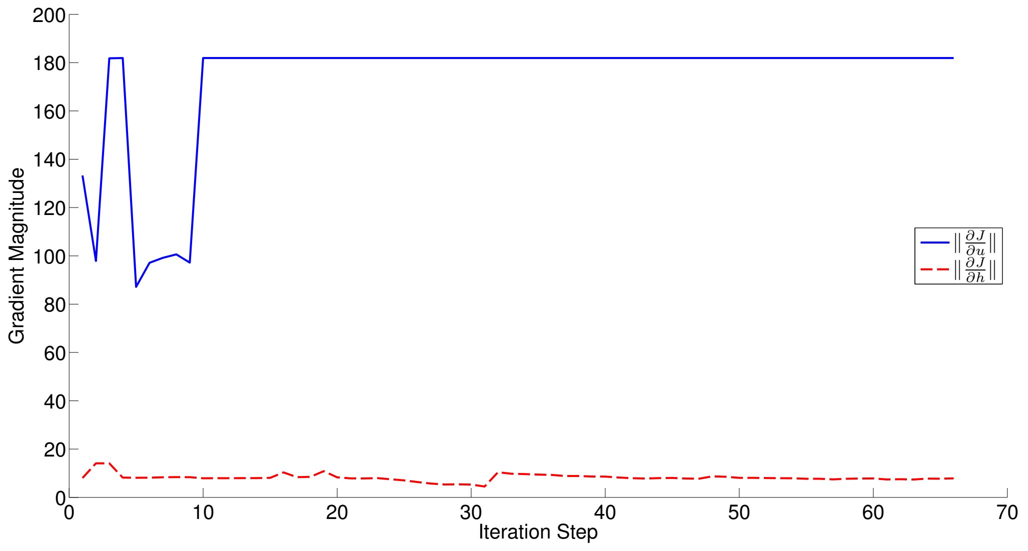

Figure A.3: Gradient comparison for -45°

A.4 Gradient comparison for -30°

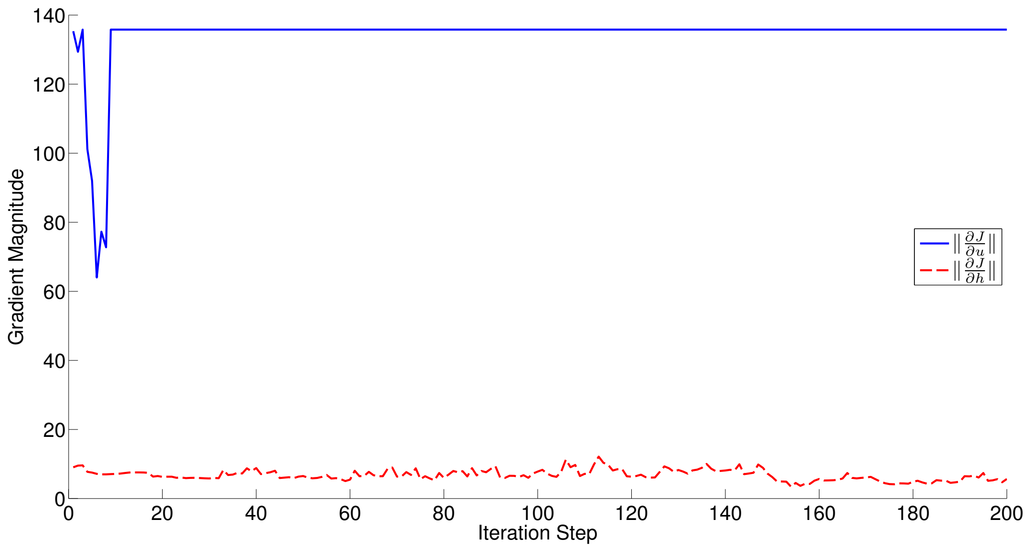

FIGURE A.4: Gradient comparison for -30°

A.5 Gradient comparison for -15°

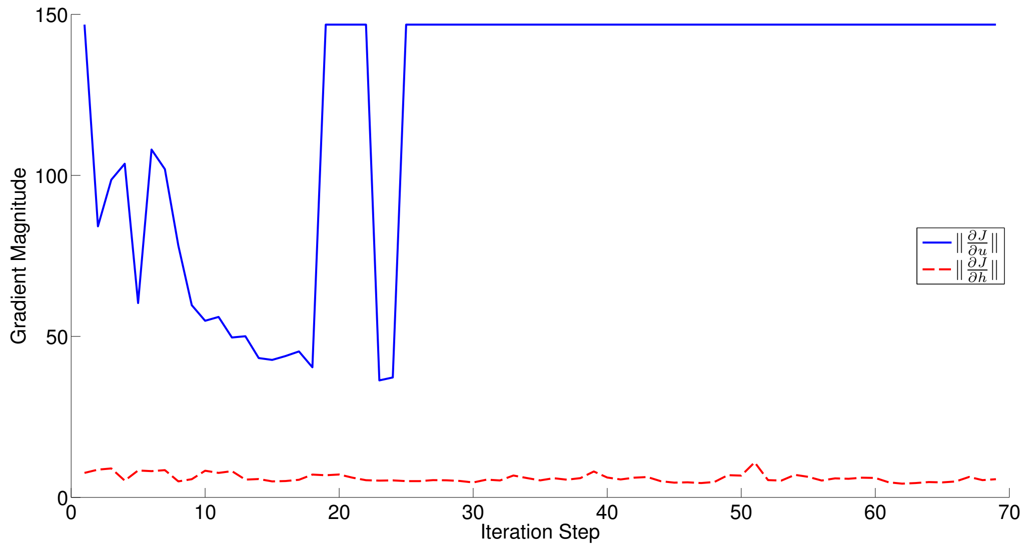

FIGURE A.5: Gradient comparison for -15°

A.6 Gradient comparison for 0°

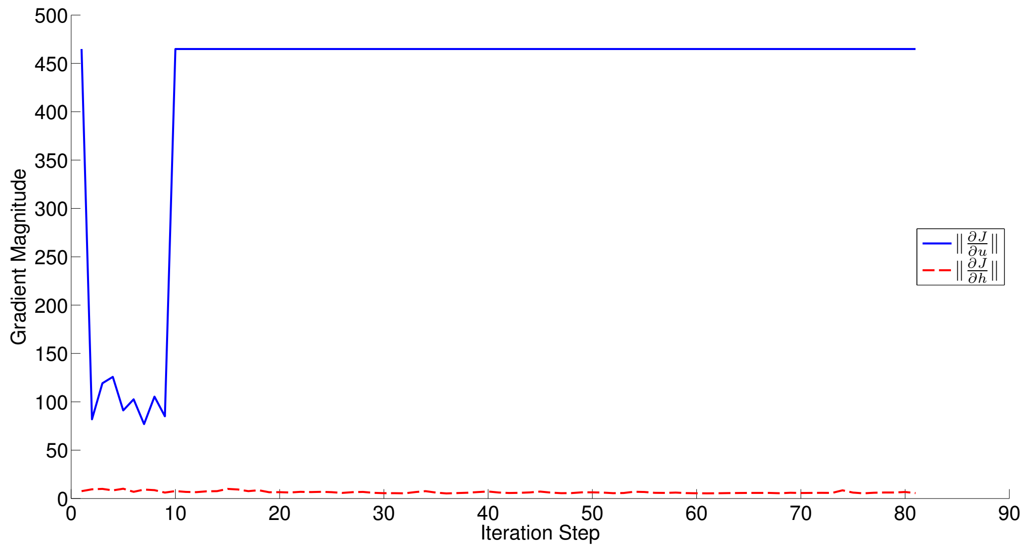

FIGURE A.6: Gradient comparison for 0°

A.7 Gradient comparison for 15°

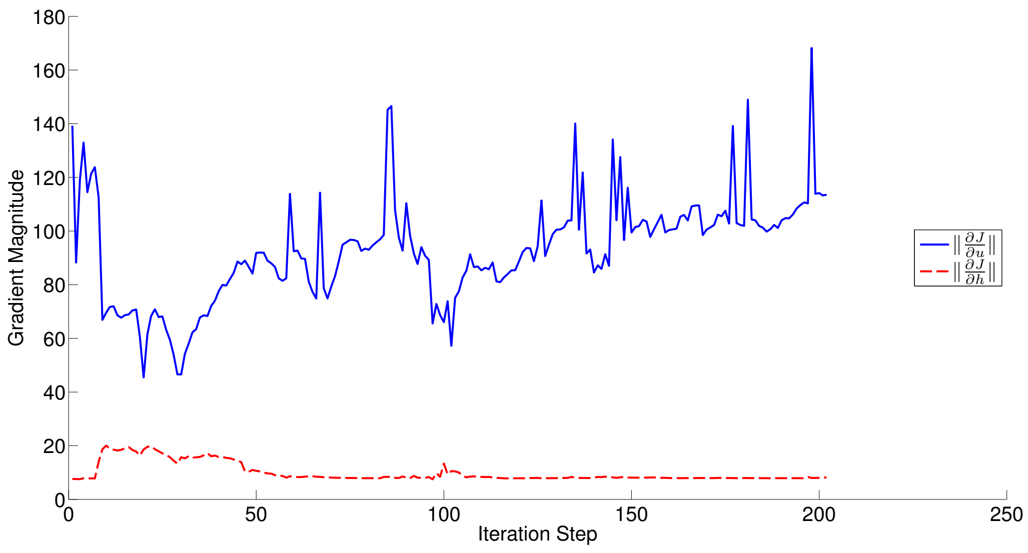

FIGURE A.7: Gradient comparison for 15°

Appendix B

Plots of Shooting Method Algorithm Tests

This appendix contains extra figures for the testing procedure documented in Section 3.4.4. This appendix only shows uniquely different paths as described in Section 4.2 and does not include any figure already shown in the main work.

B.1 Figures for an exit angle of -75°

B.1.1 Path Map

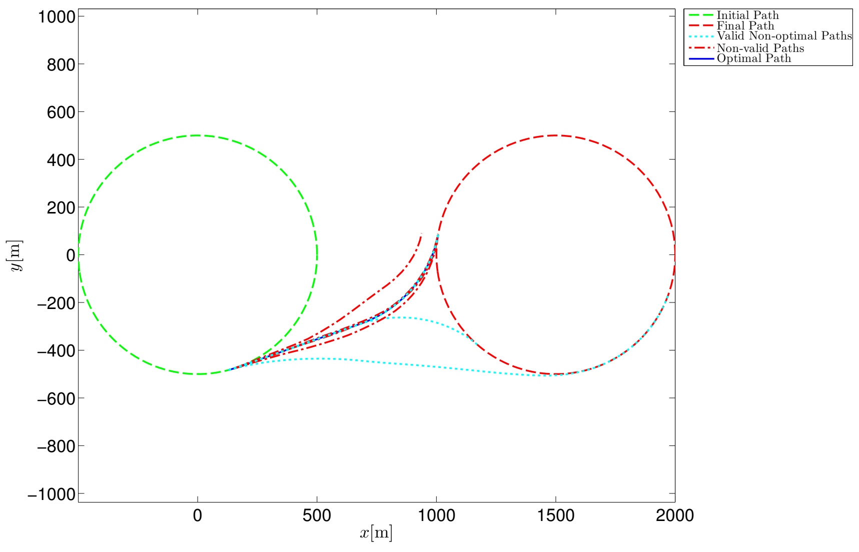

FIGURE B.1: All paths found for an exit angle of -75°

B.1.2 Velocity Magnitude Profiles

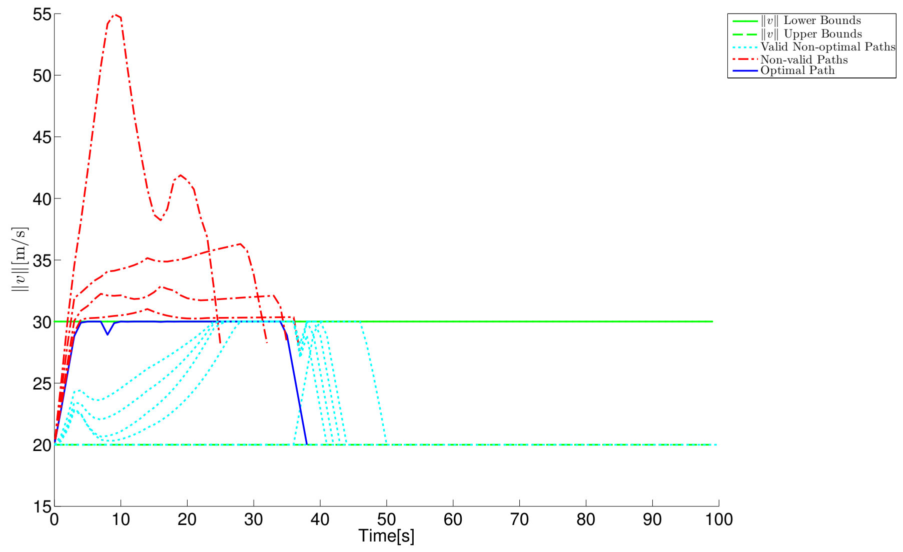

FIGURE B.2: The velocity profiles for all paths found for an exit angle of -75°

B.1.3 Control History Profiles

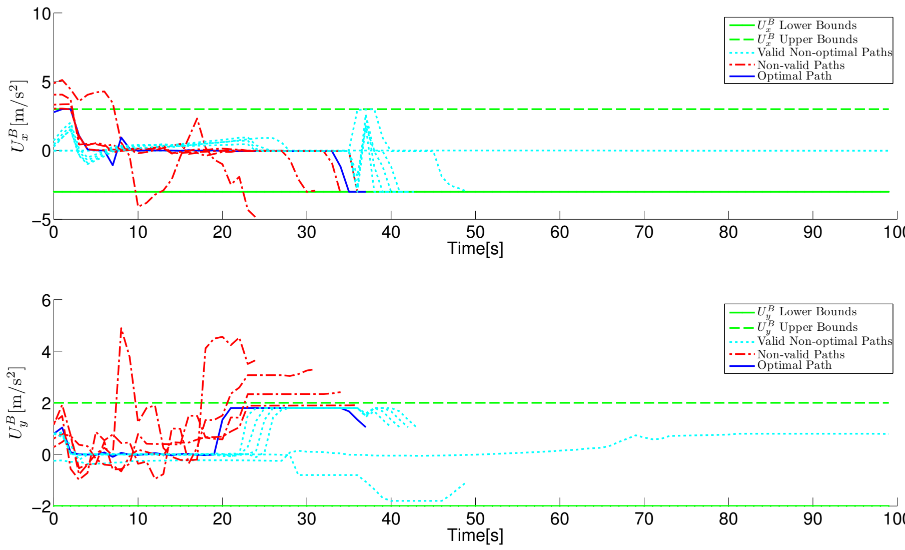

FIGURE B.3: The U_x and U_y control history for all paths found for an exit angle of -75°

