An Efficient Algorithm for Commercial Aircraft Trajectory Optimization in the Air Traffic System

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Raghuveer Devulapalli

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS OF SCIENCE

Yiyuan J. Zhao, Adviser

July 2012

Acknowledgements

I would like to thank Professor Yiyuan J. Zhao for the wonderful opportunity and the support he gave which made this work possible. I would also like to thank Professor John Gunnar Carlsson for all the assistance and feedback he provided.

Abstract

A discrete search strategy is presented to determine optimal aircraft trajectories which can be unconstrained or regulated to follow current Air Traffic Control (ATC) procedures. The heuristic based Astar (A*) search algorithm has been selected for its efficiency and its inherent ability to handle numerous constraints as a discrete method. A point-mass aircraft model is assumed to accurately simulate commercial aircraft dynamics for the provided trajectories. The two dimensional space and the states of aircraft have been divided into discrete pieces. To show the effectiveness of the algorithm, two-dimensional vertical and horizontal profile are simulated. Simulation results compare optimal trajectories that range from unconstrained to those that completely adhere to strict ATC procedures. Those trajectories following ATC procedures follow a segmented flight pattern where each segment follows specified objectives, terminating when certain criteria has been met. Trajectories are optimized for a combination of time and fuel with an emphasis on reducing fuel consumption.

Contents

List of Tables

List of Figures

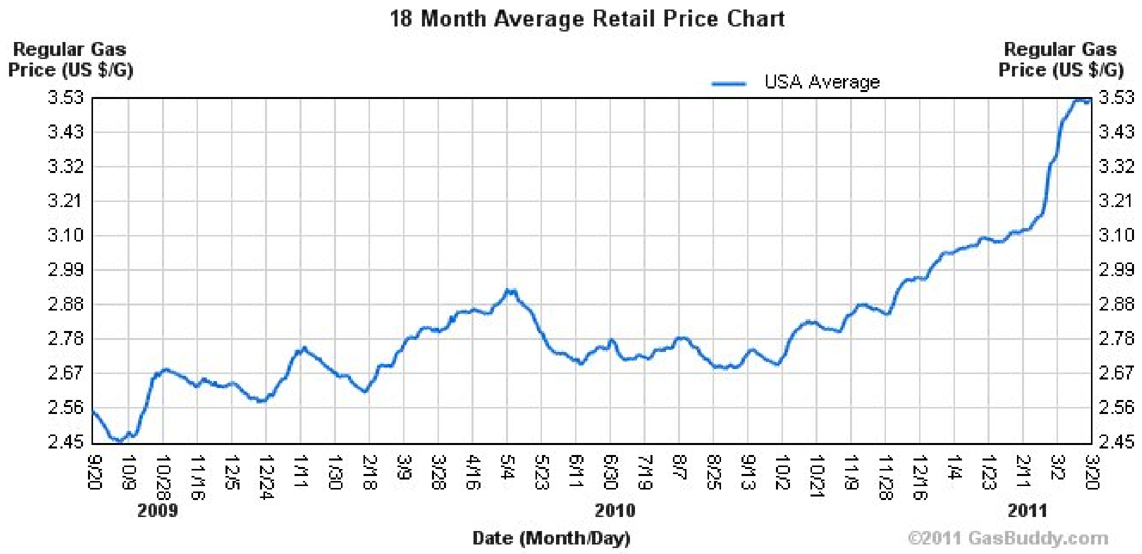
Chapter 1

Introduction

The determination of optimal aircraft trajectories has been of considerable interest to aircraft dynamicists for almost 50 years. Efforts were put in trying to minimize fuel, time and more recently emissions and noise. Looking at the growth of commercial airlines historically and the current trends it only points towards a tremendous increase in air traffic in the next decade or so. It is expected that the commercial fleet will double in the next 20 years. All of these mean that we need new ways to keep track of in flight aircraft to monitor flights in and out of airports. Building bigger airports and increasing runways is way to deal with the air traffic congestion but does not address the issue of saving fuel, reducing noise and emissions. In fact, noise and emissions remain a saturation point for enlarging airports. Continuous descent approach was designed specifically designed for noise abatement during aircraft arrivals. During tests conducted for CDA approaches at Louseville airport, it was also found that the procedure allowed fuel savings of about 400-500lbs per flight [1].

The most significant growth has been that of the ever increasing fuel prices (Figure 1.1). For the amount of fuel airline industry consumes, the fuel price is a huge factor affecting their growth and profits. Airliners are now looking for every odd possible way to reduce fuel consumptions. Singapore Airlines has implemented digital

Figure 1.1: Fuel prices over the years



versions of its in-flight magazines in a bid to reduce fuel costs [2]. Cathay Pacific strips paint off some planes (paint can weigh 440 lbs on a 747) [3]. Southwest uses Pratt & Whitney's EcoPower for its systematic engine wash program for Boeing 737-700 series aircraft in order to increase engine efficiency and save fuel [4]. Irrespective of how effective these ways turn out to be, it only goes to show how desperate airlines are to reduce weight in order to save up on fuel. This is where a flight trajectory optimization with emphasis on fuel becomes really important. The existing flight planning techniques work in a way which is convenient to use but are clearly sub-optimal. A vertical profile for the flight and cruise flight are planned independent of each other. By very nature of this technique the planning is not globally optimal. Combined with all of the Air Traffic regulations/restrictions and other constraints in place, what we get is an inefficient flight path. Hence, there is a urgent need to develop an efficient method to come up with an optimal flight path.

1.1 Problem Statement

The aim of this work is to develop a very efficient and flexible numerical method to determine an optimal flight trajectory which is realistic for commercial flights. It should be able to solve different types of optimization problem (like the 2-Dimensional level flight, 2-Dimensional vertical profile planning and a complete 3-Dimensional flight trajectory optimization problem). To have a realistic flight planning process one has to incorporate all of the constraints imposed on a flight. The most basic and obvious constraint is to model the flight dynamics and performance. Air traffic control enforces constraints for a commercial flight, to follow certain flight procedures specially during takeoff and landing. There could be possible set of waypoints to be followed during a flight or certain restricted airspace to be avoided. And one also places restrictions on certain flight maneuvers to keep in mind of passenger comforts. An efficient flight planning algorithm is expected to handle any or all of these constraints while solving for a flight path. Also for an efficient flight planning process it is of utmost importance to be able to incorporate weather, mainly the latest available wind models.

The solution obtained has to efficient and reliable. Currently the flight planning procedure involves two steps. First the vertical profile of the flight is obtained via a 2 Dimensional optimization and then the level flight optimization problem is solved separately. Even though the level flight portion of the flight is the major part of the entire flight duration, by its very nature this is a sub-optimal solution. It is highly desirable to have one consistent solution and a process where one can do a complete 3 Dimensional flight trajectory optimization, where the 2 Dimensional level flight and vertical profile just becomes a special case of the solution obtained.

1.2 Key Contributions

It is aimed to show that graph method implementation for commercial aircraft trajectory optimization can be effectively used to solve various problem formulations. There is no difference in implementation of 2-D vertical, horizontal flight profile or the full 3-D model. Initial guess does not play any part in the solution structure and the method always yields an absolute minimum, which means convergence is guaranteed. If the algorithm does not find a solution then it is safe to assume that a solution does not exist. The method can accommodate as many constraints as needed without increasing the runtime (the runtime may actually decrease as the number of constraints increases). In addition, the wind model to be incorporated can be any linear or non-linear function or in fact does not even have to be continuous. A very general wind field can be used without any simplifications. Numerical simulations are presented for two cases: horizontal flight planning and vertical profile planning.

1.3 Outline

The thesis is organized in the following way:

- Chapter 2 briefly describes flight planning process and commercial flight procedures.
- Chapter 3 provides a history of previous relevant work and literature survey.
- Chapter 4 describes the equations of motion for an aircraft using the point mass model for trajectory optimization.
- Chapter 5 describes formally the problem statement we attempt to solve and describes in details the solution strategy used.

- Chapter 6 presents the results of all the simulations performed for horizontal and vertical flight optimization.
- Chapter 7 concludes the thesis with discussions and future work recommendation.

Chapter 2

Flight Planning

The process of producing a flight plan to describe the trajectory of a proposed flight is called flight planning. It basically involves coming with an estimate of amount of fuel required for the flight and the trajectory of flight, describing the route to be taken to reach the destination safely, which complies with the air traffic control procedures/regulations. Commercial airlines would wish to plan the trajectory in such a way that it would minimize a certain cost index.

The procedure of coming up with a flight plan is highly dependent on a lot of factors and is very problem specific. It depends on specific origin-destination pair, type of aircraft being used and weather forecast. Flight planning requires accurate weather forecasts so that fuel consumption calculations can account for the fuel consumption effects of head or tail winds and air temperature. Producing an optimal flight plan even for a given origin-destination pair, a specific aircraft and initial weight, is never a one time process. The air temperature affects the efficiency/fuel consumption of aircraft engines. The wind may provide a head or tail wind component which in turn will increase or decrease the fuel consumption by increasing or decreasing the air distance to be flown. Hence, accurately updated weather forecast plays a crucial role in coming up with an optimal trajectory.

Furthermore, it is required as per safety procedures to carry fuel beyond the minimum needed to fly to the specified destination. Under the supervision of air traffic control, aircraft flying in controlled airspace must follow predetermined routes known as airways, even if such routes are not as economical as a more direct flight. Within these airways, aircraft must maintain flight levels, specified altitudes usually separated vertically by 1000 or 2000 feet, depending on the route being flown and the direction of travel. When aircraft with only two engines are flying long distances across oceans, deserts, or other areas with no airports, they have to satisfy extra safety rules to ensure that such aircraft can reach some emergency airport if one engine fails. Rate of fuel burn depends on ambient temperature, aircraft speed, and aircraft altitude, none of which are entirely predictable. Rate of fuel burn also depends on airplane weight, which changes as fuel is burned.

Coming up with an accurate optimized flight plan for commercial airlines is by itself a big industry. Producing an accurate optimized flight plan requires a large number of calculations (millions), so commercial flight planning systems make extensive use of computers . Some commercial airlines have their own internal flight planning system, while others employ the services of external planners. While developing a software tool to plan flight trajectory, it is necessary to incorporate commercial flight procedures followed. They add in a lot more constraints to the flight path. These are discussed in the folloring section.

2.1 Commercial flight procedures

In a realistic commerical aircraft flight, the complete trajectory is broken into series of flight segments, mainly broken into phases as shown in figure Figure 2.1. Each of these phases in turn include several flight segments, where each segment can be defined by control objectives and termination conditions designed to be flyable. Mathematically, each flight segment can be described by two constant control variables selected from

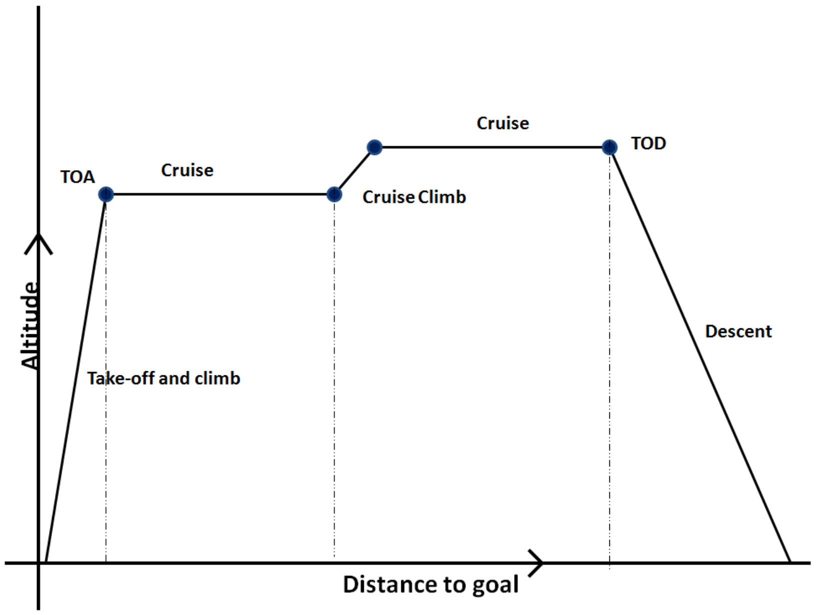


Figure 2.1: Complete trajectory for an aircraft from takeoff to landing

among engine thrust setting, Mach number or calibrated airspeed, and altitude rate or flight path angle [5]. Furthermore airline specifications often combine a number of segments in a specified order to form certain profiles. A lateral profile can be defined for an aircraft flying level and turning at constant bank angle using waypoints. Aircraft take off and descent is divided into a sequence of segments defining a vertical profile. Each vertical flight segment is defined by choosing exactly two control objectives, at most one from each category. This either explicitly or implicitly defines how the aircraft pitch and thrust are controlled. For example, choosing constant Mach and idle thrust defines a descent segment that control speed using aircraft pitch.