B.1.4 Path Curvature Conditions

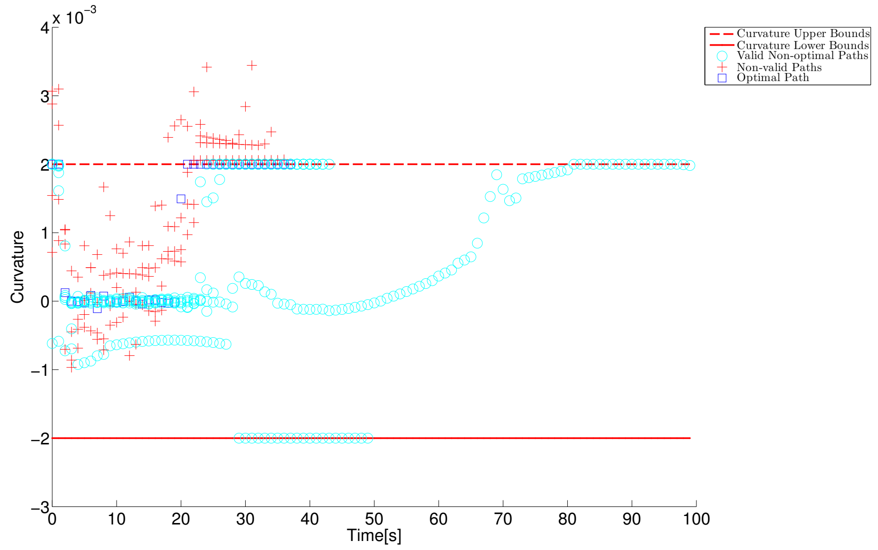

Figure B.4: Path curvatures for all paths found for an exit angle of -75°

B.1.5 Path Cost Function Curve

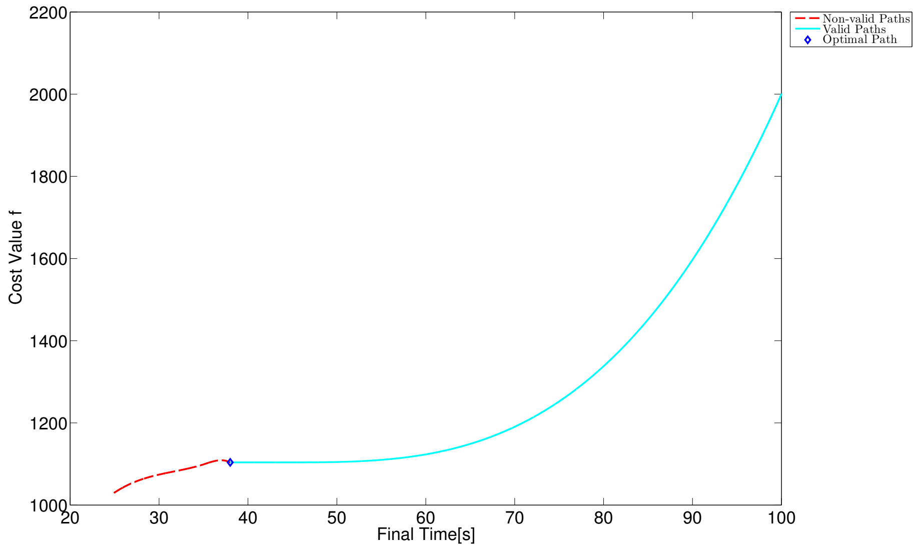

Figure B.5: Cost curve for an exit angle of -75°

B.2 Figures for an exit angle of -60°

B.2.1 Path Map

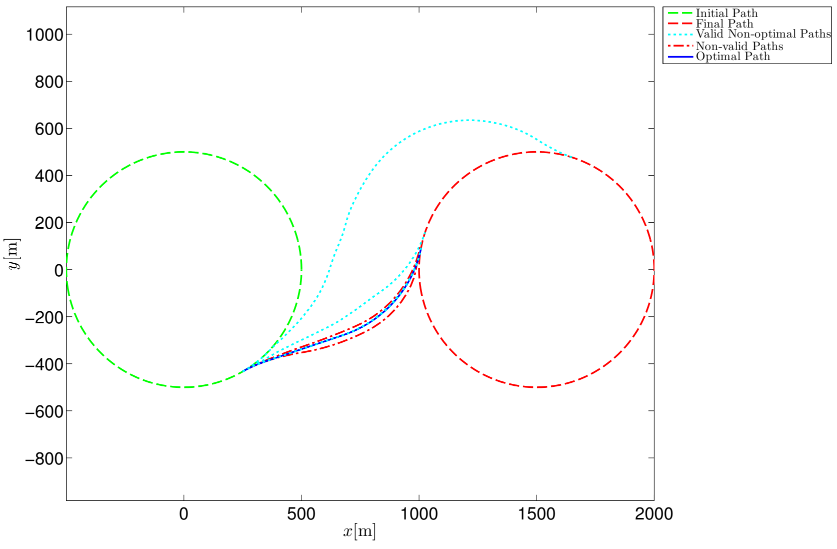

FIGURE B.6: All paths found for an exit angle of -60°

B.2.2 Velocity Magnitude Profiles

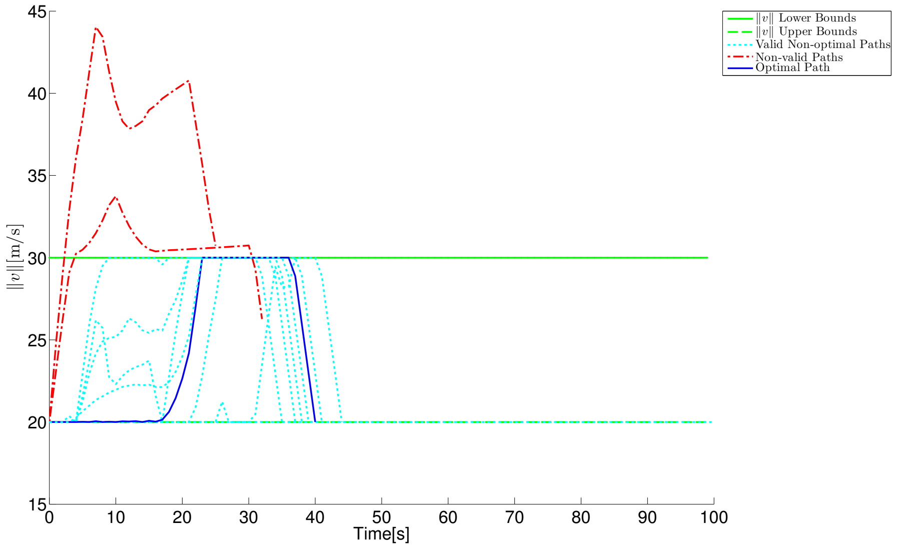

FIGURE B.7: The velocity profiles for all paths found for an exit angle of -60°

B.2.3 Control History Profiles

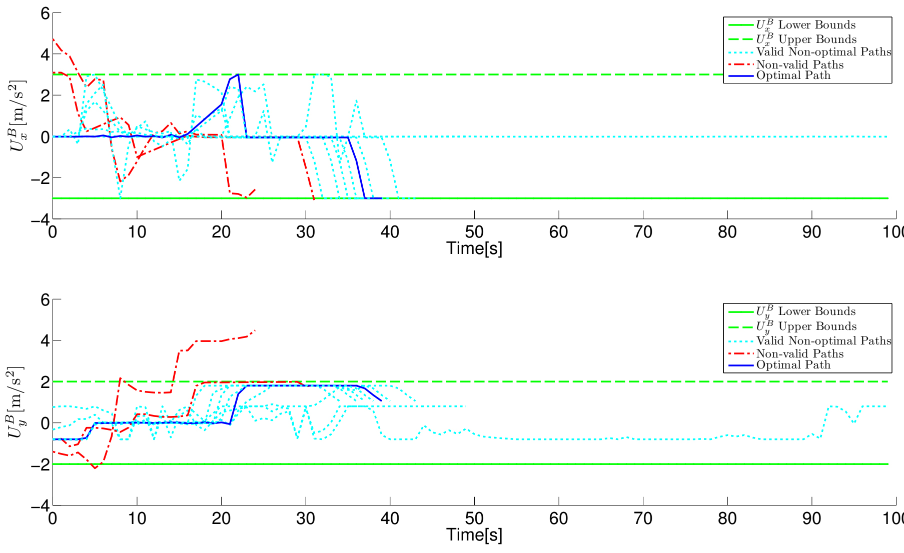

FIGURE B.8: The U_x and U_y control history for all paths found for an exit angle of -60°

B.2.4 Path Curvature Conditions

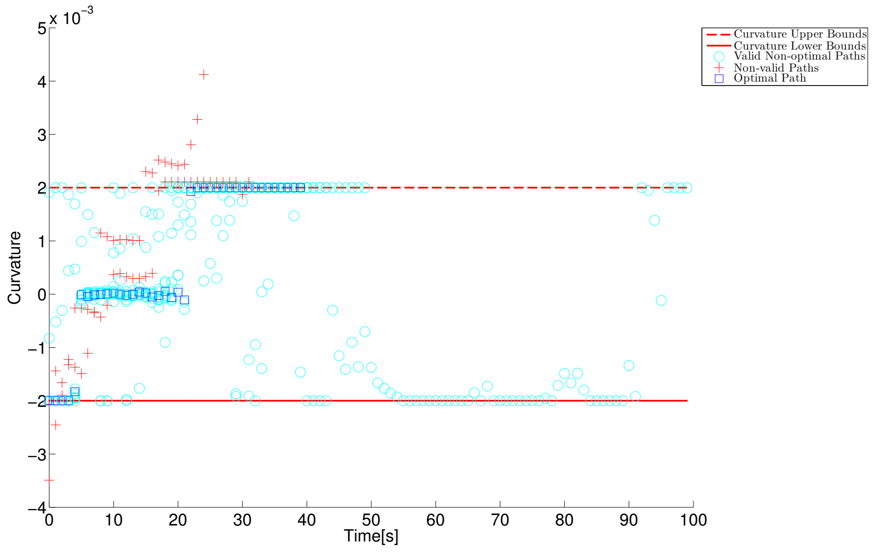

FIGURE B.9: Path curvatures for all paths found for an exit angle of -60°

B.2.5 Path Cost Function Curve

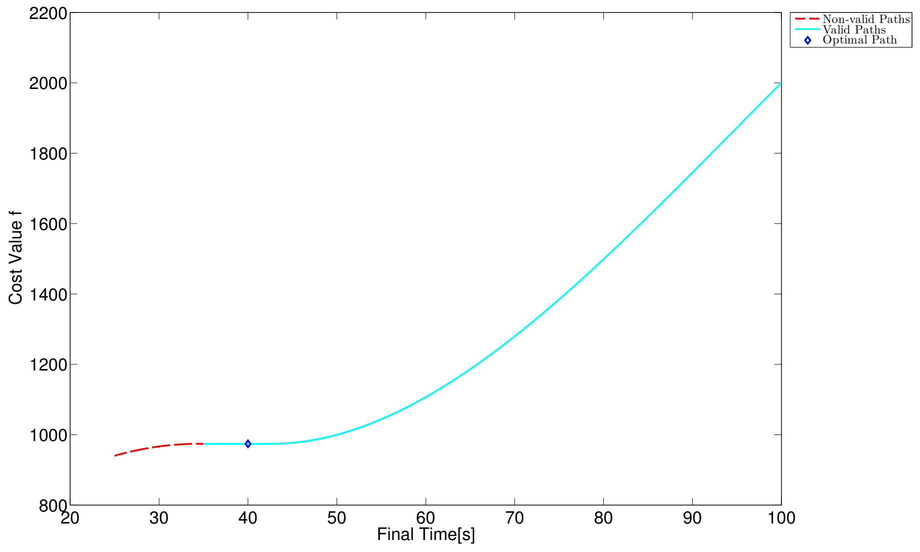

FIGURE B.10: Cost curve for an exit angle of -60°

B.3 Figures for an exit angle of -45°

B.3.1 Path Map

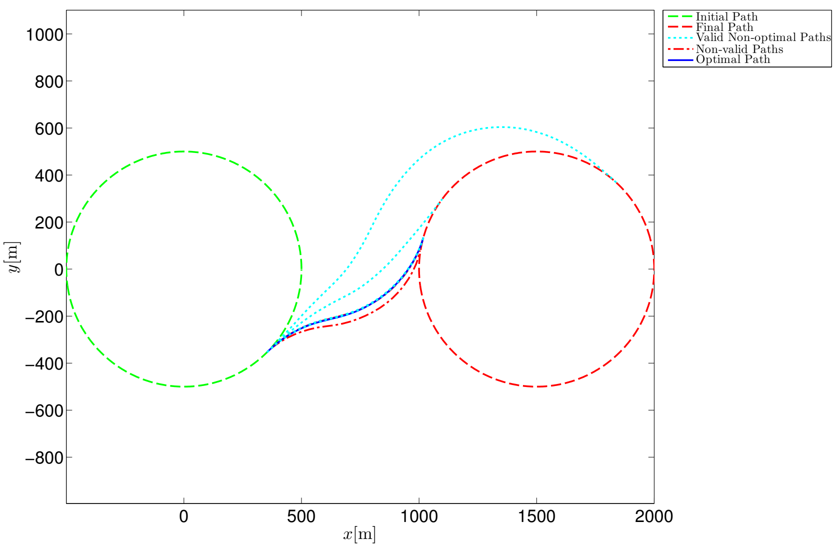

FIGURE B.11: All paths found for an exit angle of -45°

B.3.2 Velocity Magnitude Profiles

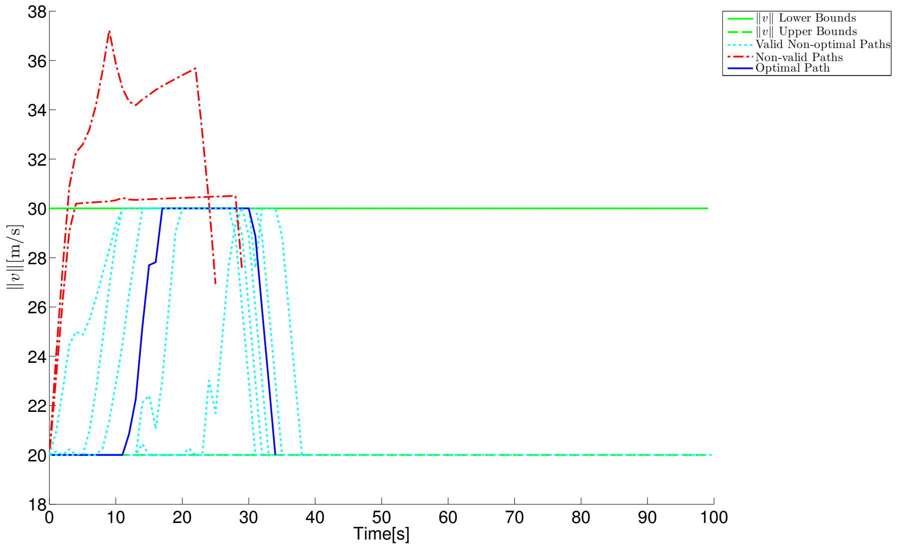

FIGURE B.12: The velocity profiles for all paths found for an exit angle of -45°

B.3.3 Control History Profiles

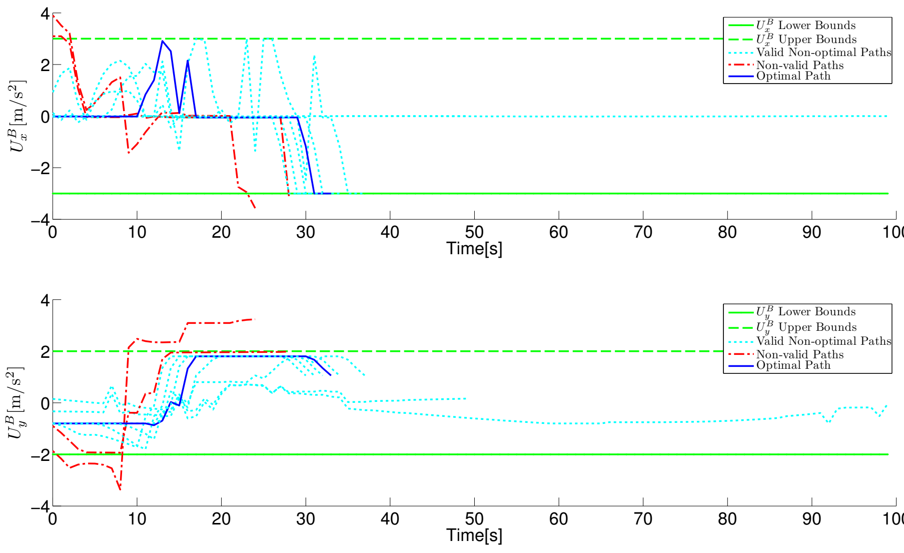

FIGURE B.13: The U_x and U_y control history for all paths found for an exit angle of -45°