2.1.1 Lateral profile

Lateral profile of a flight usually describes the level flight portion. The aircraft makes turns at a constant bank angle. A lateral profile is usually described by a sequence of waypoints (Area Navigation (RNAV)). Most waypoints are classified as compulsory reporting points, i.e. the pilot (or the onboard flight management system) reports the aircraft position to air traffic control as the aircraft passes a waypoint. There are two main types of waypoints. A named waypoint appears on aviation charts with a known latitude and longitude. Such waypoints over land often have an associated radio beacon so that pilots can more easily check where they are. Useful named waypoints are always on one or more airways. A geographic waypoint is a temporary position used in a flight plan, usually in an area where there are no named waypoints, e.g. most oceans in the southern hemisphere. Air traffic control require that geographic waypoints have latitudes and longitudes which are a whole number of degrees.

2.1.2 Vertical profile - SID & STAR

After take-off an aircraft follows a Departure Procedure (SID or Standard Instrument Departure) which defines a pathway from an airport runway to a waypoint on an airway, so that an aircraft can join the airway system in a controlled manner. Most of the climb portion of a flight will take place on the SID. Although a SID will keep aircraft away from terrain, it is optimized for ATC route of flight and will not always provide the lowest climb gradient. It strikes a balance between terrain and obstacle avoidance, noise abatement (if necessary) and airspace management considerations

Before landing an aircraft follows an Arrival Procedure (STAR or Standard Terminal Arrival Route) which defines a pathway from a waypoint on an airway to an airport runway, so that aircraft can leave the airway system in a controlled manner. STAR usually covers the phase of a flight that lies between the top of descent from cruise or en-route flight and the final approach to a runway for landing. Normally that final approach starts at the so-called Initial approach Fix (IAF). A typical STAR consists of a set of starting points, called transitions, and a description of routes (typically via waypoints) from each of these transitions to a point close to the

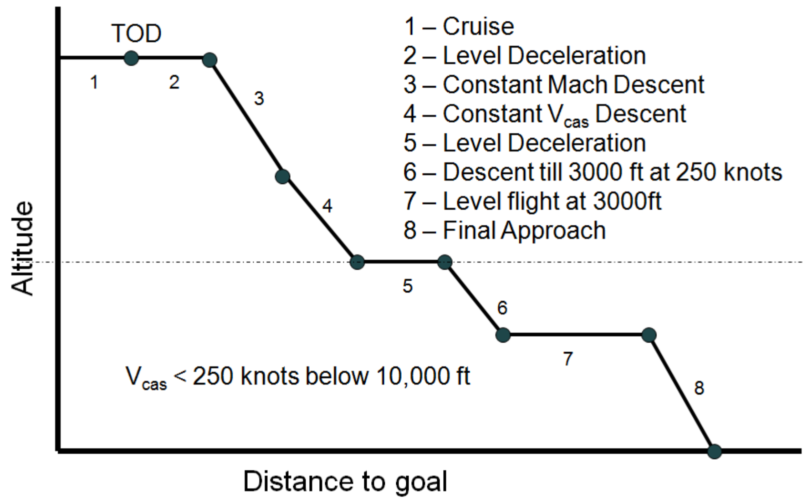

TOD
1 – Cruise
2 – Level Deceleration
3 – Constant Mach Descent
4 – Constant V_{cas} Descent
5 – Level Deceleration
6 – Descent till 3000 ft at 250 knots
7 – Level flight at 3000ft
8 – Final Approach
V_{cas} < 250 knots below 10,000 ft
1
2
3
4
5
6
7
8
Altitude
Distance to goal

Figure 2.2: Descent profile for a commercial aircraft

destination airport. There the aircraft can join an instrument approach (IAP) or will be vectored for a final approach by the APP control. Not all airports have published STARs. However, most relatively large or not easily accessible (for example, in the mountainous area) airports do. Sometimes several airports in the same area share a single STAR; in such case, aircraft destined for any of the airports in such group follow the same arrival route up until reaching the final waypoint, after which they join approaches for their respective destination airports.

2.1.3 Conventional descent approach

With the conventional aircraft descent, an aircraft would be given clearance by the air traffic control from the bottom level of the holding stack (normally an altitude of 6000 to 7000 feet) to descend to an altitude of typically 3000 feet. The aircraft would then fly level for several miles before intersecting the final 3 degree glidepath to the runway. During this period of level flight, the pilot would need to apply additional engine power to maintain constant speed.

As an example, a certain type of descent profile is described. A conventional

No.	Segment Type	Integration Variables	Constant Controls	Capture variables
1	Cruise	s	$\dot{h} = 0$, M	s
2	Constant Deceleration	s, V	h	$M = M_d$
3	Constant Mach	s, h	$M = M_d$	$V_{cas} = V_{cas_d}$
4	Constant V_{cas}	s, h	$V_{cas} = V_{cas_d}$	h = 10,000 ft
5	Level Deceleration	s, V	h = 10,000 ft	$M, V_{cas} ors$

Table 2.1: Flight Segments in a Descent Profile

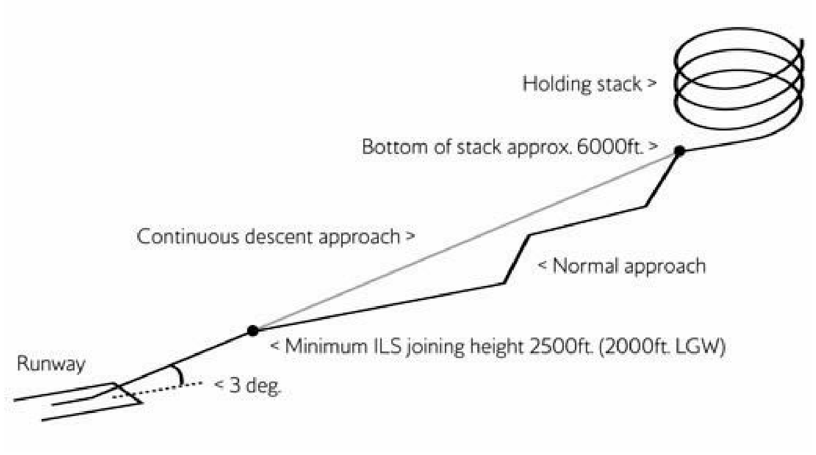

Holding stack >
Bottom of stack approx. 6000ft. >
Continuous descent approach >
< Normal approach
< Minimum ILS joining height 2500ft. (2000ft. LGW)
Runway
< 3 deg.

Figure 2.3: Comparison between a conventional approach and CDA at the London Heathrow Airport

descent profile can be described by a series of segments. Each segment ends and next segment begins when a certain scalar condition is met. These conditions are called capture or termination conditions [5]. In general, these can be either distance, speed, altitude, time etc. A nominal descent trajectory pattern is shown in the figure Figure 2.2. The first segment is the constant Mach M_c and constant altitude h_c segment until the cruise reduction point (CSR, the beginning of the last segment of the cruise phase) is reached.

2.1.4 Continuous Descent Approach

In contrast to a conventional descent approach, when a continuous descent approach (CDA) is flown, the aircraft stays higher for longer, descending continuously from the level of the bottom stack (or higher if possible) and avoiding any level segments of flight prior to intercepting the 3 degree glidepath (figure Figure 2.3). Such an approach would require significantly less engine thrust than prolonged level flight, hence a significant fuel savings can be expected for the final arrival phase of the flight. Less engine thrust also means lesser emissions and because the aircraft flying a CDA is higher above the ground for a longer period of time, the noise impact on the ground is reduced in certain areas under the approach path.

In a flight demonstration test, conducted by Boeing and MIT at the Louseville aiport, such a procedure was shown to reduce the peak level noise at several locations in the city and also showed a fuel savings of 400-500 lb [1]. However, widespread implementation of these procedures has been limited by the capabilities of both air traffic controllers and air traffic control (ATC) automation. For example, because it is difficult to predict the future position of an aircraft when its speed varies significantly, air traffic controllers typically instruct all aircraft to fly a staged approach, where at each stage the aircraft maintain a common altitude and speed. Whereas this greatly reduces the complexity of the tasks the controllers must perform, it also limits the options available to procedure designers. Most of the flight tests were conducted at very low traffic times to ease the controllers separation surveillance effort.

Chapter 3

Review of the Literature

Proposed in 1931, the Zermelo's Navigation Problem was the first posed optimal control problem posed [6]. The problem was to find an an optimal path for a boat navigating in a water body in presence of water curents and wind. Without considering any current or wind or any such external force, the optimal control is to follow a straight line segment from origin to destination. But otherwise, the optimal path is in general never the line joining origin and destination. The same problem was formulated for an aircraft and calculus of variation aproach was used to solve it assuming a flat earth. Bryson and Ho. later developed a solution techinque called neighbouring optimal control (NOC) to come up with a solution for Zermelo's problem [7]. The technique of neighboring optimal control (NOC) produces time-varying feedback control that minimizes a performance index to second order for perturbations from a nominal optimal path. This technique was later extended to handle cases of parameter change in the system dynamic model [8]. This extension is used to develop an algorithm for optimizing horizontal aircraft trajectories in general wind fields using time-varying linear feedback gains. The minimum-time problem for an airplane traveling horizontally between two points in a variable wind field (a type of Zermelo problem) was used to illustrate the above technique. The NOC solution was derived analytically for the case where the wind field was modeled as a constant wind shear in the cross-track direction.

Jardin and Bryson further extended the neighboring optimal control (NOC) technique for computing minimum-time paths through general wind fields by modeling winds along a nominally straight-line path as additional system states [9]. This advancement, referred to as Neighboring Optimal Wind Routing (NOWR), allowed the neighboring optimal control gain solution to be parameterized for different wind conditions and different origin/destination pairs. The winds were modeled at an arbitrary number of discrete points along the nominal great-circle route so that gains are computed for the wind perturbations at each point. Gains are computed once offline and then applied to a wide variety of trajectories between different locations at different altitudes and at different flight speeds. Jardin further demonstrated how to apply the solution to flights on the sphere through coordinate rotations and normalizations and presented analytical solutions for the neighboring optimal gains. In 2010, Jardin and Bryson described two methods to solve a minimum time flight path at high altitude in presence of strong horizontal winds [10]. The first method was using nonlinear feedback (dynamic programming) solutions for minimum-time flight paths. A Zermelo Problem for arbitrary winds was extended from a flat earth model to a spherical earth model as a two-state problem (latitude and longitude) with one control (heading angle). The second method is based on an analytical neighboring optimal control solution that computes neighboring optimal heading commands as a function of the winds along a nominal flight path.