B.3.4 Path Curvature Conditions

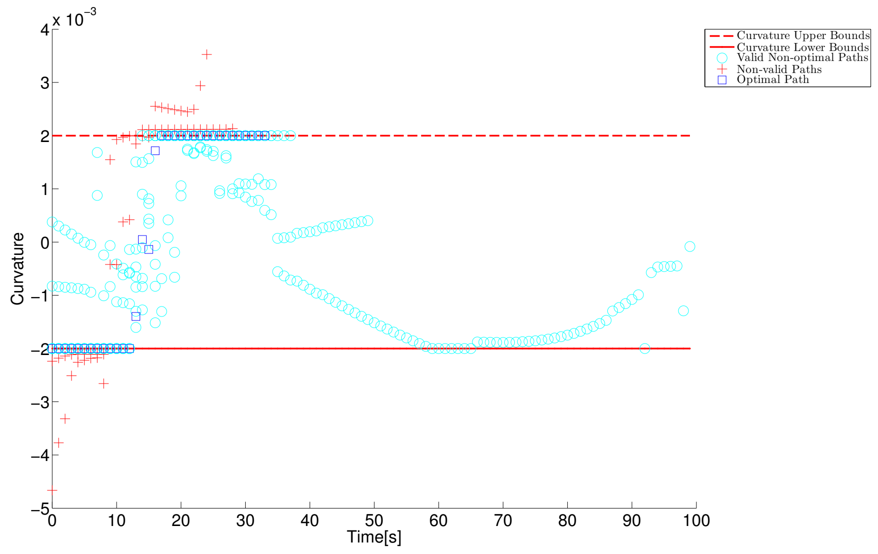

FIGURE B.14: Path curvatures for all paths found for an exit angle of -45°

B.3.5 Path Cost Function Curve

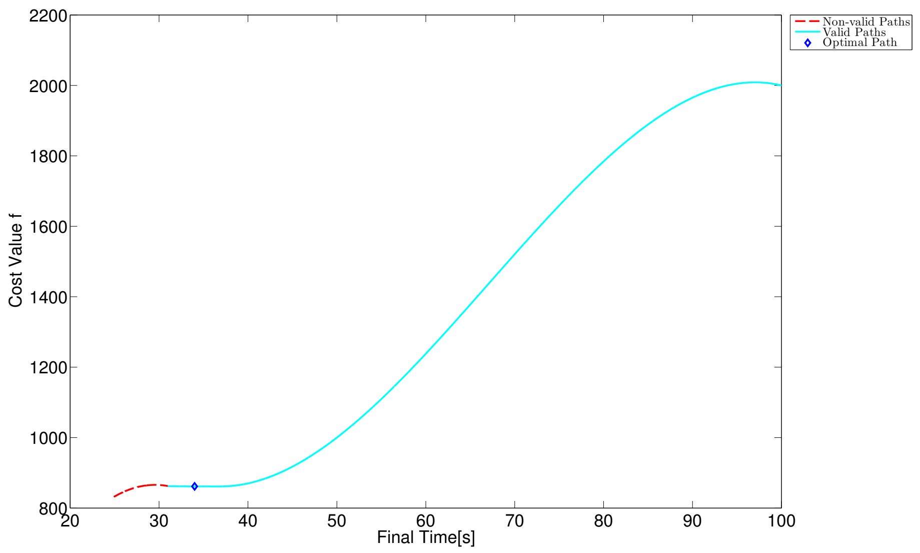

FIGURE B.15: Cost curve for an exit angle of -45°

B.4 Figures for an exit angle of -30°

B.4.1 Path Map

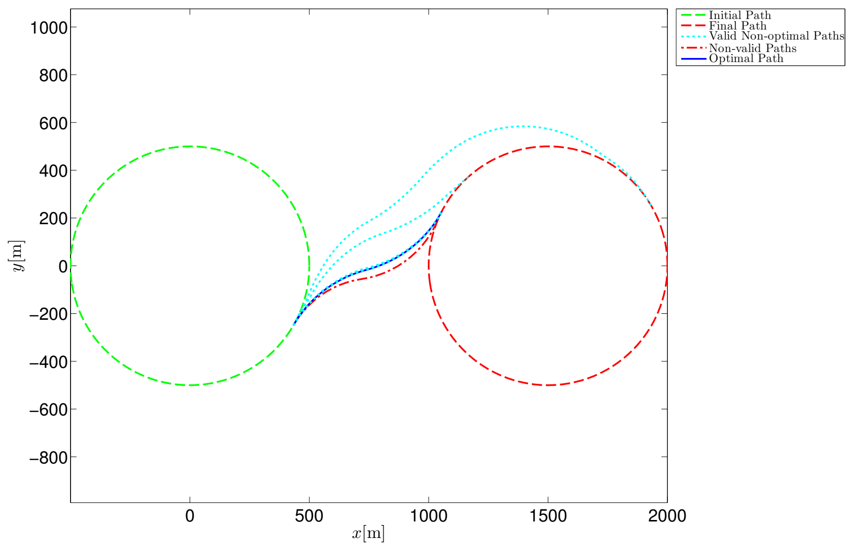

FIGURE B.16: All paths found for an exit angle of -30°

B.4.2 Velocity Magnitude Profiles

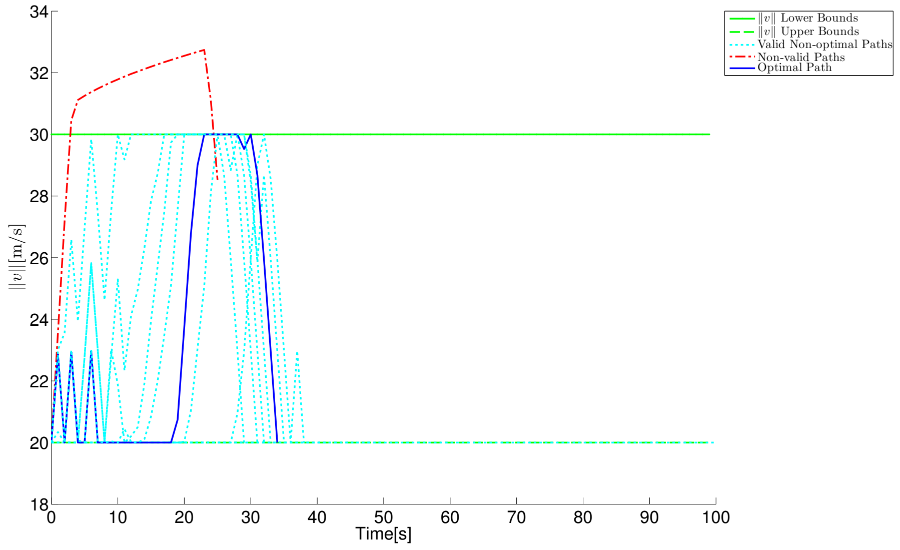

FIGURE B.17: The velocity profiles for all paths found for an exit angle of -30°

B.4.3 Control History Profiles

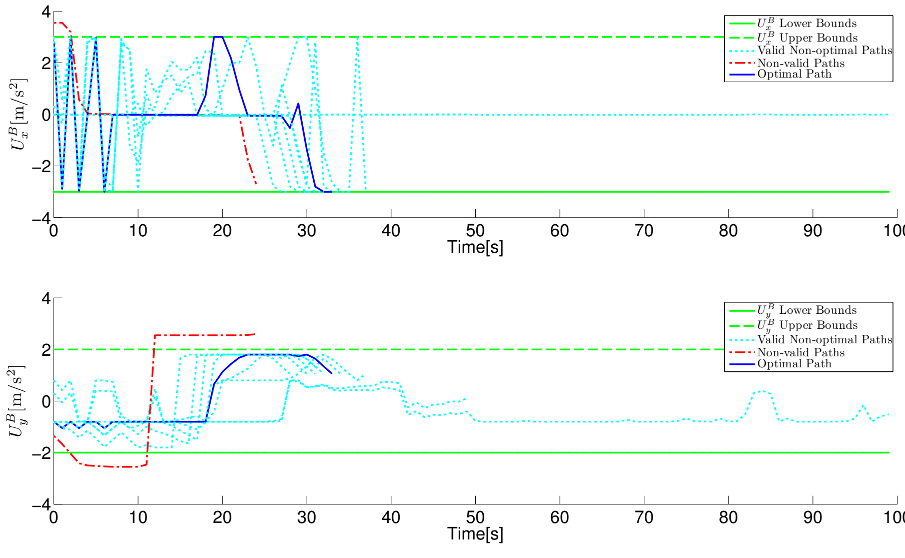

FIGURE B.18: The U_x and U_y control history for all paths found for an exit angle of -30°

B.4.4 Path Curvature Conditions

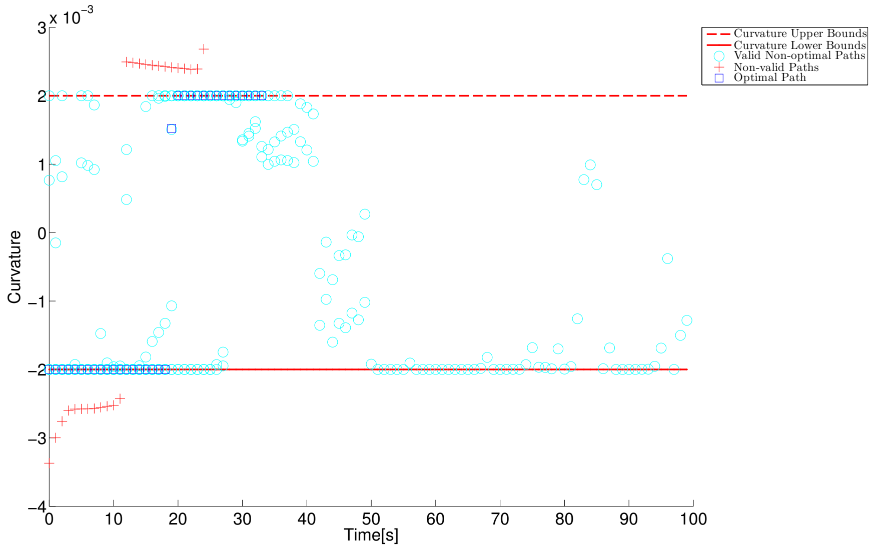

FIGURE B.19: Path curvatures for all paths found for an exit angle of -30°

B.4.5 Path Cost Function Curve

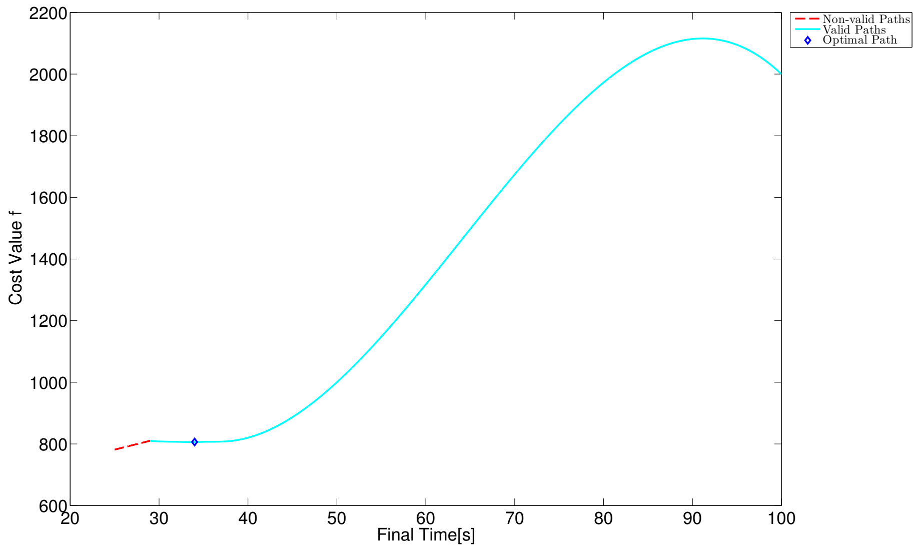

FIGURE B.20: Cost curve for an exit angle of -30°

B.5 Figures for an exit angle of -15°

B.5.1 Path Map

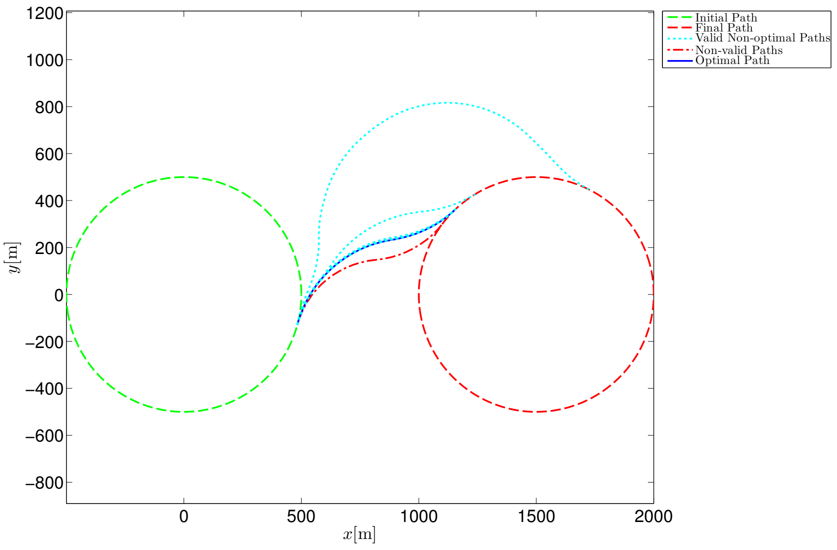

FIGURE B.21: All paths found for an exit angle of -15°

B.5.2 Velocity Magnitude Profiles

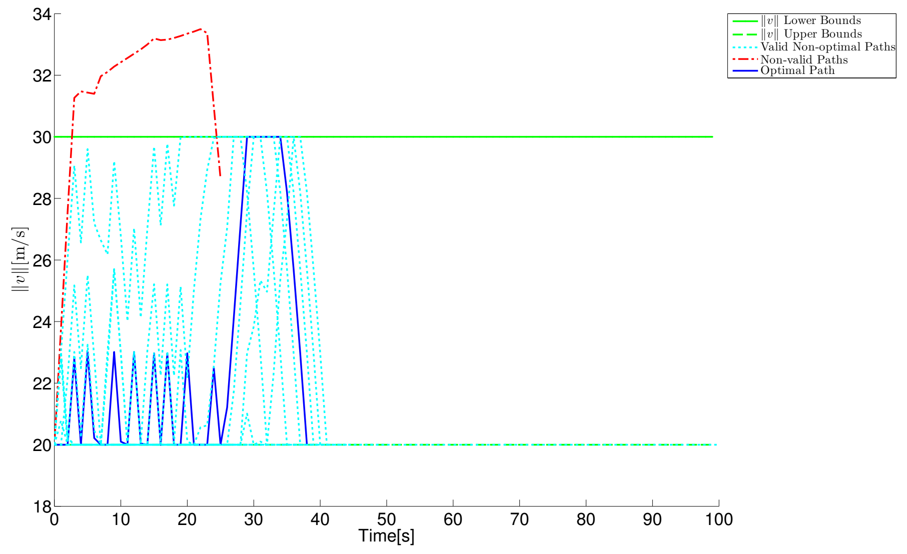

Figure B.22: The velocity profiles for all paths found for an exit angle of -15°

B.5.3 Control History Profiles

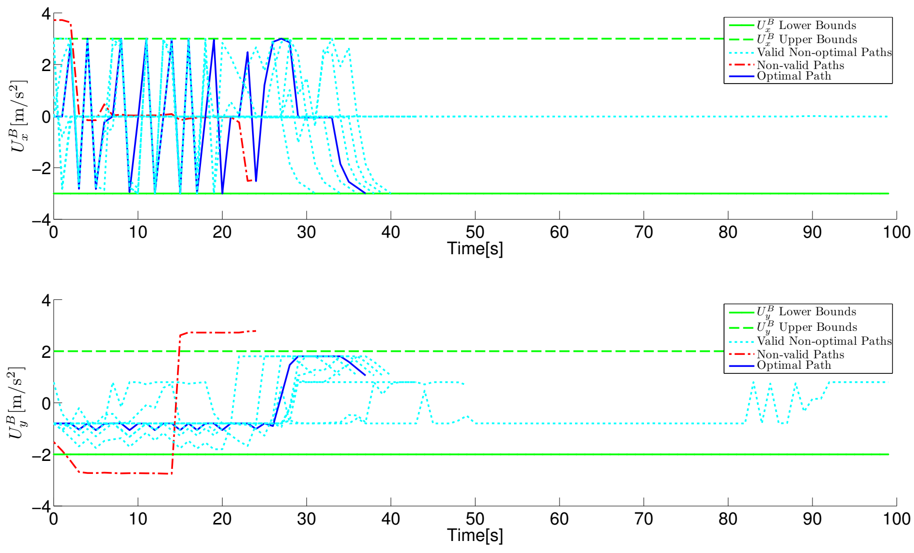

Figure B.23: The U_x and U_y control history for all paths found for an exit angle of -15°

B.5.4 Path Curvature Conditions

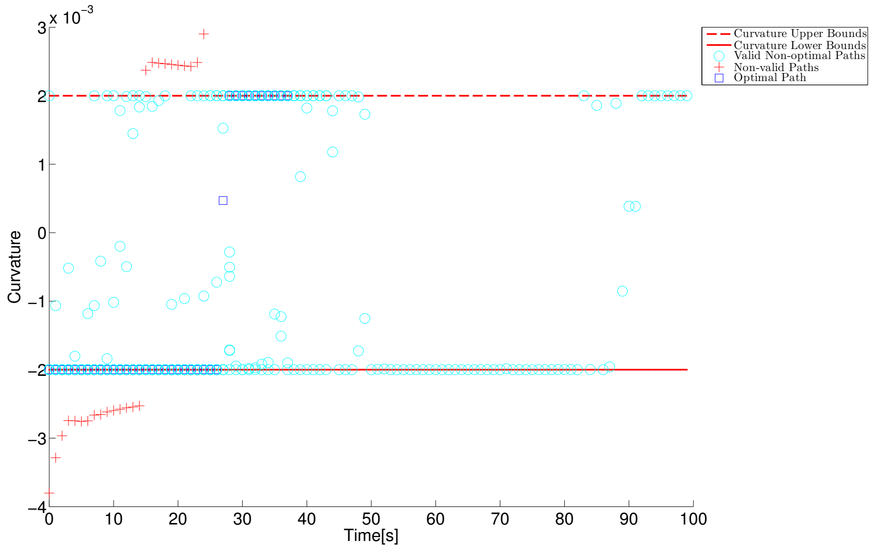

FIGURE B.24: Path curvatures for all paths found for an exit angle of -15°

B.5.5 Path Cost Function Curve

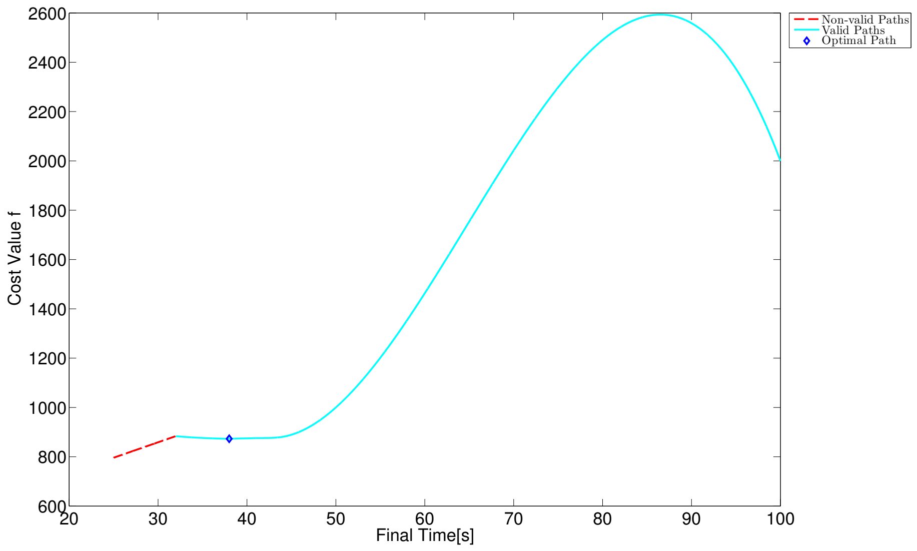

FIGURE B.25: Cost curve for an exit angle of -15°

B.6 Figures for an exit angle of 0°

B.6.1 Path Map

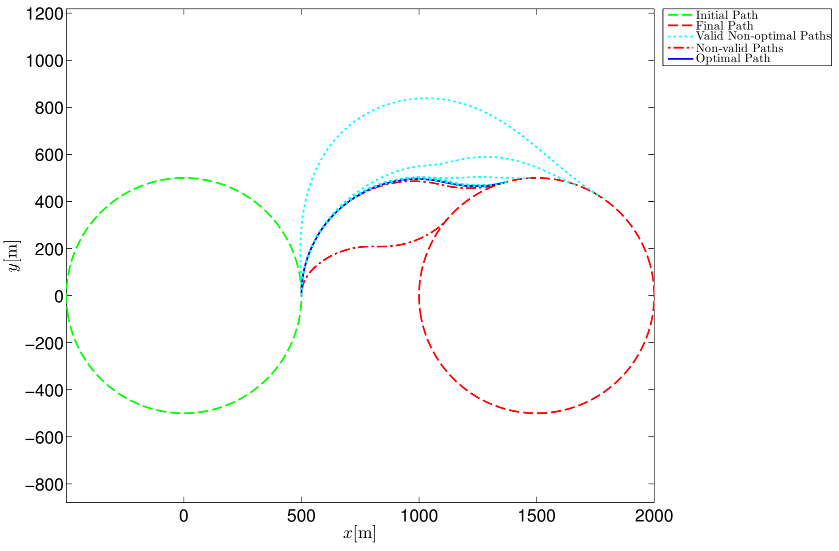

Figure B.26: All paths found for an exit angle of 0°

B.6.2 Velocity Magnitude Profiles

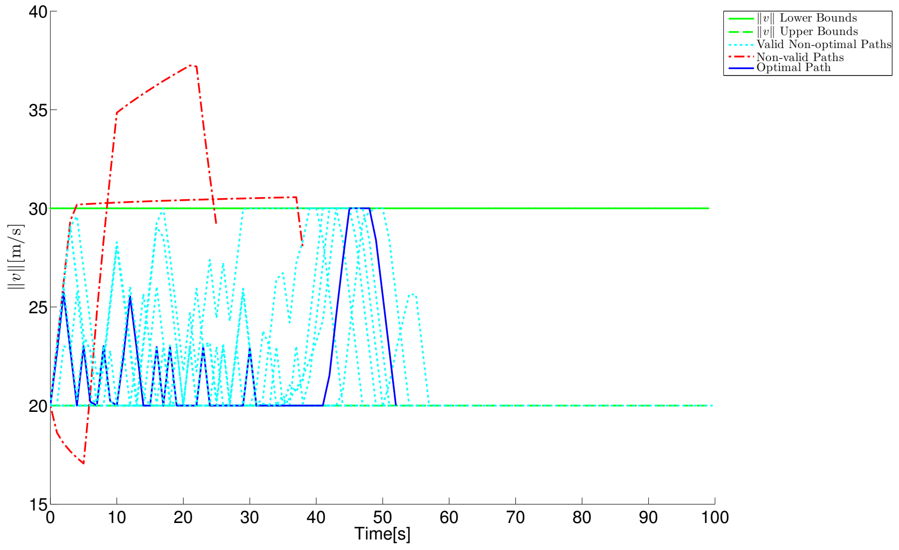

Figure B.27: The velocity profiles for all paths found for an exit angle of 0°

B.6.3 Control History Profiles

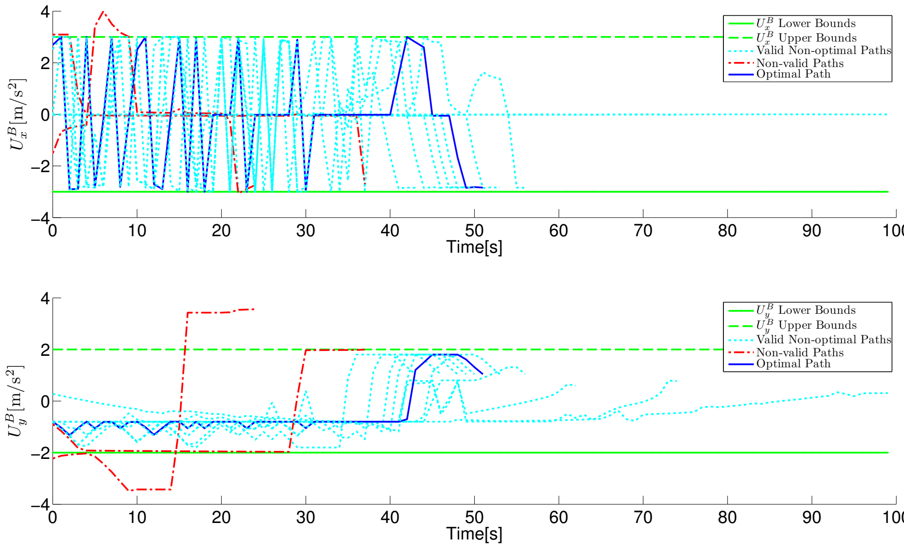

FIGURE B.28: The U_x and U_y control history for all paths found for an exit angle of $0°$

B.6.4 Path Curvature Conditions

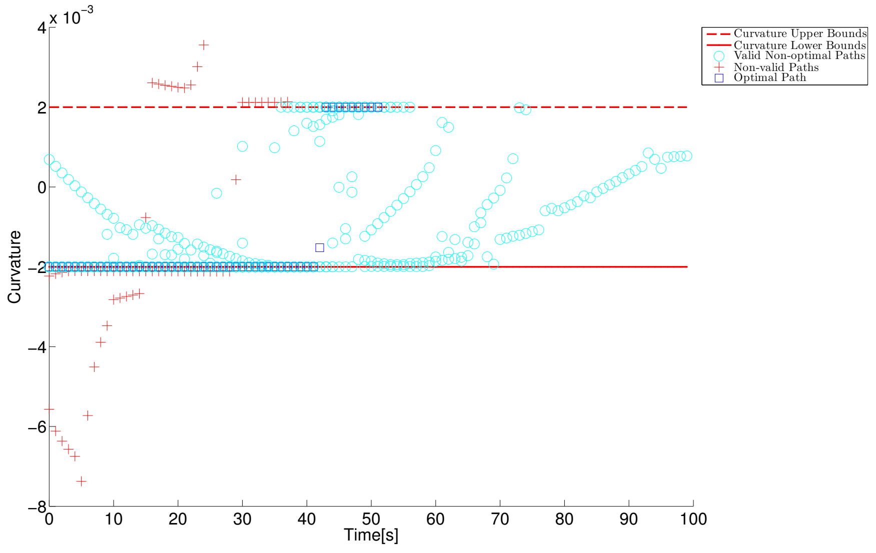

FIGURE B.29: Path curvatures for all paths found for an exit angle of $0°$

B.6.5 Path Cost Function Curve

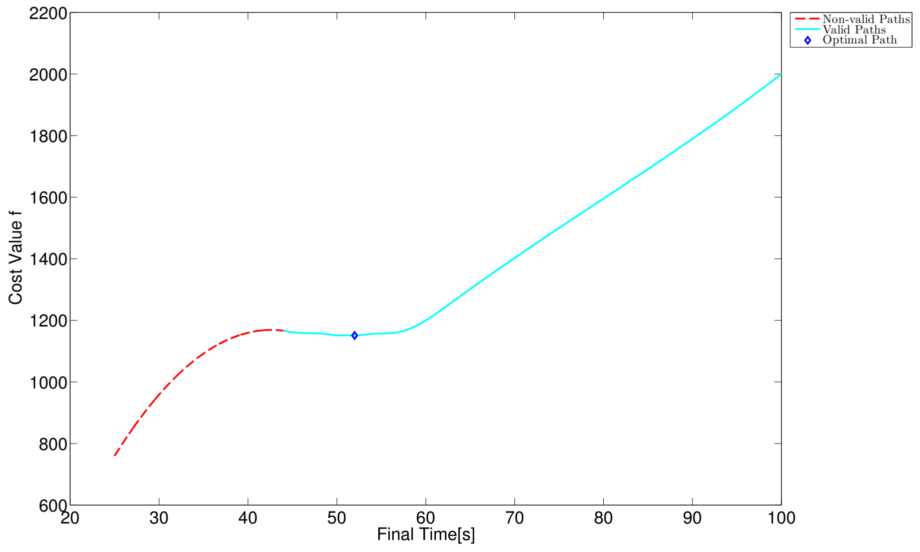

FIGURE B.30: Cost curve for an exit angle of 0°

B.7 Figures for an exit angle of 15°

B.7.1 Path Map

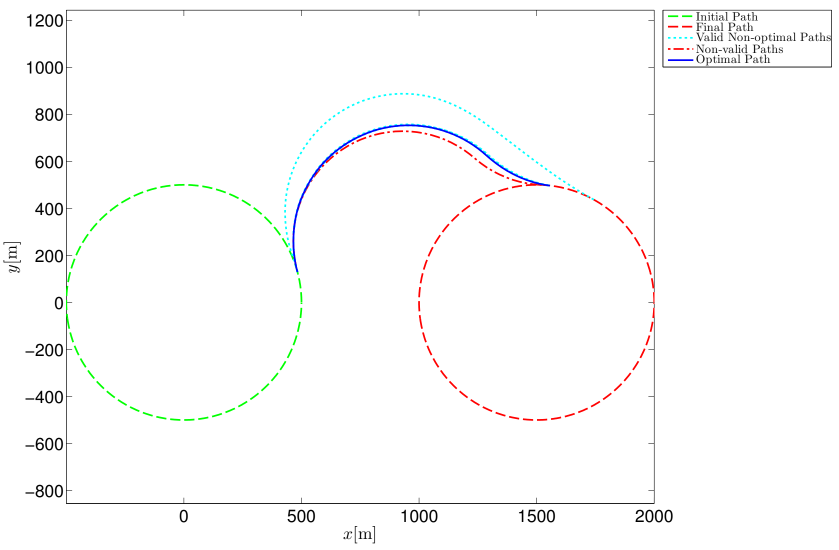

FIGURE B.31: All paths found for an exit angle of 15°

B.7.2 Velocity Magnitude Profiles

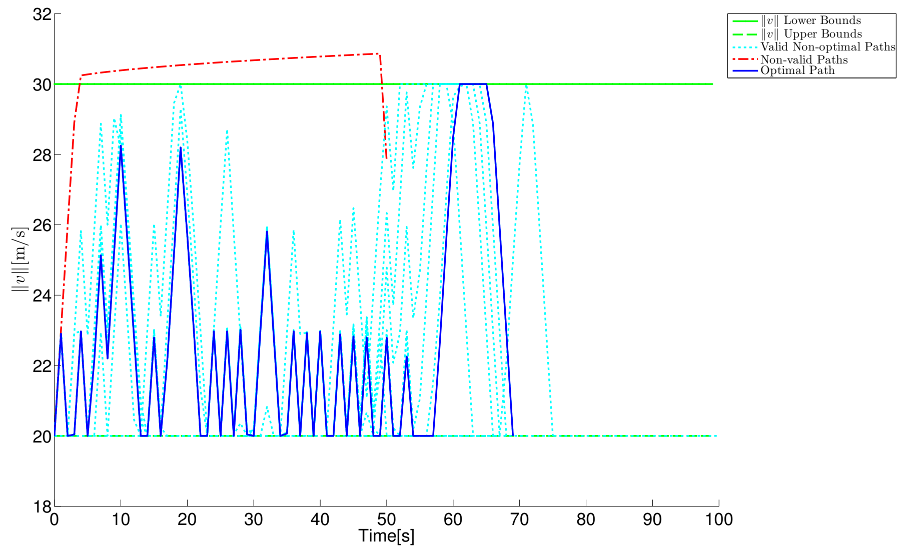

FIGURE B.32: The velocity profiles for all paths found for an exit angle of 15°

B.7.3 Control History Profiles

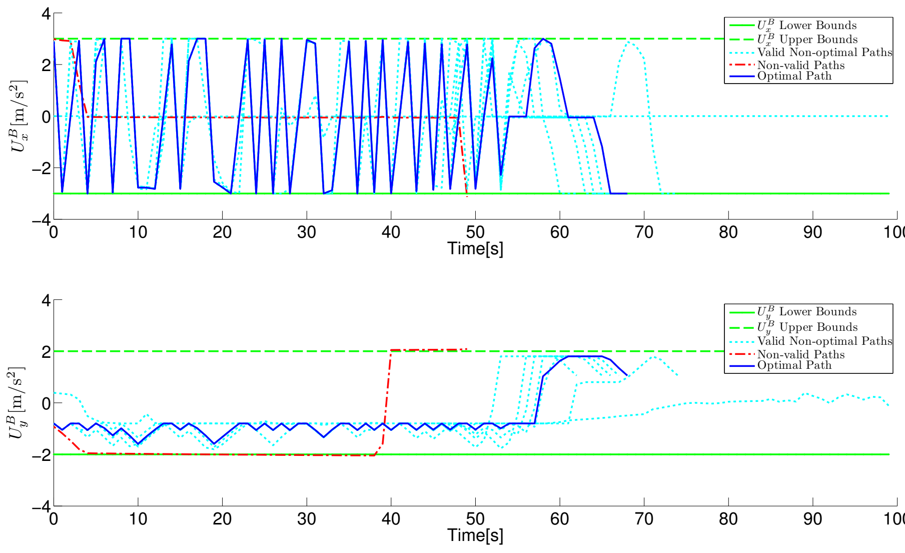

FIGURE B.33: The U_x and U_y control history for all paths found for an exit angle of 15°