Most of the work mentioned above deals with the cruise portion of the flight. Some of them even assume constant speed throughout the flight. The cruise flight being the major portion of a flight, this has indeed been research topic with most of the works done in flight trajectory optimization, beginning with a series of papers by

Zagalsky et al.,[11] Schultz and Zagalsky [12], Speyer [13] and [14], and Schultz [15]. In Ref. [11] the authors examined the long-range optimal aircraft cruise problem with the energy-range model. Speyer in 1976 [14] proved using second-order variational analysis and a frequency-domain version of the classical Jacobi (conjugate point) optimality condition that the steady-state cruise for a long time span is non-optimal with respect to fuel economy. In 1989 P.K.Menon [16] analyzed the long range cruise problem using point mass and an energy model and showed that the steady state cruise exists as central member along with several other oscillatory extremals. There has been a constant mention in the literature about fuel efficiency of periodic flights - [17], [18] and [19].

Although not as significant as the cruise portion, the climb and the descent portion of the flight has also been studied and optimal strategies were proposed. In 1975, A. Chakravarty in 1983 introduced the concept of an optimal cruise descent [20]. Optimal results were compared with the conventional strategies of constant Mach, V_{cas} and flight path angle descent segments. The effects of wind on cost of delay was also discussed. In [21], representative minimum-fuel flight paths of various types are computed for a commercial jet transport close to the terminal area. [22] studies the characteristics of optimum fixed-range trajectories whose structure is constrained to climb, steady cruise, and descent segments by using optimal control theory.

In 2002, Clarke et al. designed and flight-tested a Continuous Descent Approach (CDA or also known as OPD - optimized descent approach) procedure for UPS-operated Boeing B767-300 aircraft at the end of the nightly UPS arrival bank at Louisville International Airport [23]. This was mainly designed as a noise abatement procedure and it was shown to reduce the A-weighted peak noise level at seven locations along the flight path by 3.9 to 6.5 dBA. The CDA procedure was also shown to reduce the flight time in the terminal area of the Boeing B767-300 aircraft used in the test by up to 100 seconds relative to the nominal approach procedure, and the corresponding fuel burn by up to 500 pounds [23]. However, widespread implementation of CDA has been limited by the capabilities of both air traffic controllers and air traffic control (ATC) automation. Because it is difficult to predict the future position of an aircraft when its speed varies significantly, air traffic controllers typically instruct all aircraft to fly a staged approach, where at each stage the aircraft maintain a common altitude and speed. A lot of variations of CDA have also been studied. A tailored arrival was designed to accommodate CDA under constrained airspace [24]. A tailored arrival creates a four dimensional continuous descent from cruise altitude to the runway. Demonstrations of oceanic arrivals at San Francisco (SFO) have successfully demonstrated significant fuel savings. Since 2002, significant research has gone into studying practical implementations of CDA, its variations and comparisons with current procedures and its efficiency - [25], [26], [27], [28], [29] and [30].

Most trajectory optimization schemes use calculus of variation or optimal control theory which are continuous time methods. Discrete methods were also used as early as 1950's. Dixon Speas formed a small company to serve clients in the airline industry []. One of his services was to plan minimum-time paths for flights over the Atlantic Ocean. His engineers used discrete dynamic programming, dividing the path into 15 to 20 regions and using high altitude wind data from weather balloons. In the 1970s, Lou Reinkins at Lockheed started a flight planning service for airlines and private aircraft for flights in the United States. Starting in the 1980s, Jeppesen JetPlan did the same thing for the airlines and private pilots and included international flights. Due to large runtime and memory management issues, discrete search strategies have seldom been used to plan aircraft trajectories. Discrete algorithms have mainly been used in robotics and UAV path planning in presence of obstacles. In [31], Sellier discusses the use of discrete search methods for real time flight path optimization. It also presents discrepancies and inefficiencies of the cost index concept which is still currently in use in the most advanced flight management systems. Mixed integer linear programs have also been used to solve for real time trajectory planning for UAV's [32] and for trajectory planning with collision avoidance [33]. Iris Yang and Yiyuan in [34] present a discrete search strategy potential real-time generations of four-dimensional trajectories for a single autonomous aerospace vehicle amid known obstacles and conflicts.

There are surplus examples to show the popularity of use of discrete methods like Dijkstra's algorithm or dynamic programming in the context of UAV's. But they have rarely been used to plan commercial aircraft trajectories. In this work, we plan to show the effectiveness and flexibility of A* algorithm in incorporating the large number of trajectory constraints placed on a commercial aircraft by the air traffic regulations.

Chapter 4

Equations of Motion

4.1 Three Dimensional point mass model

Point mass equations of motion on a flat non-rotating earth are typically adequate to model subsonic commercial flight for trajectory planning purposes. A constant gravity model is assumed along with atmosphere being modelled using International Standard Atmosphere. In addition, the aircraft is assumed to fly in an atmospheric wind field comprising of all three components that are functions of x, y, h cooridnates. This paper discusses trajectory planning for three different cases: a full three dimensional flight plan, two dimensional horizontal flight profile and a two dimensional vertical flight profile. Equations of motion for a three dimensional point mass model are listed below [35]. Aircraft states is defined by airspeed V, heading angle ψ, air relative flight path angle γ, mass of the aircraft, mass of the aricraft m and (location) x, y and h coordinates. The control variables used here are Thrust T, bank angle μ and lift coefficient C_L.

$$\bar{X} = [V \;\; \psi \;\; \gamma \;\; x \;\; y \;\; h \;\; m]^T \tag{4.1}$$

$$u = [T \ \ \phi \ \ C_L]^T \tag{4.2}$$

In reality, there is a dynamics involved with how the control parameters varies and they have a response time typically on the order of a few seconds. Since we are looking at commercial flight trajectories which last on the order of hours, it is safe to assume that the control variables can be changed instantenously and hence we neglect any dynamics involved with control variables.

$$m\dot{V} = T \ - D - mg\sin\gamma - m\dot{W}_x\cos\gamma\sin\psi - m\dot{W}_y\cos\gamma\cos\psi - m\dot{W}_h\sin\gamma \tag{4.3}$$

$$mV\cos\gamma\dot{\psi} = L\sin\phi - m\dot{W}_x\cos\psi + m\dot{W}_y\sin\psi \tag{4.4}$$

$$mV\dot{\gamma} = L\cos\phi - mg\cos\gamma + m\dot{W}_x\sin\gamma\sin\psi + m\dot{W}_y\sin\gamma\cos\psi - m\dot{W}_h\cos\gamma \tag{4.5}$$

$$\dot{x} = V\cos\gamma\sin\psi + W_x \tag{4.6}$$

$$\dot{y} = V\cos\gamma\cos\psi + W_y \tag{4.7}$$

$$\dot{h} = V \sin\gamma + W_h \tag{4.8}$$

Where $\dot{W}_x, \dot{W}_y, \dot{W}_h$ are the wind acceleration term along the XYH axes.

4.2 Vertical profile: point mass model

The above equations capture full 3-Dimensional point mass model for an aircraft. Starting from these set of equations we can arrive at the simplified 2-Dimensional vertical and horizontal flight equations. The following set of equations capture vertical flight trajectory where the lateral route is given. Aircraft state is defined by airspeed V and air relative flight path angle γ. Location of an aircraft is specified by its distance from destination s and the altitude h. The controls are speed V and the flight path angle γ.

$$m\dot{V} = T \; - D - mg\sin\gamma - m\dot{W}_s\cos\gamma - m\dot{W}_h\sin\gamma \tag{4.9}$$

$$mV\cos\gamma = L - mg\cos\gamma + m\dot{W}_s\sin\gamma - m\dot{W}_h\cos\gamma \tag{4.10}$$

$$\dot{s} = V\cos\gamma + W_s \tag{4.11}$$

$$\dot{h} = V \sin\gamma + W_h \tag{4.12}$$

4.3 Two Dimensional horizontal flight: point mass model

Assuming a horizontal level flight at a certain altitude, the following set of equations represent a 2-Dimensional horizontal flight. Aircraft state is defined by airspeed V and heading angle ψ with controls as V and bank angle ϕ. Location of an aircraft is specified by its x and y coordinates.

$$m\dot{V} = T - D - m\dot{W}_x \sin\psi - m\dot{W}_y \cos\psi \tag{4.13}$$

$$mV\dot{\psi} = L\sin\phi - m\dot{W}_x \cos\psi + m\dot{W}_y \sin\psi \tag{4.14}$$

$$L\cos\phi = mg \tag{4.15}$$

$$\dot{x} = V\sin\psi + W_x \tag{4.16}$$

$$\dot{y} = V\cos\psi + W_y \tag{4.17}$$

The rate at which fuel burnt is assumed to be proportional to the thrust specific fuel consumption (tsfc) of the engine. And the C_L and C_D values are calculated for a level flight using the lift, drag, and Mach equations:

$$\dot{m}_f = -\frac{tsfc}{g}T \tag{4.18}$$

$$L = \frac{1}{2}\rho V^2 S C_L \tag{4.19}$$

$$D = \frac{1}{2}\rho V^2 S(C_{D_0} + K C_L^2) \tag{4.20}$$

$$M = \frac{V}{a} \tag{4.21}$$

$$C_{D_0} = C_{D_0}(M) \tag{4.22}$$

$$K = K(M) \tag{4.23}$$

The performance index or the cost function that is being optimized is chosen to be a linear combination of time and mass of fuel spent.

$$Cost = C_t t + C_f m_f \tag{4.24}$$

Air Traffic Controller and pilots often use variables such as Mach number, calibrated airspeed and pressure altitude to specify flight trajectories. The expression for calibrated airspeed is given next:

$$V_{cas} = \sqrt{5}a_{sl}\left[\left(\frac{p}{p_{sl}}\left[\left(\frac{V^2}{5a^2}+1\right)^{7/2}-1\right]+1\right)^{2/7}-1\right]^{1/2} \tag{4.25}$$

Chapter 5

Problem Statement and Solution Strategy

5.1 Problem Statement

The flight planning problem is to determine the flight trajectory variables from an initially specified state to a final specified state while satisfying all the constraints and minimizing the performance index.

As mentioned earlier, constraints can be of various types. There are interior path constraints, which specifies one or more waypoints the aircraft has to flyover. While the location of the waypoint is specified, the state may or may not be. Additional constraints also include certain unusable airspace, like a no fly zone or region of high turbulence or bad weather, which is to be avoided. Air traffic control has certain procedures to be followed for take off and landing which are broken into segments with one or more state variables being constant and have certain capture conditions (to determine when the switch to next segment is made) which were discussed in a previous section. And at last we have restrictions on certain state variables (or rate of change of state variables) which account for the passenger comfort level for a commercial flight. These constraints are mathematically formulated for specific cases in the numerical simulations section.

While solving such a problem, the solution is expected to be numerically efficient & flexible and of course satisfies all the given constraints. A methodology is needed which can solve a wide range of problems which includes take off and landing procedure's.

5.2 Optimization Technique

The trajectory optimization problem can be solved using continuous or discrete methods. There are four general methods to solve an optimization problem.

- **Parametric optimization** involves minimization of a certain performance index which is a function of certain parameters subject to certain constraints. These parameters can take on any real values or integers or a combination of both. Solutions techniques are well known and include approaches like combinatorial optimization, linear and non-linear programming and mixed integer programing.