B.7.4 Path Curvature Conditions

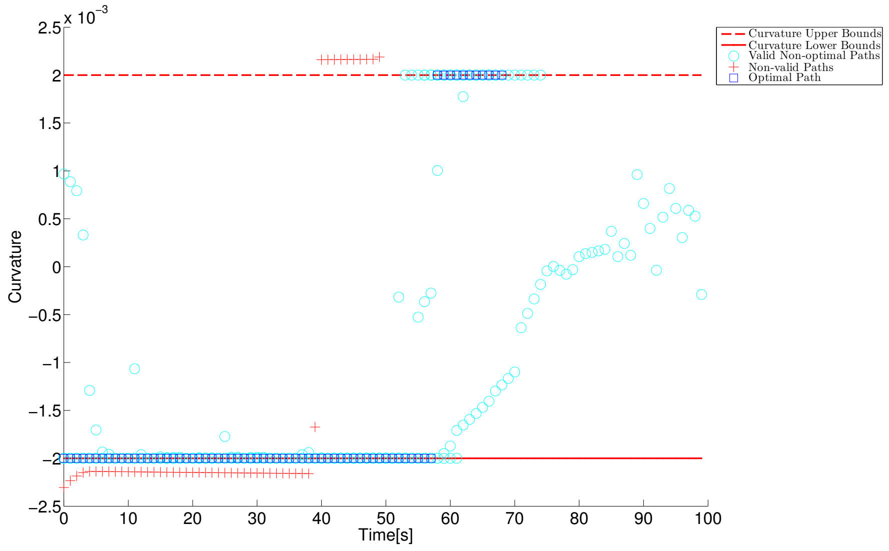

FIGURE B.34: Path curvatures for all paths found for an exit angle of 15°

B.7.5 Path Cost Function Curve

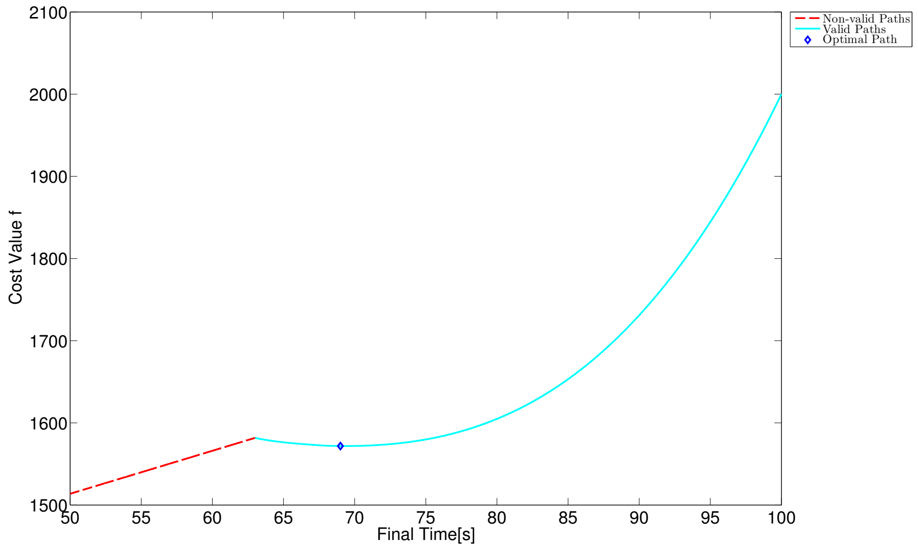

FIGURE B.35: Cost curve for an exit angle of 15°

B.8 Figures for an exit angle of 30°

B.8.1 Path Map

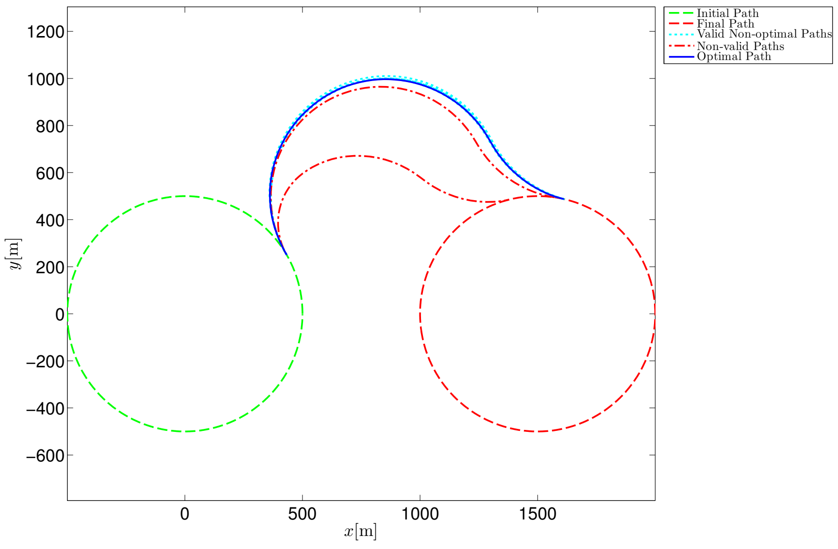

FIGURE B.36: All paths found for an exit angle of 30°

B.8.2 Velocity Magnitude Profiles

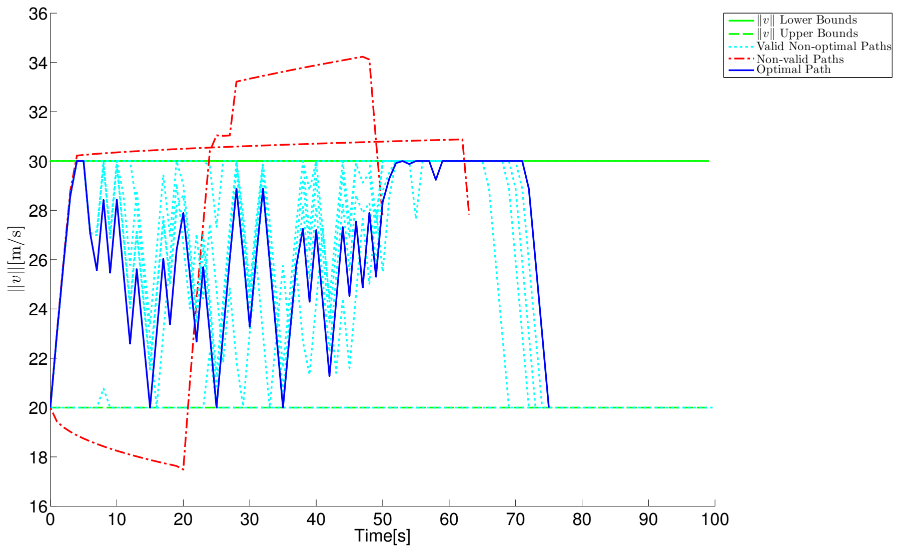

FIGURE B.37: The velocity profiles for all paths found for an exit angle of 30°

B.8.3 Control History Profiles

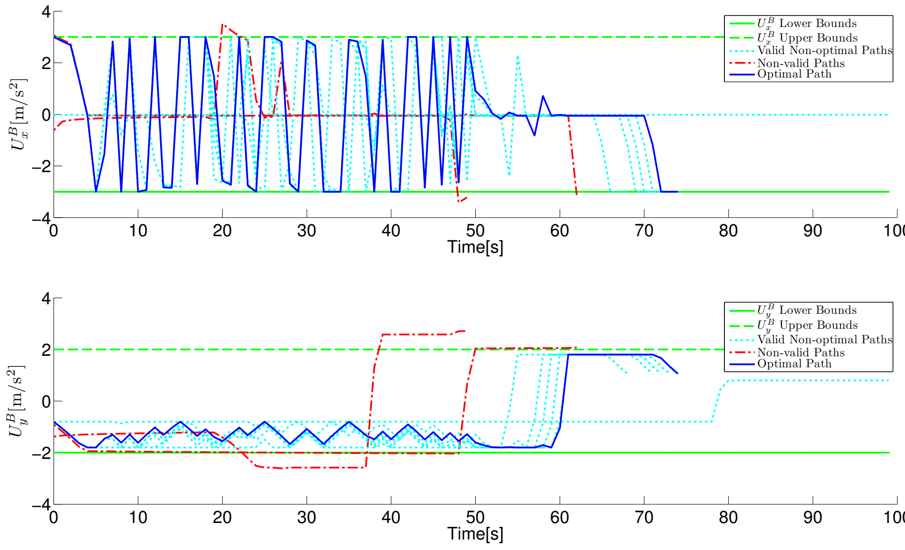

FIGURE B.38: The U_x and U_y control history for all paths found for an exit angle of 30°

B.8.4 Path Curvature Conditions

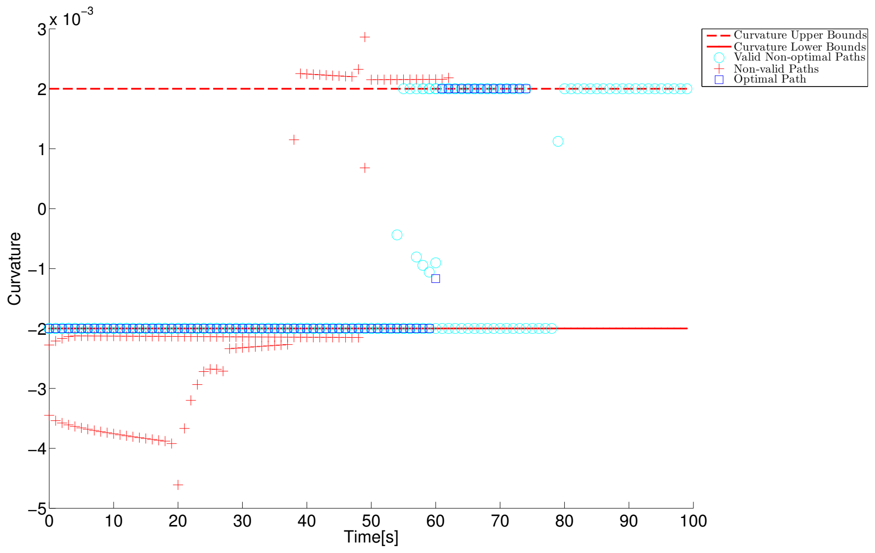

FIGURE B.39: Path curvatures for all paths found for an exit angle of 30°

B.8.5 Path Cost Function Curve

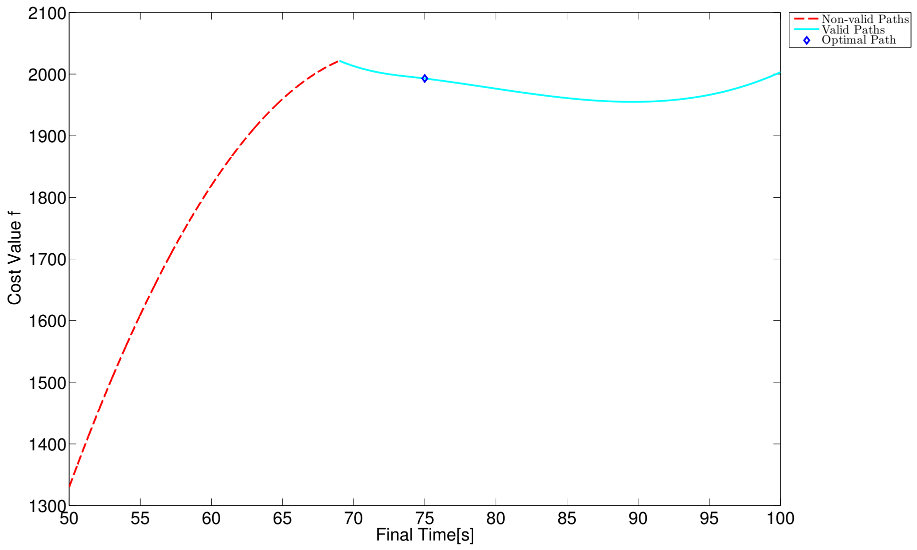

FIGURE B.40: Cost curve for an exit angle of 30°

B.9 Figures for an exit angle of 45°

B.9.1 Path Map

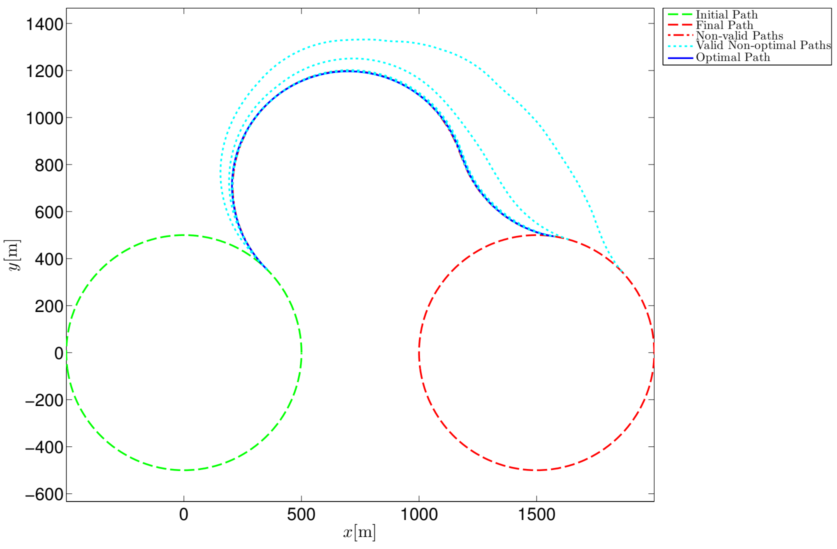

FIGURE B.41: All paths found for an exit angle of 45°

B.9.2 Velocity Magnitude Profiles

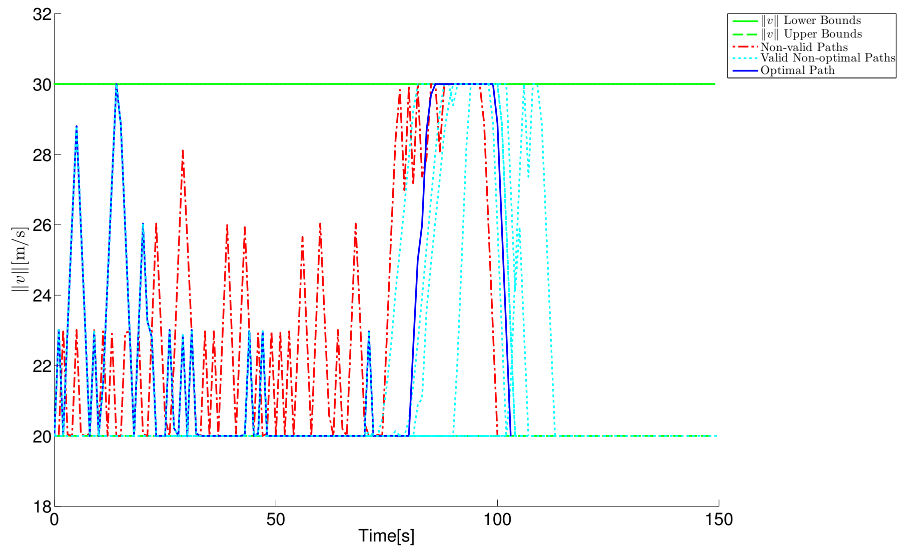

FIGURE B.42: The velocity profiles for all paths found for an exit angle of 45°

B.9.3 Control History Profiles

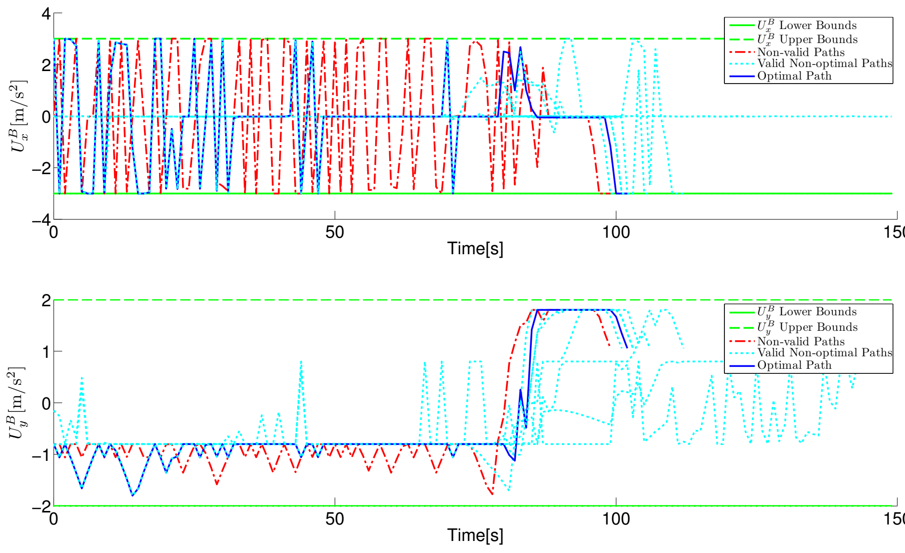

FIGURE B.43: The U_x and U_y control history for all paths found for an exit angle of 45°

B.9.4 Path Curvature Conditions

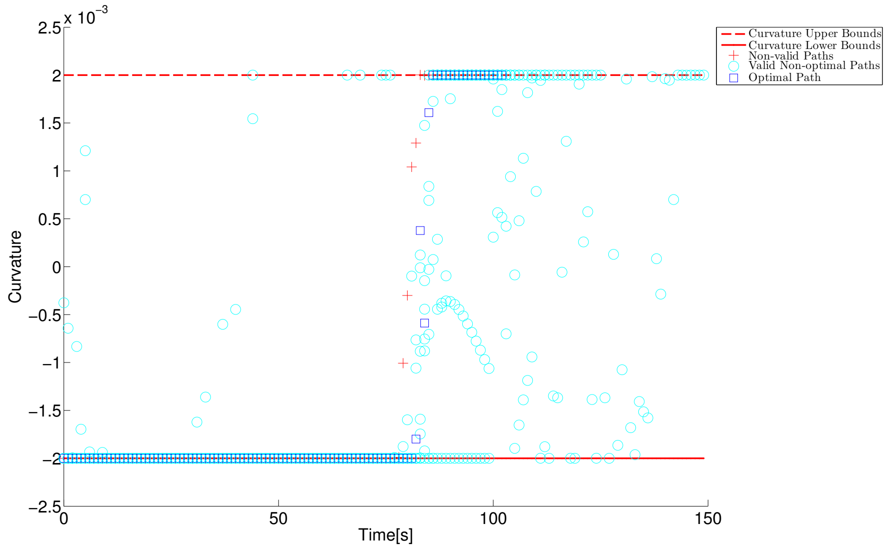

FIGURE B.44: Path curvatures for all paths found for an exit angle of 45°

B.9.5 Path Cost Function Curve

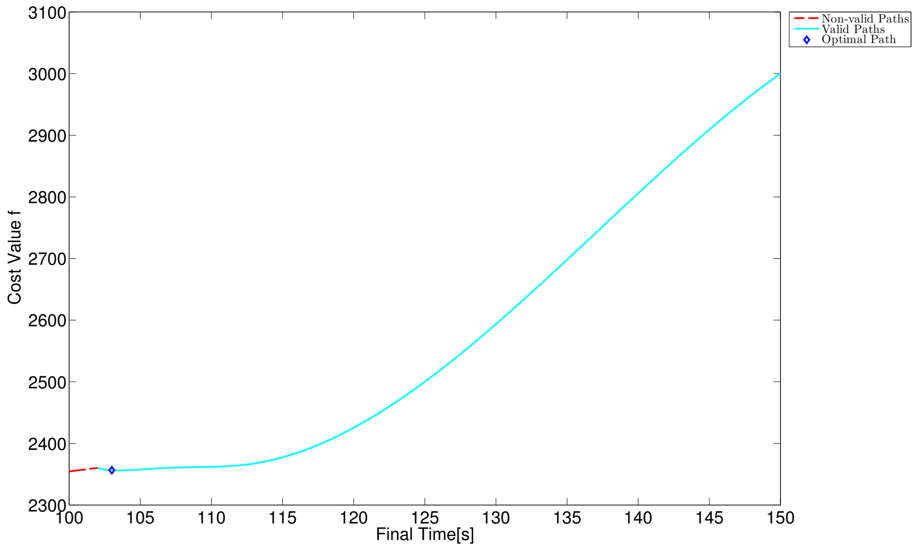

FIGURE B.45: Cost curve for an exit angle of 45°

B.9.6 Velocity Magnitude Profiles

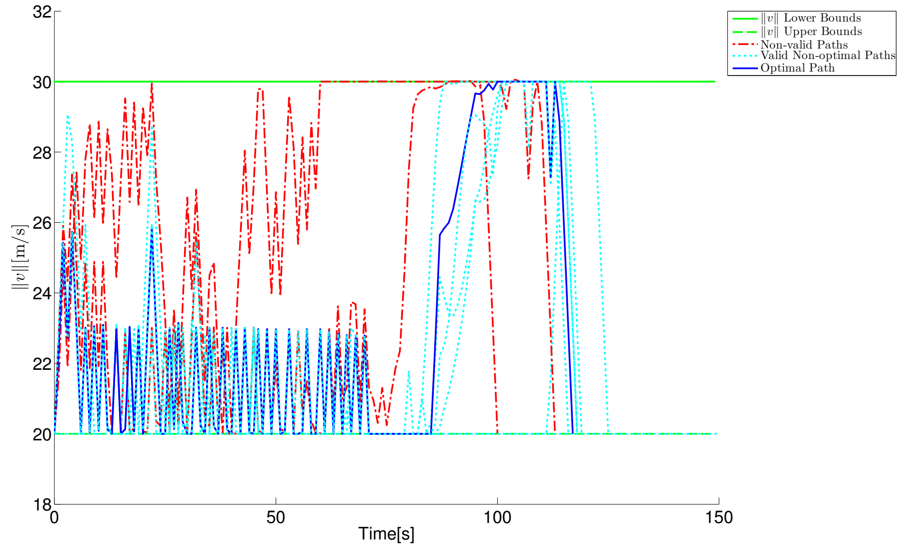

FIGURE B.46: The velocity profiles for all paths found for an exit angle of 60°

B.9.7 Control History Profiles

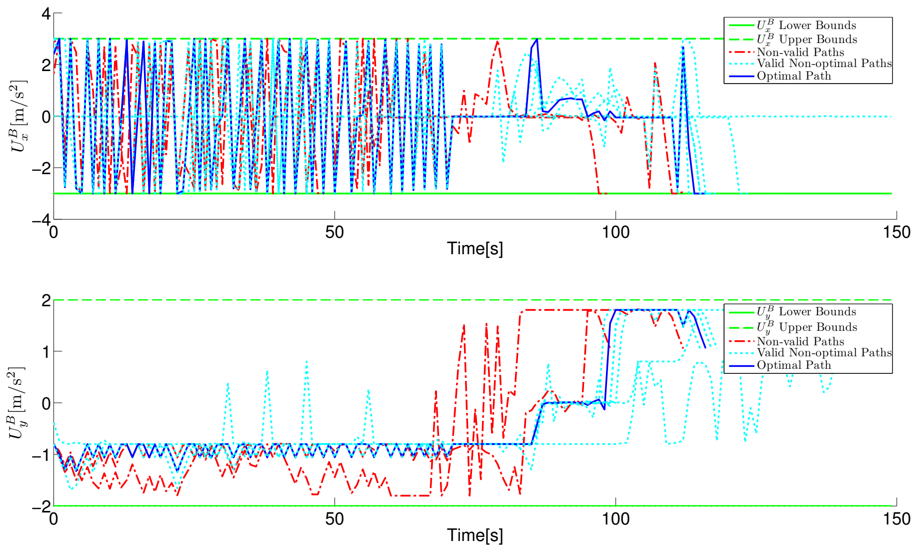

FIGURE B.47: The U_x and U_y control history for all paths found for an exit angle of 60°

B.9.8 Path Curvature Conditions

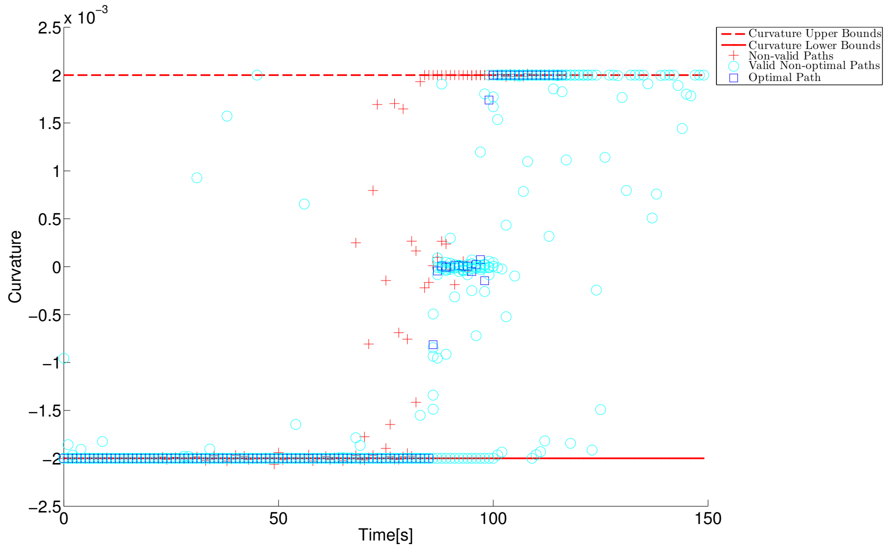

FIGURE B.48: Path curvatures for all paths found for an exit angle of 60°

B.9.9 Path Cost Function Curve

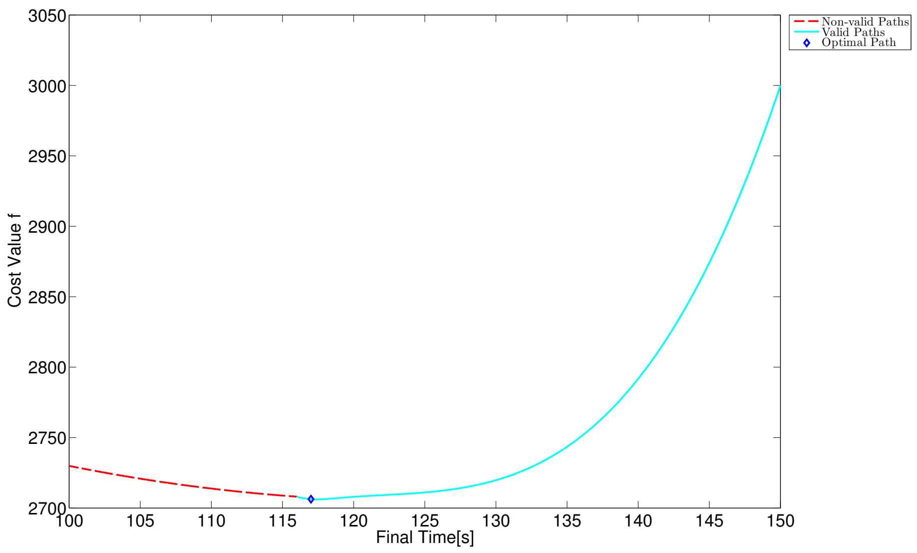

FIGURE B.49: Cost curve for an exit angle of 60°

Appendix C

Contour Plots Of The Receding Horizon Algorithm

This appendix shows the contour plots for different receding horizon algorithm parameters for the uniquely different paths as described in Section 4.2 and does not include any figure already shown in the main work.

C.1 Cost Contours for -75°Exit Angle

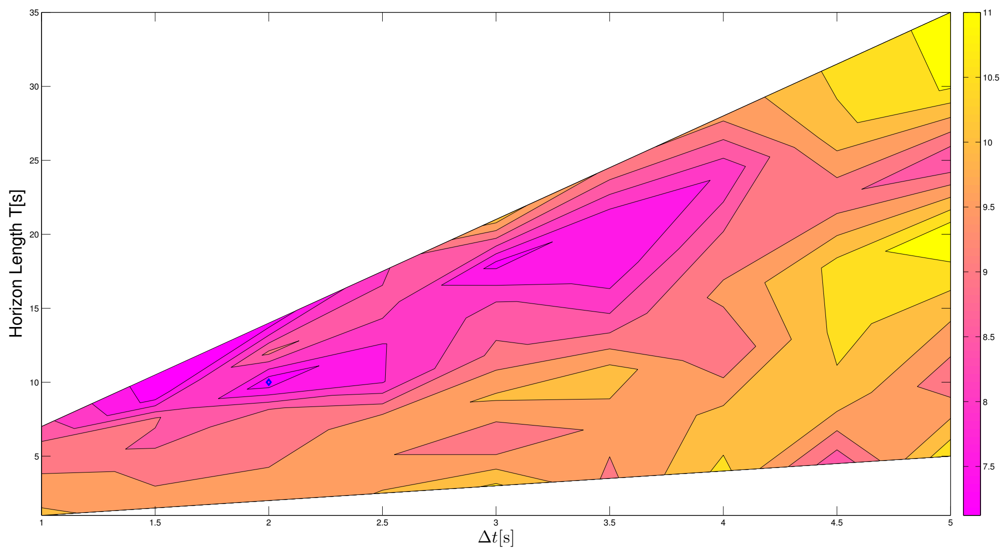

Figure C.1: Cost contours for a Receding Horizon from -75°over various horizon lengths and Δt values

C.2 Cost Contours for -60°Exit Angle

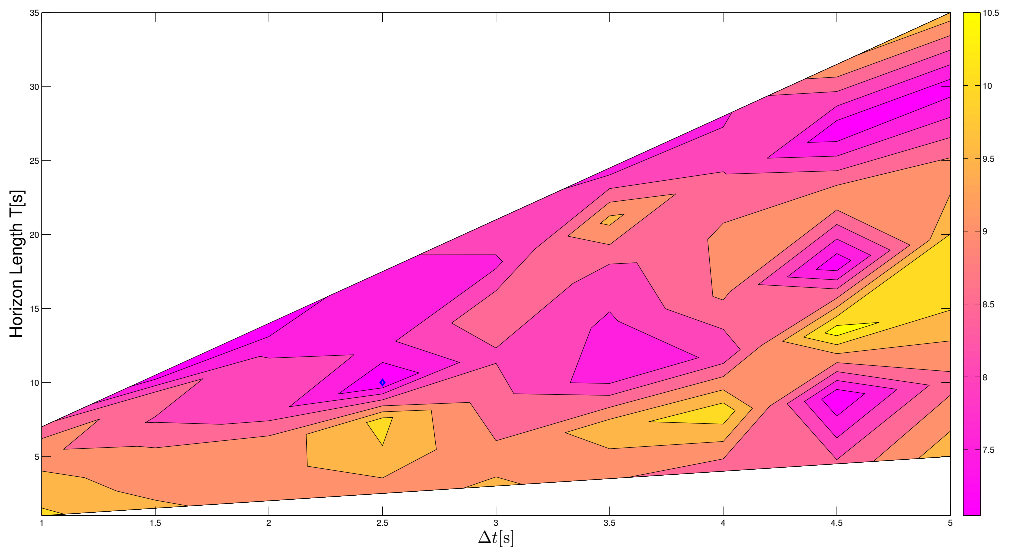

FIGURE C.2: Cost contours for a Receding Horizon from -60°over various horizon lengths and Δt values