- **Continuous time optimal control problem** deals with problem with minimization of performance indexes which are functionals of state and control variables constrained by certain dynamic equations and other constraints. Calculus of variation, Pontryagin's minimum principle and dynamic programming can be used to describe theoretical necessary and sufficient conditions for a minimum to exist which lead to a bunch of ordinary or partial differential equations which can be solved to obtain a solution. Such problems can also be converted to a parameter optimization problem by the use of methods like collocation technique and pseudo spectral methods.

- **Discrete time optimal control problem** seeks to determine a sequence of optimal control values at certain discrete times by minimizing a performance index. Control values to be solved for are finite in number. Although this is a discrete time framework, the states can assume continuous values at these discrete time values.

- **Discrete search technique** poses the problem of optimization as a discrete time and discrete state problem, meaning, the states can assume certain discrete values only which are finite in number. This essentially converts the problem to a shortest path search problem which can be solved using graph search methods like Dijkstra's algorithm, A* and Dynamic programming.

In general, in a continuous framework, the system state values can take infinitely possible values and hence the trajectory generated is smooth and continuous. But an optimal solution obtained may not be a globally optimal and could depend on the initial guess. Convergence of such a solution is not guaranteed and its complexity only increases with more constraints.

A discrete search method by its very nature is suboptimal. But within the framework of discrete search, the solution obtained is globally optimal. The solution obtained will not be smooth and hence may be not directly flyable. In addition the performance index has to satisfy the property of being additive. But such a method has its own significant advantages which is exploited in this paper. A discrete method described in rough words is an optimized brute force search for an optimal trajectory over a large, but finite number of possibilities. All the states are discretized and take only certain values. The search for a optimal trajectory is over a finite number of system variables, and hence it guarantees the completeness of the search. In other words, if the method does not find a solution then it can be concluded that there does not exist one. There exists a theoretical guarantee of convergence to an optimal solution. But the most important aspect of a discrete framework is that it has the capability of implementing a large numbers of constraints quite easily. In fact, larger the number of constraints lesser is the number of possible solutions and hence easier the problem becomes to solve.

For the problem considered here, the solution strategy is expected to handle all kinds of constraints and should be able to incorporate all of the restrictions imposed by air traffic control. Keeping this in mind, we chose attempt to solve this problem by using a discrete search technique.

5.3 Shortest path routing problem

A graph refers to a collection of vertices (or nodes) and a collection of edges that connect pairs of vertices. Graphs are often used to model networks in which one travels from one point to another. As a result, a basic algorithm problem is to determine the shortest path between nodes in a graph. This is more popularly known as shortest path routing problem. Formally put, the problem of shortest path routing problem is the problem of finding a path between two vertices (or nodes) such that the sum of weights of its constituent edges is minimized. It is also called single pair-shortest path problem, as opposed to single-source shortest path problem, single-destination shortest path problem and all-pairs shortest path problem, which deal with multiple source or destinations.

Graphs are simple to define, we just take a collection of things and join them by edges. The maps of routes served by airline carrier naturally form a graph: airports are the nodes and there is edge between two airports if there exists a direct flight between the tow airports. A rail or road network, collection of computers connected via a communication network, the world wide web and a social network are few of the many examples which can also be modeled as a graph. In all of the examples stated so far the the nodes and edges physically exist. A shortest path between a given source-destination pair along a road network can be physically modeled as a graph by using any intersection of roads as nodes and roads themselves as edges along which we travel. Shortest path algorithms have also applied in aerospace engineering, specially finding optimal paths in a known terrain with obstacles for unmanned aerial vehicles [34].

The theory and solutions for shortest path problem is quite old and is very mature. The well known algorithms for solving this problem are:

- Dijkstra's algorithm: solves the single-pair, single-source, and single-destination shortest path problems.
- Bellman-Ford algorithm: solves the single source problem if edge weights may be negative.
- A* search algorithm: solves for single pair shortest path using heuristics to try to speed up the search.
- Floyd-Warshall algorithm: solves all pairs shortest paths.
- Johnson's algorithm: solves all pairs shortest paths, and may be faster than Floyd-Warshall on sparse graphs.
- Perturbation theory: finds (at worst) the locally shortest path.

The problem under consideration here is to find an optimal path between two specific nodes for an aircraft, which is a single-pair shortest path problem with non-negative edge weights. For such a formulation, A* search algorithm seems the best applicable method. To understand how an A* algorithm works, we first explain the working of Dijkstra's algorithm and extend that to A*.

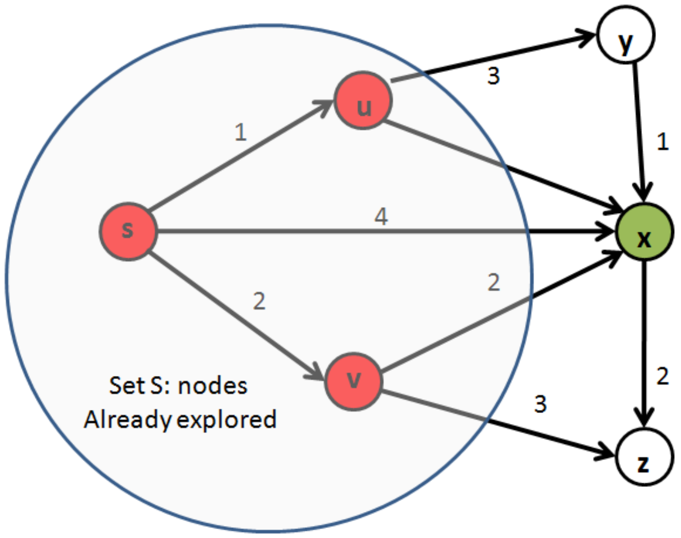


Figure 5.1: Snapshot of execution of Dijkstra's algorithm

5.3.1 Dijkstra's Algorithm

A simple greedy algorithm to solve single-source shortest-paths problem called Dijkstra's algorithm was conceived by a Dutch computer scientist Edsger Dijkstra in 1956 and was first published in 1959 [36]. For a given source vertex in the graph, the algorithm finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined. For example, if the vertices of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and all other cities.

Dijkstras algorithm works by visiting vertices in the graph starting with the objects starting point. It then repeatedly examines the closest not-yet-examined vertex, adding its vertices to the set of vertices to be examined. it expands outwards from the starting point until it reaches the goal. Dijkstras algorithm is guaranteed to find

```
Dijkstra's Algorithm $(G, l)$

Let $S$ be the set of explored nodes
  For each $u \in S$, store a distance $d(u)$
Initially $S = \{s\}$ and $d(s) = 0$
While $S \neq V$
  Select a node $v \in S$ with at least one edge from $S$ for which
    $d^{'}(v) = min_{e=(u,v):\, u \in S}\, d(u) + l_e$ is as small as possible
  Add $v$ to $S$ and define $d(v) = d^{'}(v)$
End While
```

Table 5.1: Dijkstra's Algorithm pseudo code

a shortest path from the starting point to the goal, as long as none of the edges have a negative cost. In the following diagram, the pink square is the starting point, the blue square is the goal, and the teal areas show what areas Dijkstras algorithm scanned. The lightest teal areas are those farthest from the starting point, and thus form the frontier of exploration Figure 5.2.

The pseudo code for Dijkstra's algorithm is given in table [37]. The algorithm maintains a set S of vertices u for which the shortest path from the source s has been determined; which is the 'explored' part of the graph. Initially $S = \{s\}$, and $d(s) = 0$ (d being the cost function). Now for every node $v \in V - S$, the shortest path is determined that can be constructed by traveling along a path through the explored part S to some $u \in S$, followed by the single edge (u, v). That is, a node $v \in V - S$ is chosen for which the quantity

$$d^{'}(v) = min_{e=(u,v):\, u \in S}\, d(u) + l_e \tag{5.1}$$

is minimized, and $d(v)$ is defined to be $d^{'}(v)$.

5.3.2 Performance and runtime

An upper bound of the running time of Dijkstra's algorithm on a graph with edges E and vertices V can be expressed as a function of $|E|$ and $|V|$ using Big-O notation. For any implementation of set Q the running time is $O(|E|.dk_Q+|V|.em_Q)$, where dk_Q and em_Q are times needed to perform decrease key and extract minimum operations in set Q, respectively. The simplest implementation of the Dijkstra's algorithm stores vertices of set Q in an ordinary linked list or array, and extract minimum from Q is simply a linear search through all vertices in Q. In this case, the running time is $O(|V|^2 + |E|) = O(|V|^2)$. For sparse graphs, that is, graphs with far fewer than $O(|V|^2)$ edges, Dijkstra's algorithm can be implemented more efficiently by storing the graph in the form of adjacency lists and using a binary heap, pairing heap, or Fibonacci heap as a priority queue to implement extracting minimum efficiently. With a binary heap, the algorithm requires $O((|E|+|V|)\log|V|)$ time (which is dominated by $O(|E|\log|V|)$, assuming the graph is connected), and the Fibonacci heap improves this to $O(|E| + |V|\log|V|)$.

5.3.3 A* shortest path algorithm

A* is chosen as the discrete search method we would use to solve trajectory optimization problem. A* is a shortest path graph search algorithm. It works in a very similar way as Dijkstra's algorithm but in addition, what makes A* an optimized search is that it uses a heuristic cost function to guide itself to the goal. The actual cost associated with a given node, say N_c, is the cost of moving from source to this node and the heuristic is an approximate estimate of moving from node N_c to the goal. The successor point is chosen by minimizing the sum of the actual cost plus the heuristic cost, which reduces the burden of searching through unnecessary nodes. The heuristic can be used to control the behavior of A*. If the heuristic is set to be equal to zero the A* algorithm behaves exactly like Dijkstra's algorithm and essentially ends up searching through all possible points before reaching the goal (in essence, Dijkstra's algorithm is the worst case runtime scenario for A* implementation). If the heuristic is less than or equal to the optimal cost of moving from current node to goal then it searches through less number of nodes before reaching the goal. If it is exactly equal to the optimal cost then the algorithm searches through only the nodes along the optimal path. Hence, the closer the heuristic is to the optimal cost the faster the algorithm is. Care should taken that heuristic should not overestimate the optimal cost or else the resulting trajectory might turn out be a suboptimal path. If, $g(N)$ defined as the cost of traveling from source to the node N and $h(N)$ is the heuristic estimated cost of traveling from node n to goal, each time through the main loop, A* examines the vertex n that has the lowest $f(N) = g(N) + h(N)$. The proper choice of a heuristic function can effectively reduce the amount of work the algorithm does and critically effects the convergence rate. The heuristic is so chosen such that it does not overestimate the optimal cost and should be zero at the goal. The time term of the heuristic is chosen as the shortest time to reach goal from the current location, which is flying and along the line joining current node and the goal at maximum speed. No Heuristic for fuel was used and was assumed to be zero. More complicated Heuristics can be modeled to improve the runtime of the algorithm. It is to be noted that as long as the heuristic does not overestimate the optimal cost the choice of heuristic does not affect the solution generated, it only affects the runtime of the algorithm. Figure 5.3 shows the search space of A* while solving the same problem which was solved by using Dijkstra's algorithm (Figure 5.2). It can be clearly seen that A* searches over much fewer nodes than Dijkstra's making it an optimized search.