C.3 Cost Contours for -45°Exit Angle

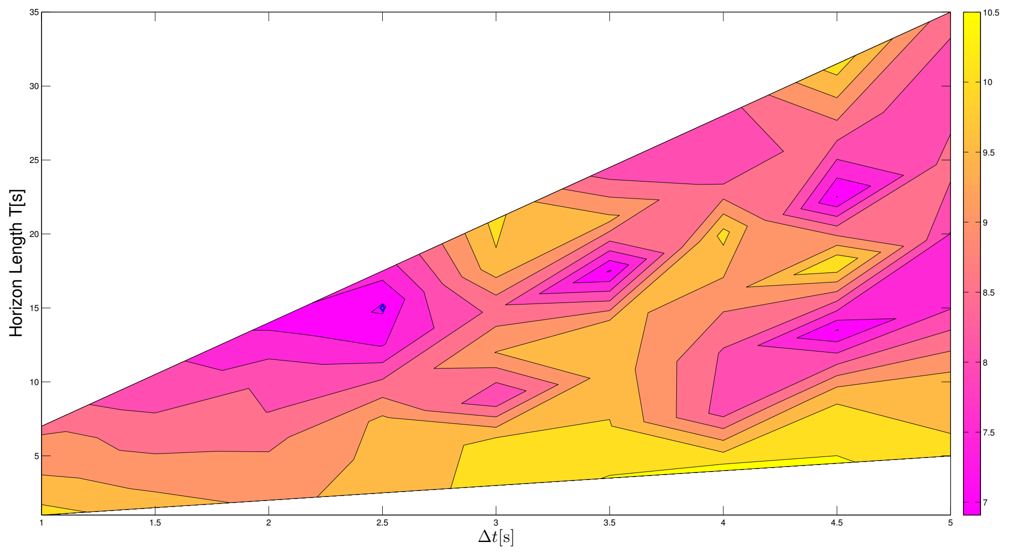

FIGURE C.3: Cost contours for a Receding Horizon from -45°over various horizon lengths and Δt values

C.4 Cost Contours for -30°Exit Angle

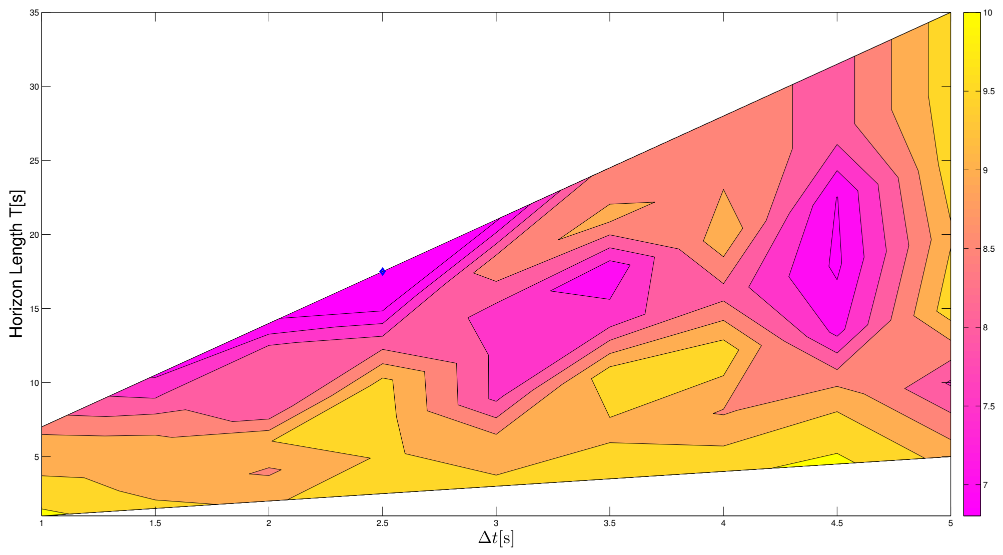

FIGURE C.4: Cost contours for a Receding Horizon from -30°over various horizon lengths and Δt values

C.5 Cost Contours for -15°Exit Angle

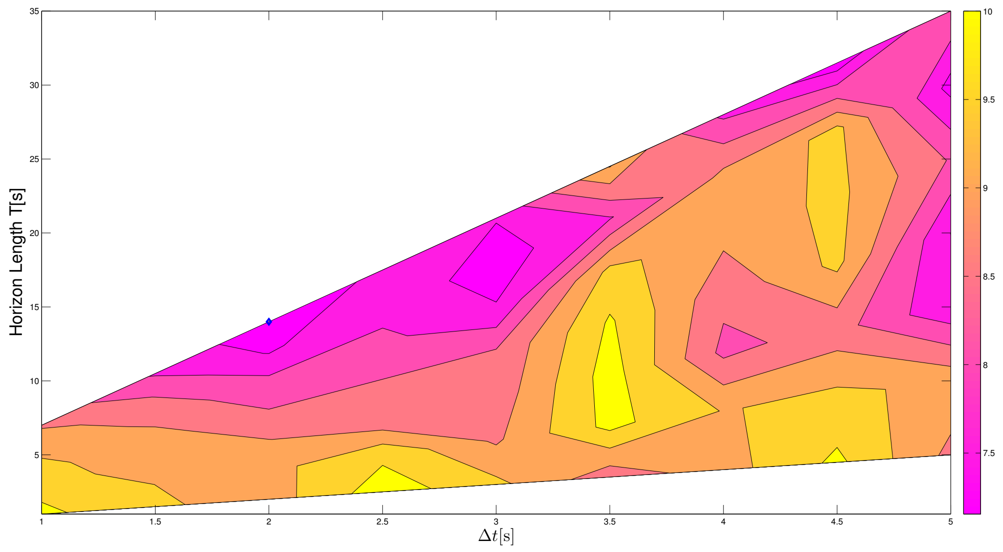

FIGURE C.5: Cost contours for a Receding Horizon from -15°over various horizon lengths and Δt values

C.6 Cost Contours for 0°Exit Angle

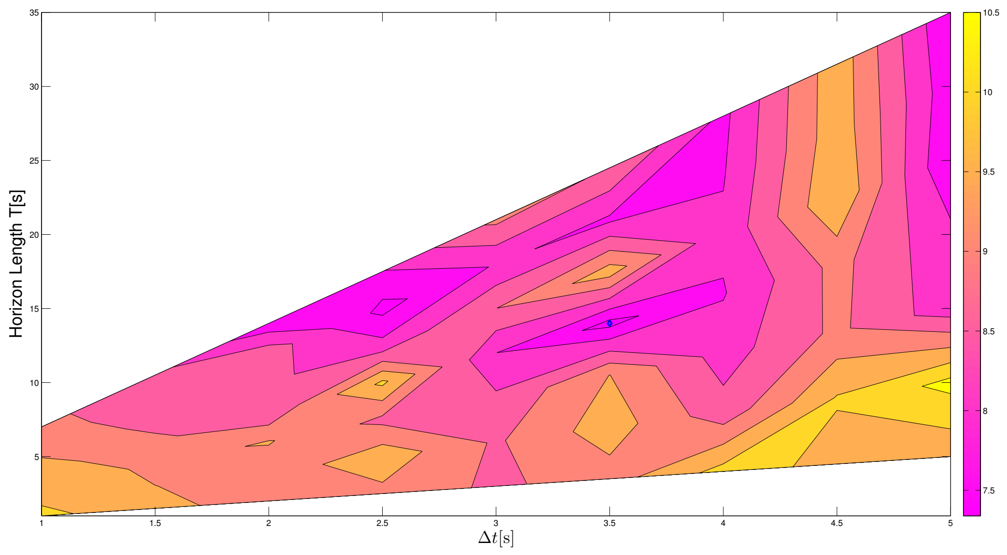

FIGURE C.6: Cost contours for a Receding Horizon from 0°over various horizon lengths and Δt values

C.7 Cost Contours for 15°Exit Angle

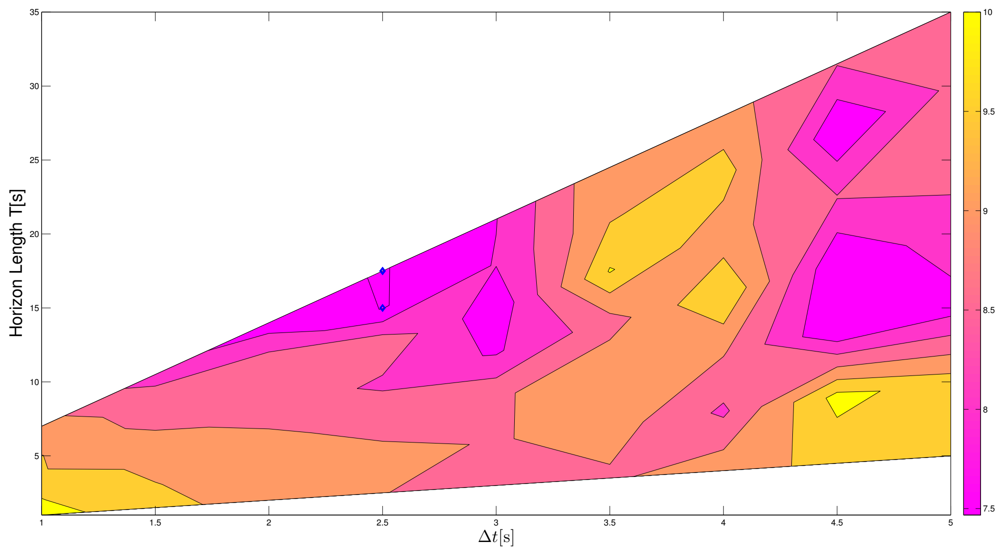

FIGURE C.7: Cost contours for a Receding Horizon from 15°over various horizon lengths and Δt values

C.8 Cost Contours for 30°Exit Angle

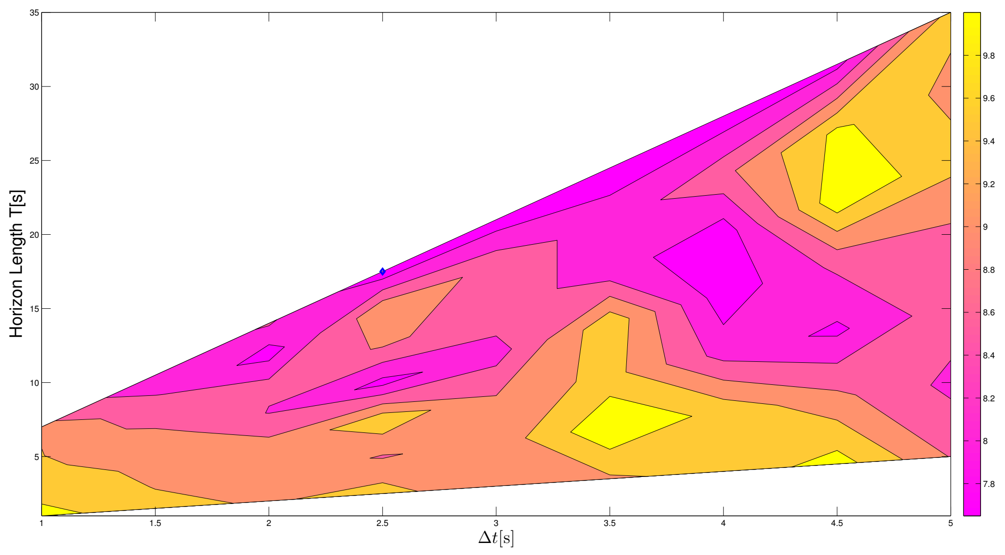

FIGURE C.8: Cost contours for a Receding Horizon from 30°over various horizon lengths and Δt values

C.9 Cost Contours for 45°Exit Angle

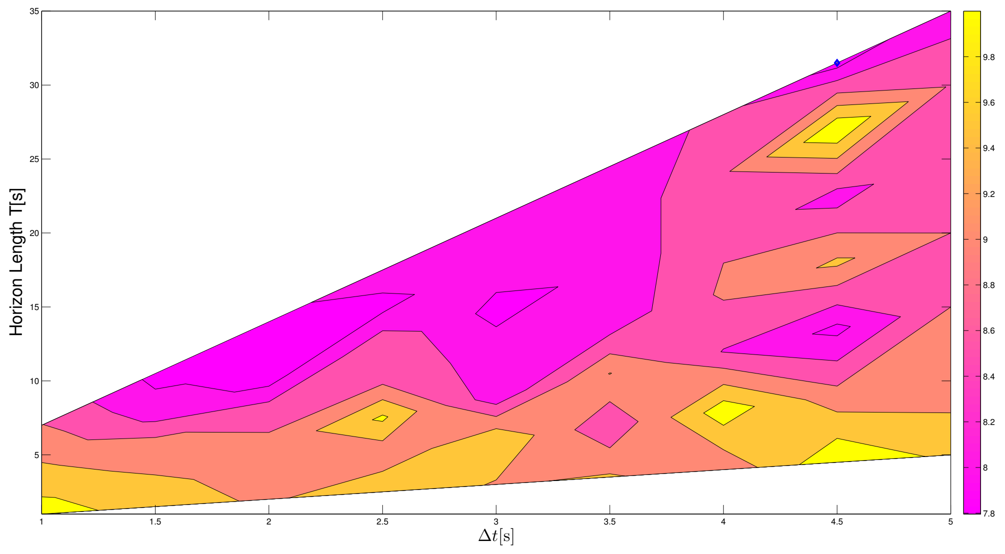

FIGURE C.9: Cost contours for a Receding Horizon from 45°over various horizon lengths and Δt values

C.10 Cost Contours for 60°Exit Angle

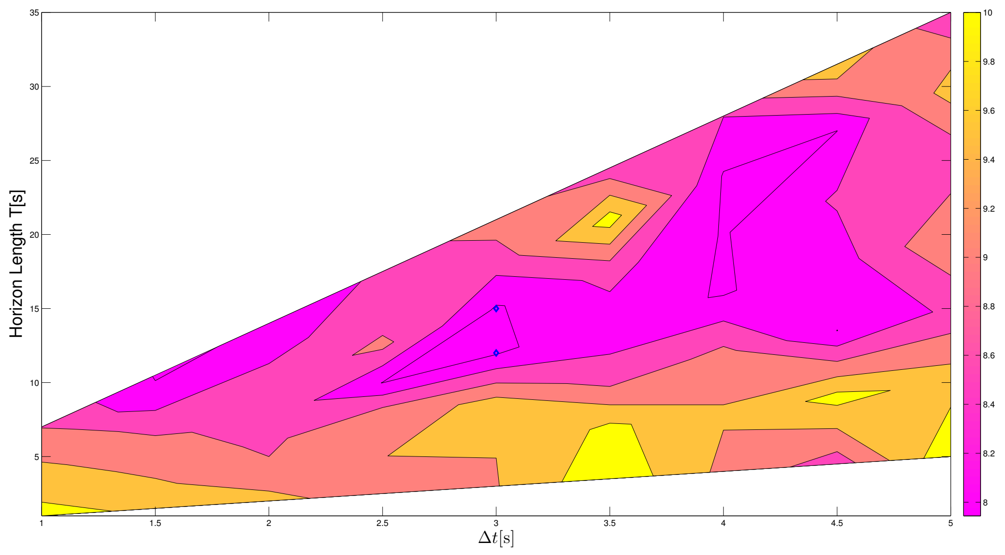

FIGURE C.10: Cost contours for a Receding Horizon from 60°over various horizon lengths and Δt values

Appendix D

Discrete Shooting Method Algorithm Flow Diagram

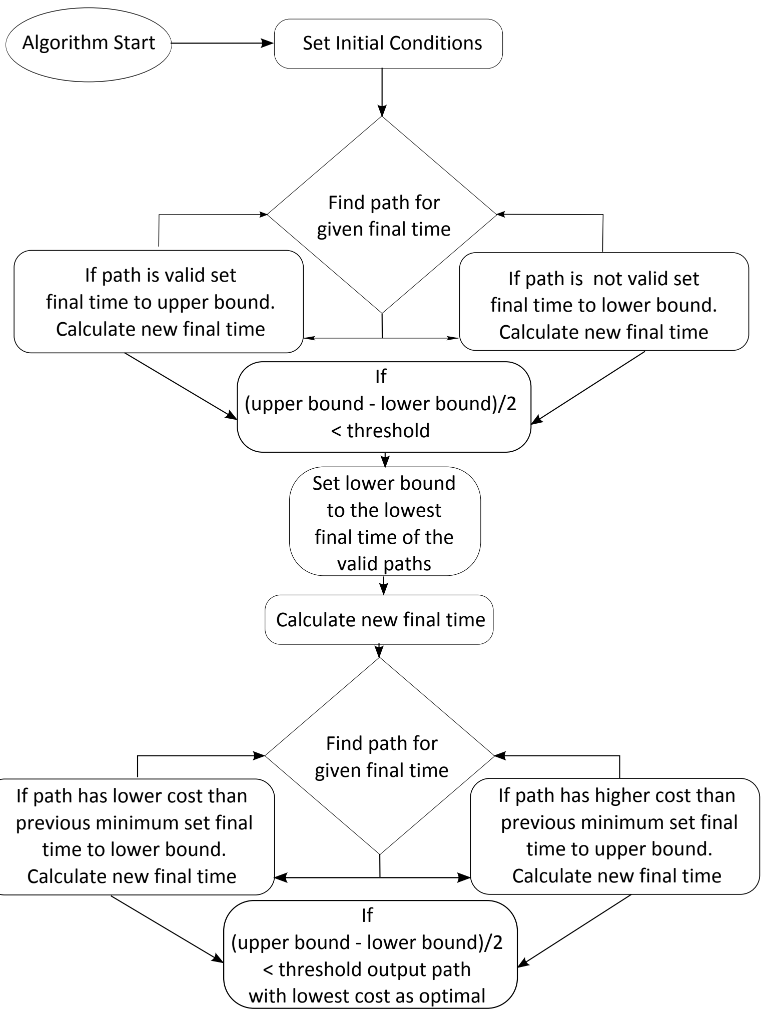
Algorithm Start
Set Initial Conditions
Find path for given final time
If path is valid set final time to upper bound. Calculate new final time
If path is not valid set final time to lower bound. Calculate new final time
If (upper bound - lower bound)/2 < threshold
Set lower bound to the lowest final time of the valid paths
Calculate new final time
Find path for given final time
If path has lower cost than previous minimum set final time to lower bound. Calculate new final time
If path has higher cost than previous minimum set final time to upper bound. Calculate new final time
If (upper bound - lower bound)/2 < threshold output path with lowest cost as optimal

Figure D.1: Discrete Shooting Method Algorithm Flow Diagram

Bibliography

[1] Dr Kimon P. Valavanis, editor. *Advances in Unmanned Aerial Vehicles*, volume 33 of *Intelligent Systems, Control and Automation: Science and Engineering*. Springer, 2007.

[2] Paul Lewis. Cctv in the sky: police plan to use military-style spy drones, Jan 2010. URL `http://www.guardian.co.uk/uk/2010/jan/23/cctv-sky-police-plan-drones`.

[3] Jim Loney. Remote controlled planes to explore hurricanes, May 2008. URL `http://uk.reuters.com/article/2008/05/26/us-storm-hurricanes-drones-idUKN2531234020080526`.

[4] Luca De Filippis, Giorgio Guglieri, and Fulvia Quagliotti. Path planning strategies for uavs in 3d environments. *Journal of Intelligent & Robotic Systems*, pages 1–18, 2011. ISSN 0921-0296. URL `http://dx.doi.org/10.1007/s10846-011-9568-2`. 10.1007/s10846-011-9568-2.

[5] Diclehan Tezcaner and Murat Kksalan. An interactive algorithm for multi-objective route planning. *Journal of Optimization Theory and Applications*, 150:379–394, 2011. ISSN 0022-3239. URL `http://dx.doi.org/10.1007/s10957-011-9838-y`. 10.1007/s10957-011-9838-y.

[6] Xu Chu Ding, A.R. Rahmani, and M. Egerstedt. Multi-uav convoy protection: An optimal approach to path planning and coordination. *Robotics, IEEE Transactions on*, 26(2):256 –268, april 2010. ISSN 1552-3098. doi: 10.1109/TRO.2010.2042325.

[7] Yong Bao, Xiaowei Fu, and Xiaoguang Gao. Path planning for reconnaissance uav based on particle swarm optimization. In *Computational Intelligence and Natural Computing Proceedings (CINC), 2010 Second International Conference on*, volume 2, pages 28 –32, sept. 2010. doi: 10.1109/CINC.2010.5643794.

[8] Vitaly Shaferman and Tal Shima. Unmanned aerial vehicles cooperative tracking of moving ground target in urban environments. *Journal Of Guidance, Control, And Dynamics*, 31(5):1360–1371, September-October 2008.

[9] Royal Air Force Directorate of Defence Studies. *Air Power: UAVs: The Wider Context*. No1 Aeronautical Information Documents Unit (No 1 AIDU), 2009. URL http://www.airpowerstudies.co.uk/UAV-Book.pdf.

[10] K.L.B. Cook. The silent force multiplier: The history and role of uavs in warfare. In *Aerospace Conference, 2007 IEEE*, pages 1 –7, march 2007. doi: 10.1109/AERO.2007.352737.

[11] J.M. Sullivan. Revolution or evolution? the rise of the uavs. In *Technology and Society, 2005. Weapons and Wires: Prevention and Safety in a Time of Fear. ISTAS 2005. Proceedings. 2005 International Symposium on*, pages 94 – 101, june 2005. doi: 10.1109/ISTAS.2005.1452718.

[12] R.M. Howard and I. Kaminer. Survey of unmanned air vehicles. In *American Control Conference, 1995. Proceedings of the*, volume 5, pages 2950 –2953 vol.5, jun 1995. doi: 10.1109/ACC.1995.532054.

[13] B.J. Hendrey and R.A. Jarvis. Robot manipulator path planning. In *TENCON '92. "Technology Enabling Tomorrow : Computers, Communications and Automation towards the 21st Century.' 1992 IEEE Region 10 International Conference.*, pages 865 –870 vol.2, nov 1992. doi: 10.1109/TENCON.1992.271847.

[14] H J Sussman and G Tang. Shortest paths for the reeds-shepp car: a worked out example of the use of geometric techniques in nonlinear optimal control. Technical report, 1991.

[15] M. Shibata, S. Takakura, and K. Ohnishi. Path planning for robot manipulator based on middle goal method. In *Industrial Electronics, Control and Instrumentation, 1991. Proceedings. IECON '91., 1991 International Conference on*, pages 1017 –1022 vol.2, oct-1 nov 1991. doi: 10.1109/IECON.1991.239151.

[16] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.

[17] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986. doi: 10.1177/027836498600500106. URL http://ijr.sagepub.com/content/5/1/90.abstract.

[18] H. Chitsaz and S.M. LaValle. Time-optimal paths for a dubins airplane. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2379 –2384, dec. 2007. doi: 10.1109/CDC.2007.4434966.

[19] Jr. Arthur E. Bryson and Yu-Chi Ho. *Applied Optimal Control.* Hemisphere Publishing Corporation, 1975.

[20] Donald E. Kirk. *Optimal Control Theory an Introduction.* Prentice-Hall, 1970.

[21] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497516, 1957.

[22] P. Soueres and J.-P. Laumond. Shortest paths synthesis for a car-like robot. *Automatic Control, IEEE Transactions on*, 41(5):672 –688, may 1996. ISSN 0018-9286. doi: 10.1109/9.489204.

[23] J.-D. Boissonnat, A. Cerezo, and J. Leblond. Shortest paths of bounded curvature in the plane. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2315 –2320 vol.3, may 1992. doi: 10.1109/ROBOT.1992.220117.

[24] Jongrae Kim and Yoonsoo Kim. Moving ground target tracking in dense obstacle areas using uavs. In *The 17th IFAC World Congress, Seoul, Korea*, 2008.

[25] S. Mittal and K. Deb. Three-dimensional offline path planning for uavs using multiobjective evolutionary algorithms. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 3195 –3202, sept. 2007. doi: 10.1109/CEC.2007.4424880.

[26] I.K. Nikolos and A.N. Brintaki. Coordinated uav path planning using differential evolution. In *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, pages 549 –556, june 2005. doi: 10.1109/.2005.1467074.

[27] H. Rezaee and F. Abdollahi. Adaptive artificial potential field approach for obstacle avoidance of unmanned aircrafts. In *Advanced Intelligent Mechatronics (AIM), 2012 IEEE/ASME International Conference on*, pages 1 –6, july 2012. doi: 10.1109/AIM.2012.6305268.

[28] Mr. Joel George Manathara and Prof. Debasish Ghose. Reactive collision avoidance of multiple realistic uavs. *Aircraft Engineering and Aerospace Technology*, 83(6), 2011.

[29] J.J. Rebollo, I. Maza, and A. Ollero. A two step velocity planning method for real-time collision avoidance of multiple aerial robots in dynamic environments. In *Proceedings of the 17th World Congress The International Federation of Automatic Control*, 2008.

[30] Zhang Hong, Yang Liu, Gao Zhongguo, and Cao Yi. The dynamic path planning research for mobile robot based on artificial potential field. In *Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on*, pages 2736 –2739, april 2011. doi: 10.1109/CECNET.2011.5768480.

[31] 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. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, volume 9, pages 633 –637, july 2010. doi: 10.1109/ICCSIT.2010.5565069.

[32] Ding Fu-guang, Jiao Peng, Bian Xin-qian, and Wang Hong-jian. Auv local path planning based on virtual potential field. In *Mechatronics and Automation, 2005 IEEE International Conference*, volume 4, pages 1711 –1716 Vol. 4, 2005. doi: 10.1109/ICMA.2005.1626816.

[33] P. Vadakkepat, Kay Chen Tan, and Wang Ming-Liang. Evolutionary artificial potential fields and their application in real time robot path planning. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 1, pages 256 –263 vol.1, 2000. doi: 10.1109/CEC.2000.870304.

[34] T. Tsuji, P.G. Morasso, and M. Kaneko. Trajectory generation for manipulators based on artificial potential field approach with adjustable temporal behavior. In *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 2, pages 438 –443 vol.2, nov 1996. doi: 10.1109/IROS.1996.570811.

[35] J. Barraquand and J.-C. Latombe. A monte-carlo algorithm for path planning with many degrees of freedom. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 1712 –1717 vol.3, may 1990. doi: 10.1109/ROBOT.1990.126256.

[36] J. Binney and G.S. Sukhatme. Branch and bound for informative path planning. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2147 –2154, may 2012. doi: 10.1109/ICRA.2012.6224902.

[37] A. Eele and A. Richards. Rapid updating for path-planning using nonlinear branch-and-bound. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3575 –3580, may 2010. doi: 10.1109/ROBOT.2010.5509732.

[38] Hoai Le Thi, Duc Nguyen, and Tao Pham Dinh. Globally solving a nonlinear uav task assignment problem by stochastic and deterministic optimization approaches.

Optimization Letters, pages 1–15, 2010. ISSN 1862-4472. URL http://dx.doi.org/10.1007/s11590-010-0259-x. 10.1007/s11590-010-0259-x.

[39] Alison Eele and Arthur Richards. Path-planning with avoidance using nonlinear branch-and-bound optimization. *Journal Of Guidance, Control, And Dynamics*, 32 (2):384–394, March-April 2009.

[40] S.J. Rasmussen and T. Shima. Branch and bound tree search for assigning cooperating uavs to multiple tasks. In *American Control Conference, 2006*, page 6 pp., june 2006. doi: 10.1109/ACC.2006.1656541.

[41] J.A. Cobano, R. Conde, D. Alejo, and A. Ollero. Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4429 –4434, may 2011. doi: 10.1109/ICRA.2011.5980246.

[42] Douglas Macharet, Armando Neto, and Mario Campos. Feasible uav path planning using genetic algorithms and bzier curves. In Antnio da Rocha Costa, Rosa Vicari, and Flavio Tonidandel, editors, *Advances in Artificial Intelligence SBIA 2010*, volume 6404 of *Lecture Notes in Computer Science*, pages 223–232. Springer Berlin / Heidelberg, 2011. URL http://dx.doi.org/10.1007/978-3-642-16138-4_23.

[43] Ze Cheng, Ying Sun, and Yanli Liu. Path planning based on immune genetic algorithm for uav. In *Electric Information and Control Engineering (ICEICE), 2011 International Conference on*, pages 590 –593, april 2011. doi: 10.1109/ICEICE.2011.5777407.

[44] Oktay Baysal Y. Volkan Pehlivanoglu and Abdurrahman Hacioglu. Path planning for autonomous uav via vibrational genetic algorithm. *Aircraft Engineering and Aerospace Technology: An International Journal*, 79(4):352–359, 2007.

[45] T. Shima, S.J. Rasmussen, and A.G. Sparks. Uav cooperative multiple task assignments using genetic algorithms. In *American Control Conference, 2005. Proceedings of the 2005*, pages 2989 – 2994 vol. 5, june 2005. doi: 10.1109/ACC.2005.1470429.

[46] Dong Jia and Juris Vagners. Parallel evolutionary algorithms for uav path planning. In *AIAA 1st Intelligent Systems Technical Conference*, September 2004.

[47] Liu Juan, Cai Zixing, and Liu Jianqin. Premature convergence in genetic algorithm: analysis and prevention based on chaos operator. In *Intelligent Control and Automation, 2000. Proceedings of the 3rd World Congress on*, volume 1, pages 495 –499 vol.1, 2000. doi: 10.1109/WCICA.2000.860016.

[48] M.H. Korayem, M. Bamdad, and S. Bayat. Optimal trajectory planning with maximum load carrying capacity for cable suspended robots. In *Mechatronics and its Applications, 2009. ISMA '09. 6th International Symposium on*, pages 1 –6, march 2009. doi: 10.1109/ISMA.2009.5164817.

[49] S. Khanmohammadi, G. Alizadeh, J. Jassbi, and M. Pourmahmood. A new artificial intelligence approach for 2d path planning for underwater vehicles avoiding static and energized obstacles. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 1988 –1995, june 2008. doi: 10.1109/CEC.2008.4631061.

[50] Hans Georg Bock and Karl J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *International Federation of Automatic Control 9th World Congress*, 1984.

[51] T. Khuswendi, H. Hindersah, and W. Adiprawita. Uav path planning using potential field and modified receding horizon a* 3d algorithm. In *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, pages 1 –6, july 2011. doi: 10.1109/ICEEI.2011.6021579.

[52] J.R. Riehl, G.E. Collins, and J.P. Hespanha. Cooperative graph-based model predictive search. pages 2998 –3004, dec. 2007. ISSN 0191-2216. doi: 10.1109/CDC.2007.4435025.

[53] R. Prazenica, A. Kurdila, R. Sharpley, and J. Evers. Vision-based geometry estimation and receding horizon path planning for uavs operating in urban environments. In *American Control Conference, 2006*, page 6 pp., june 2006. doi: 10.1109/ACC.2006.1657155.

[54] Jongrae Kim and J.L. Crassidis. Uav path planning for maximum visibility of ground targets in an urban area. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1 –7, july 2010.