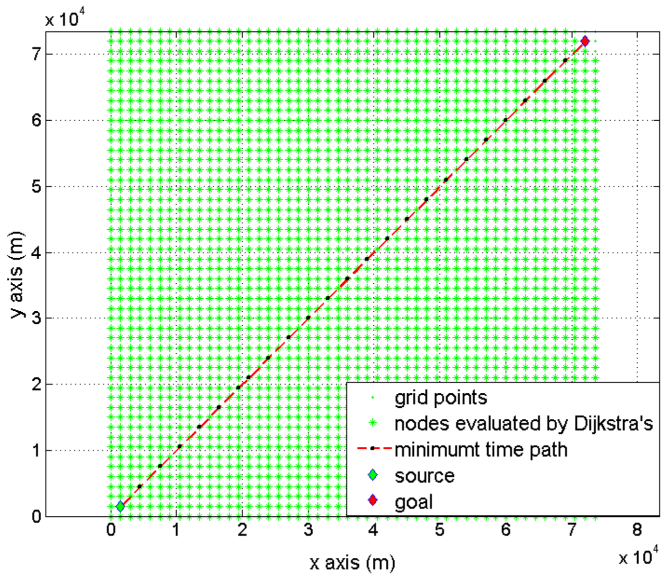


Figure 5.2: Working of Dijkstra's algorithm

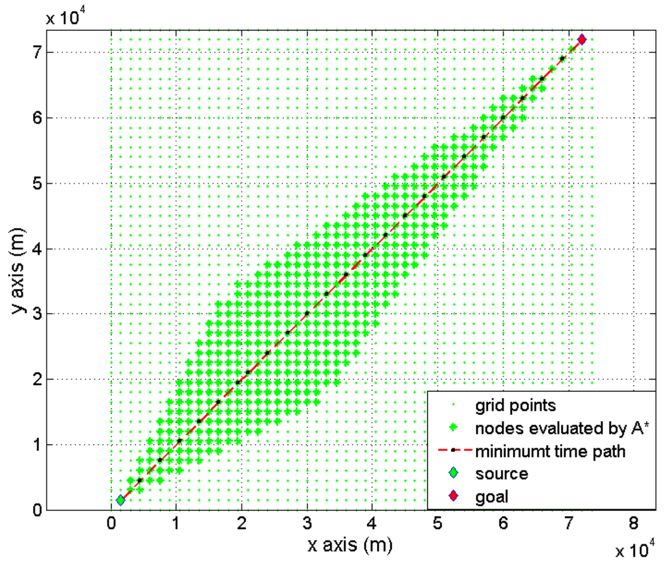


Figure 5.3: Working of A* algorithm

5.4 A* applied to flight trajectory optimization

When trying to find a shortest path for an aircraft between two given points in space, the first step is to build a graph. Consider a simple case where a robot moves along the ground in a two dimensional space. The only states of the system is its x and y coordinates. Therefore to build a graph, the $x-y$ space is broken into discrete values by forming a grid. Any possible combination to x and y forms a node or vertex of the graph.

The three dimensional space is divided into uniform or non-uniform grids. Every grid point, modeled as a node in the graph, specifies a certain state value. Set of neighboring nodes of node in space is defined as a reachable set of that node, which consist of all the immediate neighbor nodes. Edges of the graph are defined between every node and its neighbors.

On similar lines, to find an optimal trajectory for an aircraft between two specified locations, the states are divided into discrete points and modeled as a graph with nodes and edges. A basic x, y, h coordinate system is used where y axis points to the north and x axis points towards the east. The three other aircraft states are V (or equivalently M), ψ & γ representing the velocity (or Mach number), heading and flight path angle respectively (heading angle is measured clockwise from the north). The aircraft state is broken into discrete values, forming a six dimensional grid system. Which means, every node in space is associated with a specific x, y and h coordinate and aircraft state M, ψ and γ at that point (Note that two nodes can have same location in the space but are different in the sense that they are associated with different states).

$$x = x_i \quad i \in [1, N_x] \tag{5.2}$$

$$y = y_j \quad j \in [1, N_y] \tag{5.3}$$

$$h = h_k \quad k \in [1, N_h] \tag{5.4}$$

$$M = M_l \quad l \in [1, N_M] \tag{5.5}$$

$$\psi = \psi_m \quad m \in [1, N_\psi] \tag{5.6}$$

$$\gamma = \gamma_n \quad n \in [1, N_\gamma] \tag{5.7}$$

If N represents a node then:

$$N_{ijklmn,p} = (x_{ijklmn,p}, y_{ijklmn,p}, h_{ijklmn,p}, M_{ijk,l}, \psi_{ijklmn,p}, \gamma_{ijklmn,p} : t_{ijklmn,p}) \tag{5.8}$$

where: $t_{ijklmn,p}$ is minimum time of arrival at that node.

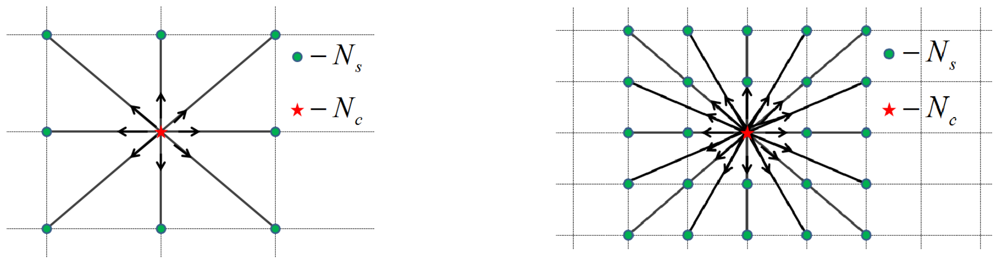
5.4.1 Generation of successor nodes

To allow for sufficient freedom in trajectory generations, successor points are chosen from neighboring grid points in all directions. If $N^c_{ijklmn,p}$ represents the current node, then the set of neighboring points with depth n_d around $N^c_{ijklmn,p}$ define the immediate reachable set R of node N:

$$R[N^s_{ijklmn,p}] = [N_{i'j'k'l'm'n',p'} : -n_d \leq (i-i'), (j-j'), (k-k') \leq n_d] \tag{5.9}$$

$$n(R) = \left[(2n_d+1)^3 - 1\right] N_M N_\psi N_\gamma \tag{5.10}$$

where n_d (named as search depth) is an integer parameter that defines the range of the neighborhood of current node and Equation (5.10) gives the number of elements in this set for a three dimensional case. From the set of neighboring nodes around the current node, successor nodes are chosen such that they are outside any region of avoidance, line segments joining the current node and successor node are not in region of avoidance and also satisfy all the constraints of trajectory and aircraft dynamics. The factor n_d along with the grid sizes Δx, Δy and Δh determines the discrete values of heading and flight path angle the aircraft can take Figure 5.4. As an example, let $n_d = 1$, and $\Delta x = \Delta y$ for a horizontal flight problem. Heading angle varies from zero to 2π (measured clockwise from north) and for $n_d = 1$, it is easy to see that the set R has eight nodes (figure Figure 5.4a). Hence set of discrete values of heading angle has the elements defined in equation (5.11). Similarly if $n_d = 2$ (figure Figure 5.4b),the



(a) Reachable set R for $n_d = 1$: $n(R) = 8$ (b) Reachable set R for $n_d = 2$: $n(R) = 24$

Figure 5.4: Spatially reachable set of nodes N_s from the current node N_c for a two-dimensional case

reachable set has twenty four nodes in it and heading angle set is given in equation (5.12). It is easy to understand that number of elements in the set of reachable nodes R increases by factor proportional to n_d^3 for a three dimensional case. Although a higher value of n_d gives more finer divisions of ψ and hence more accuracy, it increases the runtime of the algorithm quite significantly.

$$n_d = 1: \quad \psi_m \in \{\frac{m\pi}{4} \;\forall\; m = [0,7]\} \tag{5.11}$$

$$n_d = 2; \quad \psi_m \in \{\frac{m\pi}{8} \;\forall\; m = [0,15]\} \tag{5.12}$$

5.4.2 Calculating cost between two nodes

To calculate the weight of an edge is to find the cost of moving from completely specified state to another. Since the initial and final aircraft states are completely known, it becomes a simple two point boundary value problem. If the discrete values of state are very finely divided, to make the calculation slightly easier, it is assumed that the states vary linearly with distance between the nodes and time of arrival at a node is calculated by integrating inverse of ground speed along the distance.

The total velocity (ground speed):

$$\bar{V} = \frac{ds}{dt} = (V\cos\gamma\sin\psi + W_x)\hat{i} + (V\cos\gamma\cos\psi + W_y)\hat{j} + (V\sin\gamma + W_h)\hat{k} \quad (5.13)$$

Time taken to move a distance L:

$$\Delta t = \int_0^L \frac{1}{\sqrt{(V\cos\gamma\sin\psi + W_x)^2 + (V\cos\gamma\cos\psi + W_y)^2 + (V\sin\gamma + W_h)^2}}\, ds \quad (5.14)$$

Using the assumption that states and wind vary linearly as a function of distance between nodes, Δt is calculated using a simple Euler integration method.The second term of the cost function is the fuel consumption and is calculated using equation (4.18), where the thrust specific fuel consumption is assumed to be constant for the specified engine. The thrust required is calculated by using the state dynamic equation (4.3). The cost function is then implemented as a combination of time and fuel, where weights can be adjusted to shift the emphasis between time to fuel.

An error that incurs in such an approximation is explained as follows. Consider a two dimensional level flight where an aircraft is flying from current node $N_c = [x_c, y_c, V_c, \psi_c]$, to a successor node $N_s = [x_s, y_s, V_s, \psi_s]$ as shown in the figure Figure 5.5. If we were to assume that the state ψ and V varies linearly with distance between nodes and since the aircraft's initial heading is not directly towards N_s, does not follow the line joining N_c and N_s, instead takes a curvilinear path (dotted line in figure Figure 5.5) and ends at $N_{s'}$, where the node $N_{s'} = [x_{s'} \neq x_s, y_{s'} \neq y_s, V_s, \psi_s]$.

The difference between the two coordinates of N_s and $N_{s'}$ might not be too large between two neighboring nodes, but over the course of the entire flight over on order

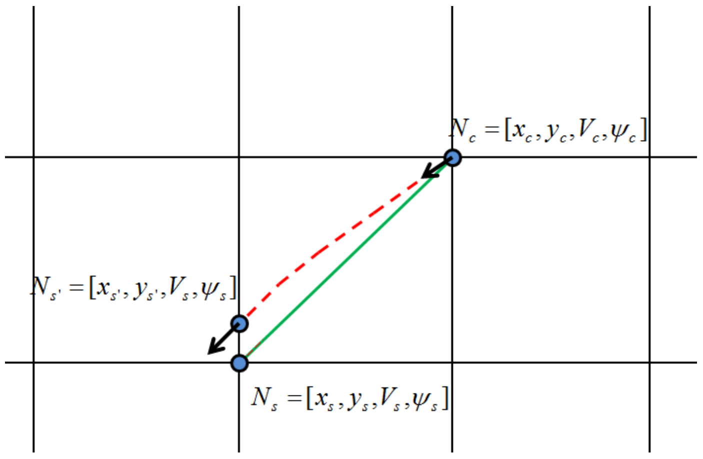

$N_c = [x_c, y_c, V_c, \psi_c]$
$N_{s'} = [x_{s'}, y_{s'}, V_s, \psi_s]$
$N_s = [x_s, y_s, V_s, \psi_s]$

Figure 5.5: Error in position due to approximation

of hundreds of nodes, this error adds up and will end up being significantly large. Although the nodes were initialized at a fixed location, an easy way of getting rid of this error completely is to think of the nodes as not being rigidly located in space. Instead, every node is defined using fixed state (heading and speed) but the coordinates depend on its predecessor node. It means to say once we have calculated $x_{s'}$ and $y_{s'}$, we now assign the node N_s with these new coordinates. So, when we are calculating the successor of N_s we begin at coordinates $x_{s'}$ and $y_{s'}$ instead of x_s and y_s.

5.5 Constraints

5.5.1 Bounds on States

Physically, the control parameters are used to change state values are thrust, bank angle and lift coefficient. But while finding an optimal trajectory problem using A*, we are searching over nodes with fixed state values. So, the bounds of controls need to translated to bounds on states. The aerodynamics of the wing results in a maximum lift coefficient (Equation (5.17)). The maximum and minimum speeds of an aircraft depend on the divergence Mach number and stall limits (safety factor of 20 % taken into account) respectively (equations (5.23), (5.24)). For passenger comforts, the rate of descent and ascent are limited, which translate to a limit on the flight path angle (equation (5.25)). Air Traffic Control regulations specify that the calibrated airspeed for altitudes below 10,000 ft has to be less than 250 knots.

$$T_{min} \leq T \leq T_{max} \tag{5.15}$$

$$-\phi_{max} \leq \phi \leq \phi_{max} \tag{5.16}$$

$$0 \leq C_L \leq C_{L_{max}} \tag{5.17}$$

Equation (5.9) gives us the list of immediate neighboring nodes which are defined as nodes present in the vicinity of the current node. Nodes which do not satisfy the dynamic constraints of the aircraft and other limitations are eliminated from this set. If N_c and N_s are current and successor nodes respectively with,

$$N_c = \{x_c \;\; y_c, \;\; h_c \;\; M_c \;\; \psi_c \;\; \gamma_c\} \tag{5.18}$$

$$N_s = \{x_s \;\; y_s, \;\; h_s \;\; M_s \;\; \psi_s \;\; \gamma_s\} \tag{5.19}$$

then, the successor node is chosen such that N_s satisfies the following criteria:

$$\frac{V_s - V_c}{\Delta s} \leq \frac{1}{mV_c}\left(T_{max_c} - D_c - m_c g \sin\gamma_c\right) \tag{5.20}$$

$$\frac{V_s - V_c}{\Delta s} \geq \frac{1}{mV_c}\left(T_{min_c} - D_c - m_c g \sin\gamma_c\right) \tag{5.21}$$

$$\frac{|\psi_s - \psi_c|}{\Delta s} \leq \frac{1}{mV_c^2 \cos\gamma_c} L \sin\phi_{max} \tag{5.22}$$

$$V_s \geq 1.2 V_{stall} = 1.2\sqrt{\frac{2mg}{\rho S C_{L_{max}}}} \tag{5.23}$$

$$M_s \leq M_{max} \tag{5.24}$$

$$-3^\circ \leq \gamma_s \leq 8^\circ \tag{5.25}$$

$$V_{cas} \leq 250\, knots \quad \forall h \leq 10,000 ft \tag{5.26}$$

5.5.2 Obstacles: No fly zone

Handling a no fly zone or a region of unusable airspace is the easiest to implement in A*. A no fly zone in space can be defined by the use of any geometric shape or a combination of shapes using x, y and h coordinates. Any node or line joining two nodes which lies in this region is marked as an obstacle and is never considered for evaluation by A*, thereby completely avoiding this specified region. This type of implementation reduces the number of nodes to be considered for evaluation by A*, and hence can only improve runtime, thus making it easy and very effective way. Numerical simulations are provided to show the implementation of no fly zones.

5.5.3 ATC regulation

Air traffic control regulations require that any aircraft flying below 10,000 ft has to fly with a calibrated airspeed less than 250 knots. The easiest way of handling this constraint is by marking every node below 10,000ft with calibrated airspeed above 250 knots as an obstacle (or equivalent to an obstacle) and hence A* completely ignores any such node.

5.5.4 Vertical profile segments

Landing and take off procedures defined by air traffic control regulations are broken up into segments in order to emulate manual pilot procedure and create a simple set of rules which can be issued to pilots. Hence a trajectory can be described as a series of segments. Each flight segment can be described by two having a certain state held constant. There are three types of variables which can be held constant: engine control (idle or maximum thrust), speed (V_{cas} or M) and vertical rate (altitude rate or inertial flight path angle). The values of the constant variables are specified to be within constraints of ATC as well as performance limits of the aircraft. As an example, consider the descent profile shown in figure Figure 2.2. It has a sequence of segments beginning with constant altitude, followed by constant Mach number M_d and constant calibrated airspeed V_{cas_d}. The end of every segment is defined by a capture condition. A capture condition occurs when a non-constant variable reaches some specific value. The connecting order of the segments in a trajectory defines the capture variables used. To simplify things, in this thesis, the interval of of occurrence of a segment is specified by distance s. Hence a trajectory profile is specified by sequence of segments which are in turn specified by variable s and the variable to be held constant in that segment.

Given a profile, trajectory optimization means finding the optimal values of capture variables or the value of the variable being held constant in a every segment. The main reason of choosing A* search algorithm to solve trajectory optimization is its ability to handle these types of profile constraints. Incorporating flight profiles in A* is explained as follows: A* is a search algorithm which searches over discrete values of state.

Consider a simple climb profile shown in Figure 6.4. Each segment has a different variable that is held constant (h, M and V_{cas}). The set of variables that define a segment are the ones that are discretized to form vertices or nodes of the graph. For each segment, the variables held constant is the variable which is broken into discrete values. That is, the set of variables that define a constant Mach segment is given by: (s, h, M and γ), whereas the variables that define a constant V_{cas} segment is (s, h, V_{cas} and γ). And within a segment edges are placed only between nodes which have the same value of the variable that defines the segment. This ensures that the solution returned by the shortest path algorithm always follows the profile specified and satisfies all of the segment constrains.

Chapter 6

Numerical Simulations

6.1 Performance Model

A generic commercial aircraft with the following specifications has been used to obtain numerical results.

$$W_0 = 10,000 \ \ lbs \tag{6.1}$$

$$\frac{W_0}{S} = 120 \ \ \frac{lb}{ft^2} \tag{6.2}$$

$$T_{max,sl} = 0.25 W_0 \ \ lbs \tag{6.3}$$

$$T_{max} = \frac{\rho}{\rho_{sl}} T_{max,sl} \ \ lbs \tag{6.4}$$

$$tsfc = 0.725 \ \frac{lb}{lb.hr} \tag{6.5}$$

The Thrust specific fuel consumption is assumed to be independent of altitude and speed. The drag polar is expressed in terms of parasite drag coefficient and induced lift coefficient which in turn as assumed to be functions of Mach number [38] only (equations (6.6)-(6.9)).

$$C_D = C_{D_0}\left(M\right) + K\left(M\right) C_L^2 \tag{6.6}$$

$$C_{D_0} = \begin{cases} 0.0123 & 0 < M < M_1 \\ 0.0123 - 0.0057z^2 + 0.158z^3 - 0.911z^4 + 1.6178z^5 & M_1 < M < M_{max} \end{cases} \tag{6.7}$$

$$K = \begin{cases} 0.06056 & 0 < M < M_1 \\ 0.06056 - 0.0707z^2 + 1.6097z^3 - 4.843z^4 + 6.4154z^5 & M_1 < M < M_{max} \end{cases} \tag{6.8}$$

$$C_{L_{max}} = \begin{cases} 1.8000 & 0 < M < M_1 \\ 1.8000 + 11.7z^2 - 317z^3 + 1432.7z^4 - 2018.8z^5 & M_1 < M < M_{max} \end{cases} \tag{6.9}$$

where $z = M - M_1$ and $M_1 = 0.55$.

Numerous simualtions are performed to show all the different features of the algorithm.

6.2 Horizontal flight planning

All of the level flight simulations were performed on a map with 100 x 100 nodes at an altitude of 8000m. The spacings along east and north directions were 1500m. Airspeed was broken into 30 discrete points between the maximum and minimum values given by (5.23) and (5.24) respectively. The parameter n_d was chosen as 2, which meant there were 16 different heading values (5.12).

6.2.1 A* v/s Dijkstra's algorithm: Reduction in computational time

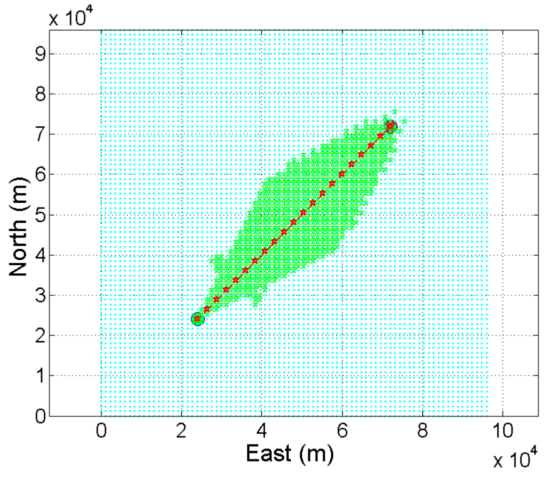
The Figure 6.1 shows the result of level flight optimization using A* and Dijkstra's algorithm. The two dimensional discrete grid points, the POD, POA, and the trajectory are shown. The grid points are shown in light green and the nodes evaluated by the algorithm are shown in darker green. The final solution of both the algorithms are exactly the same but it is easy to note that the number of nodes evaluated by A* (Figure 6.1a) is far less than that by Dijkstra's (Figure 6.1b). This is because Dijkstra's algorithm has no heuristic and hence, it has no sense of direction. It ends up searching almost entire search space until it reaches the destination. A* on the other hand is much faster because it always tries to move towards the goal rather than search through all directions.

There are other effective ways to reduce runtime while continuing to use Dijkstra's algorithm. If we can identify with certainty, a reduced search space for the algorithm to search over, then we avoid visiting nodes outside this search space because the optimal solution is guaranteed to be within that search space. This has been demonstrated in the Figure 6.1c. The search space is now an elliptic region rather than a square. The ellipse is built such that the source and goal nodes are the foci, and the eccentricity of the ellipse is determined by the aircraft turn radius in order to accommodate the initial and final heading constraints. Even though Dijkstra's algorithm does search through almost all of the nodes before terminating, the runtime is still much lesser than that of the case of a square grid (Figure 6.1b).

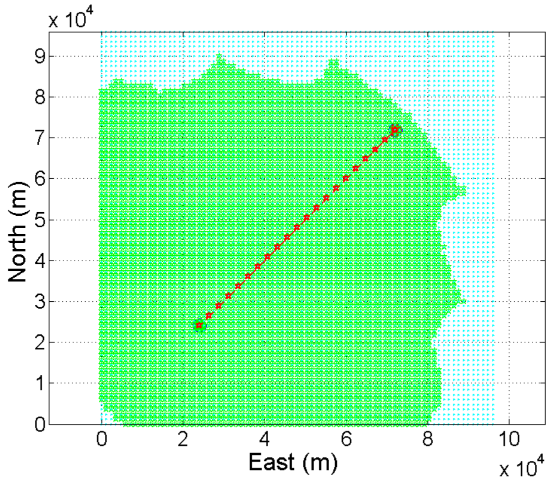
6.2.2 Various Initial and terminal constraints on states

This section of results demonstrates the capability of the algorithm in handling the initial and final constraints on states. Trajectories were found between the same POD and POA for various combinations of initial and final constraints. The accuracy and runtime of the algorithm depends on the grid sizing and selection of parameter search depth n_s. Figure 6.2 shows three trajectories, each with different initial and terminal constraints.

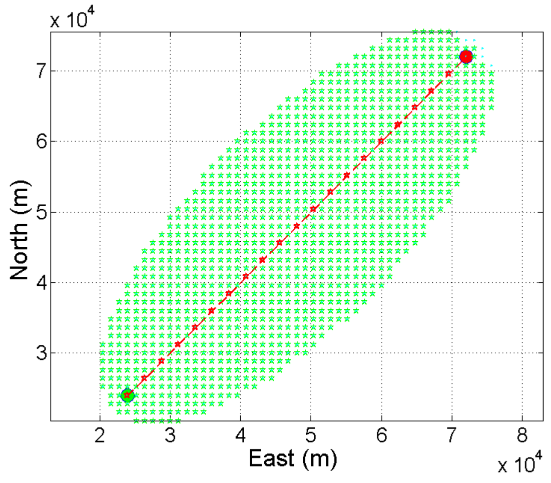
Another important feature of the algorithm is demonstrated in Figure 6.3, which shows the level flight trajectories for various initial and terminal constraints in presence of no-fly zone or a region of avoidance.



(a) A* algorithm search space

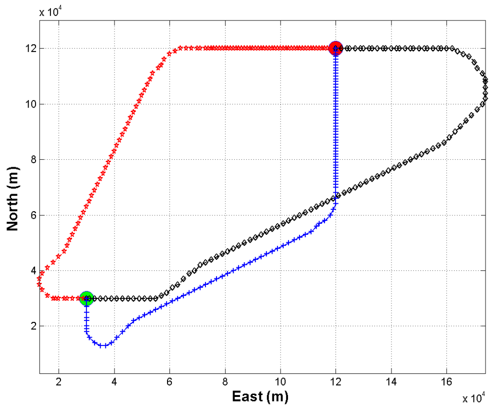


(b) Dijkstra's algorithm search space

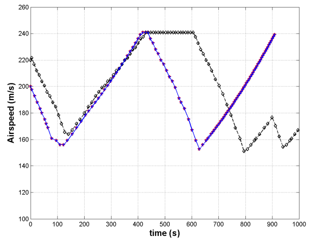


(c) Dijkstra's algorithm using an elliptical search space

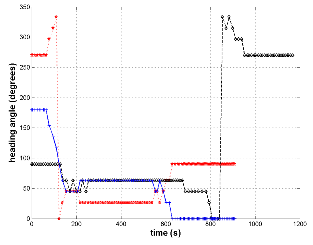
Figure 6.1: Minimum time paths for a horizontal level flight using A*, Dijkstra's: $V_i = 240m/s$, $\psi_i = 45^o$ & $V_f = 240m/s$, $\psi_f = 45^o$



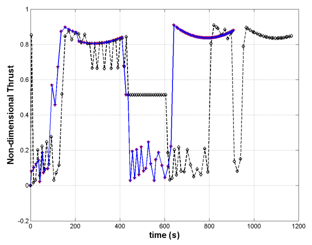
(a) Minimum time path



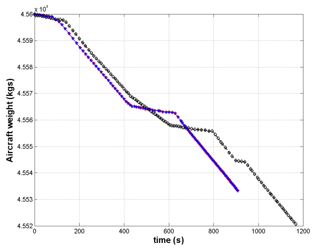
(b) Airspeed



(c) Heading angle



(d) Non-dimensional thrust



(e) Mass of aircraft

Figure 6.2: Minimum time paths for various initial and terminal conditions

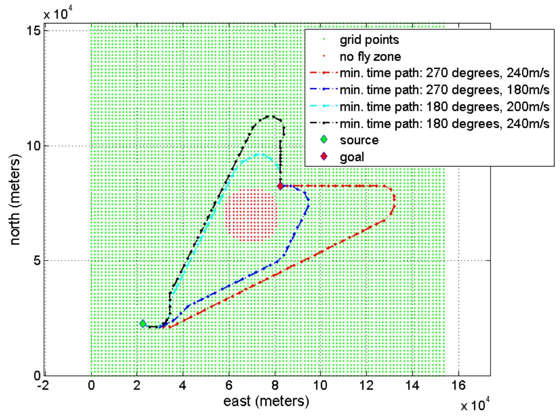


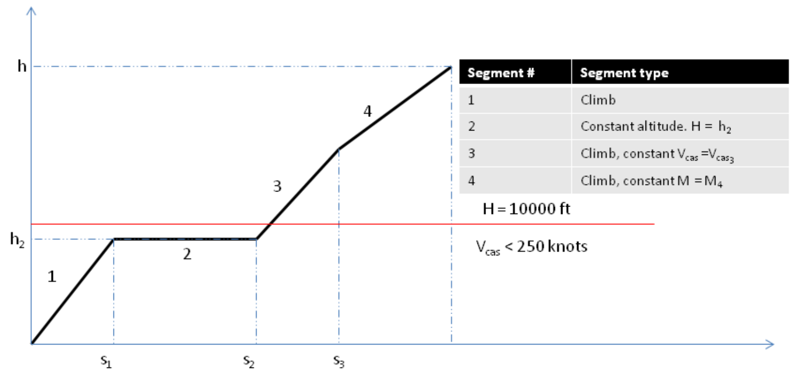
Figure 6.3: Minimum time paths in presence of a no-fly-zone/obstacle

6.3 Vertical profile flight planning

The vertical profile optimization was studied in three phases: climb, descent and full trajectory. Finding an optimal climb or descent profile does not necessarily translate to an optimal trajectory. But the reason it is studied separately is because they ATC regulates most part of the climb and descent of an aircraft (in and out of an airport). These ATC regulations are modeled physically as trajectory constraints, which can be of various types. These are listed below:

- Bounds on trajectory variables
- Initial and terminal constraints
- Avoiding certain airspace or altitude band
- Altitude-speed constraints (ATC regulation limits the speed of an aircraft to less than 250 knots for all altitude below 10000 ft)
- Climb and descent pattern (segmented flight profile)

For each of the phases, we demonstrate the capability of the algorithm in handling various types of constraints. Unless specified otherwise, the discretization of state-



Segment #	Segment type
1	Climb
2	Constant altitude. $H = h_2$
3	Climb, constant $V_{cas} = V_{cas_3}$
4	Climb, constant $M = M_4$

Figure 6.4: A typical climb profile in the air traffic system

space used in all of the vertical profile optimization algorithms are:

$$N_m = 35, \quad \Delta s = 1400, \quad \Delta h = 72, \quad C_t = 0.2, \quad C_f = 2, \quad n_s = 2 \tag{6.10}$$

6.3.1 Climb phase optimization

Here the optimization problem is to find an optimal climb path from ground to a specified altitude (usually the cruise altitude). Three different climb profiles are considered for optimization which are summarized below. The results of optimization of these profiles is shown in Figure 6.7.

- The first one is an unconstrained optimization (shown in red) which means, the ATC does not impose any restrictions on the trajectory of the aircraft.
- The second one shown in black, has an altitude-speed constraint imposed. Practically, this is a standard ATC imposed restriction which limits the calibrated airspeed of an aircraft to 250 knots below 10,000 ft.
- The third one is a typical climb profile followed by the air traffic system in a regulated airspace. It is described in the Figure 6.4. The aircraft here is

constrained to follow a particular specified sequence of segments. That is, the parameters s_1, s_2 and s_3 are specified and the trajectory variables which describe each segment are the parameters of optimization, i.e, γ_1,h_2, V_{cas_3} and M_4. Each segment of the profile has a different trajectory variable held constant and the discretization of the state is done accordingly as explained in an earlier section.

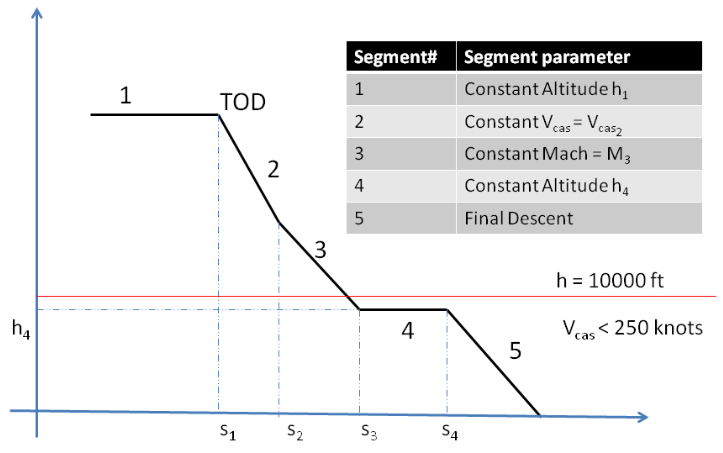
The results clearly show that an unconstrained trajectory is the one which is minimum time and minimum fuel among all the three profiles. Whereas the segmented profile is the one that is worst performer. This goes without saying that the more constraints you place the worse the performance. The optimal parameters that define the segment of this profile are: $h_2 = 3600m$, $V_{cas_3} = 335.3Knots$ and $M_3 = 0.6775$.

6.3.2 Descent phase optimization

Two types of descent profiles are considered for simulation purposes. Figure 6.5 shows a typical descent pattern of an aircraft in the air traffic system. This follows a similar pattern as the climb profile. The second type of descent is completely unconstrained. And as observed in the climb phase, getting rid of the structured descent profile and making an unconstrained descent leads to significant amount of fuel and time savings (Figure 6.8).

6.3.3 Full trajectory optimization

A complete trajectory involves a climb, cruise and descent. Earlier section provides the details of various types of profiles we can have for climb and descent. There are three common practices for a cruise too. These include - constant altitude, cruise climb and step climb. All of them are performed at constant speed. The constant altitude cruise and the step climb cruise are most commonly used in practice. A cruise climb is more fuel efficient but is not used in practice because of ATC regulations (an



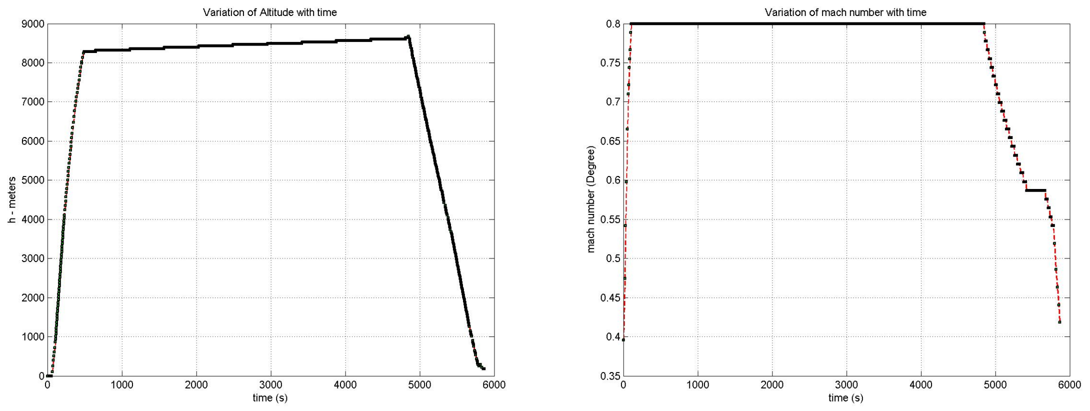
Segment#	Segment parameter
1	Constant Altitude h_1
2	Constant $V_{cas} = V_{cas_2}$
3	Constant Mach = M_3
4	Constant Altitude h_4
5	Final Descent

Figure 6.5: A typical descent profile in the air traffic system

aircraft constantly changing altitude is difficult to keep track of and hence, it harder for the ATC to ensure separation between aircrafts).

As in the case of climb and descent, we optimize three different types of vertical flight profiles for a full trajectory. The first type is a completely unconstrained trajectory optimization. No ATC rules are imposed and the aircraft is free to take and descend without any procedures. The result of the optimization is shown in Figure 6.6. As it can be seen from the figure, the climb and descent occur at a constant rate and the cruise portion is a repeated step climb. This is ideally a cruise climb, but since the optimization technique used here is a discrete state-space version, we see a discrete version of cruise climb.

The next two profiles we consider are more realistic with respect to air traffic control point of view. The constraints imposed on the two profiles are mentioned below:

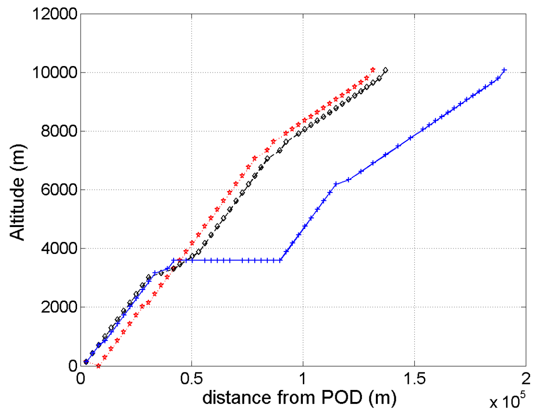


(a) Altitude profile (b) Mach number

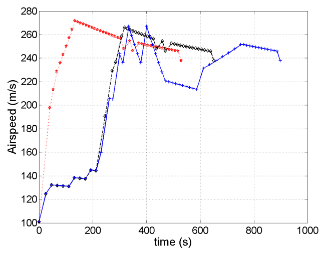
Figure 6.6: Full vertical profile optimization for unconstrained trajectory

- Unconstrained: No ATC regulations imposed.
- Altitude - speed constraint: The calibrated airspeed below altitudes of $10000ft$ has to be less than $250knots$.
- Apart from the above altitude-speed constraint, the climb and descent profile has to follow a certain structure which are similar to what was mentioned in the climb and descent phase analysis. The cruise is restricted to be a constant altitude phase.

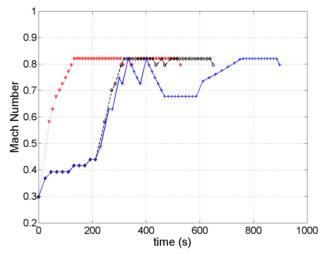
The results of the optimization are shown in Figure 6.9.



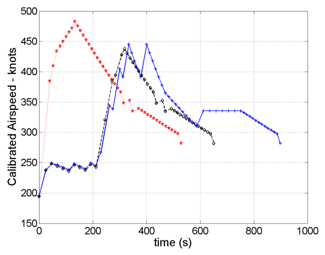
(a) Altitude



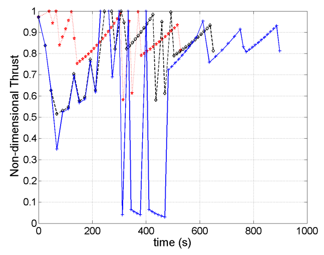
(b) Air speed



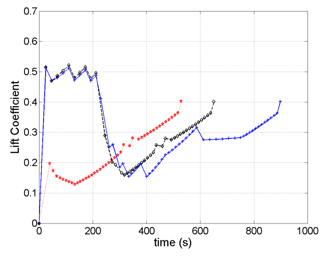
(c) Mach number



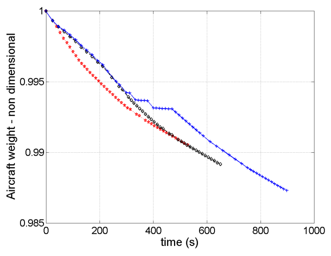
(d) Calibrated air speed



(e) Non dimensional thrust

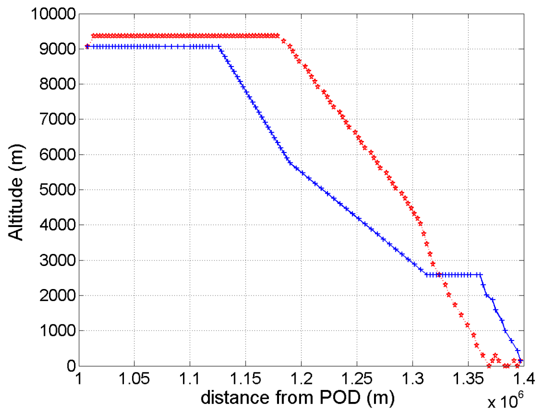


(f) Lift coefficient

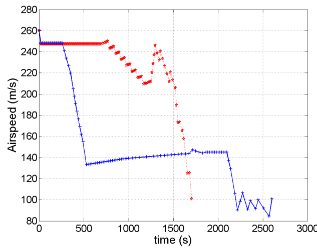


(g) Aircraft mass

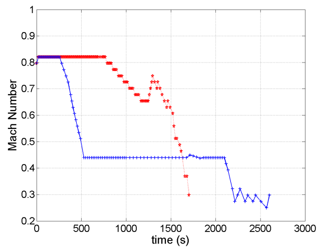
Figure 6.7: optimal climb trajectories



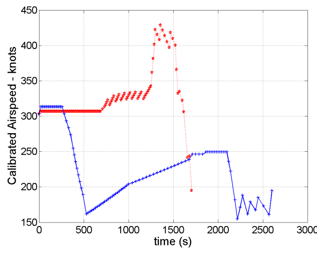
(a) Altitude



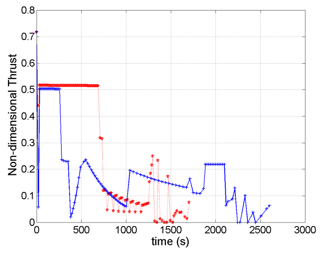
(b) Air speed



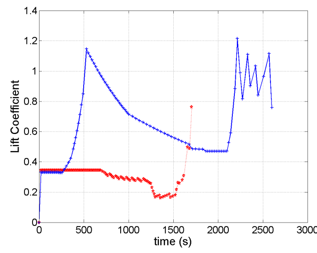
(c) Mach number



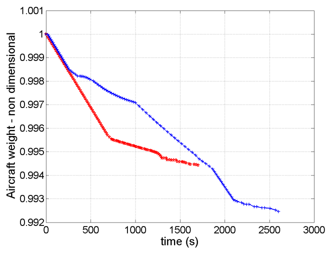
(d) Calibrated air speed



(e) Non dimensional thrust

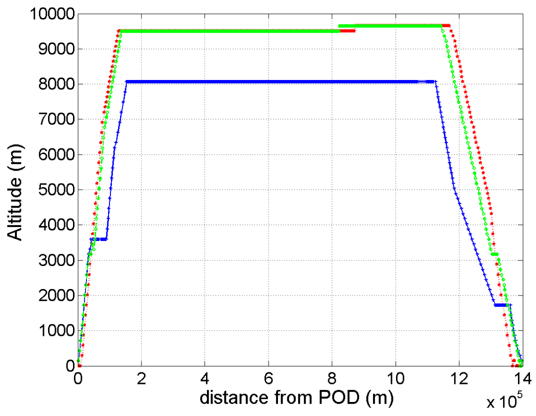


(f) Lift coefficient

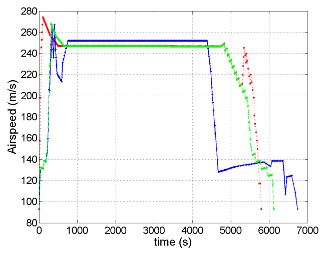


(g) Aircraft mass

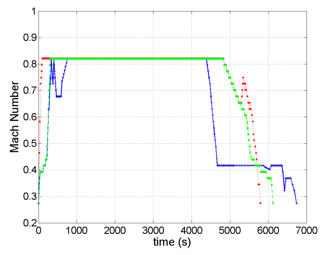
Figure 6.8: Optimal descent trajectories



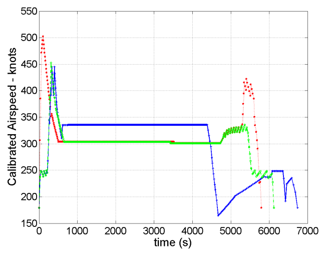
(a) Altitude



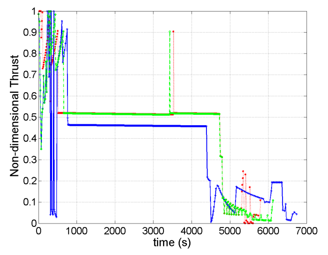
(b) Air speed



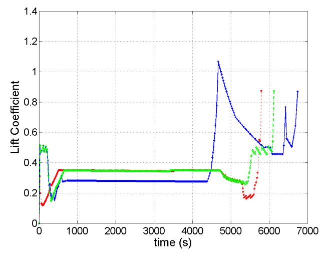
(c) Mach number



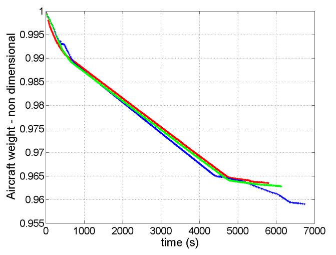
(d) Calibrated air speed



(e) Non dimensional thrust



(f) Lift coefficient



(g) Aircraft mass

Figure 6.9: Complete trajectory optimization

Chapter 7

Conclusion and Discussion

7.1 Concluding remarks

This works shows how discrete search techniques can be used to effectively used to find optimal trajectories for commercial airlines in the current air traffic system. Such a technique can be used to compare different types of profiles that aircrafts fly. A realistic wind model can be used in such a simulation. We have shown that a discrete search has significant advantages which easily over weigh the issues of runtime and computational memory required and it can be used as a very effective method to be considered for future flight planning methodologies.

7.2 Recommendations for future work

On the same lines as implementation of two dimensional case, a full three dimensional trajectory optimization can be performed. This will complete the algorithm and it will result in a true optimization methodology. Wind models have to implemented in both the two and three dimensional cases to model more realistic flight optimization. A statistical analysis with several wind models can be performed to determine the sensitivity of the algorithm.

In addition, there are several ways in which one can significantly improve the runtime. Fuel heuristics can be used to increase the speed of A* search. An approximate optimal flight profile always consists of climb, cruise and a descent segment. The grid points can be generated in such a way as to remove all grid points which for sure do not lie in regions of climb, cruise or descent. Dijkstra's and A* can be implemented using parallel computing techniques [39] which will reduce the runtime by quite a significant amount. The use of Fibonacci heaps rather than a conventional binary heap is also reported to be more efficient [40] .

Bibliography

[1] John-Paul Clarke, John Brown, Kevin Elmer, Nhut Ho, Liling Ren, Kwok-On Tong, and Joseph Wat. Continuous Descent Approach: Design and Flight Test for Louisville International Airport. *Journal of Aircraft*, 41(5):1054–1066, September 2004.

[2] Singapore Airlines. *Singapore Airlines Announces Trial To Go Paperless*, 2010. [http://www.singaporeair.com/mediacentre/pacontent/news/NE_2410.jsp].

[3] Cathay Pacific Airlines. *Cathay Pacific silver bullet freighter takes to the skies*, 2006. [http://www.cathaypacific.com/cpa/en _LK/aboutus/pressroomdetails?refID=bc79f0b86b9ab010VgnVCM22000022d21c39].

[4] South West Airlines. *Fuel Savings: Reducing Our Environmental Impact*, 2009. [http://www.southwest.com/html/southwest-difference/southwest-citizenship/environmental-initiatives/index.html].

[5] Yiyuan Zhao and Rhonda a. Slattery. Capture conditions for merging trajectory segments to model realistic aircraft descents. *Journal of Guidance, Control, and Dynamics*, 19(2):453–460, March 1996.

[6] E. Zermelo. *ber die Navigation in der Luft als Problem der Variationsrechnung [On the Navigation in the Air as a Problem of the Calculus of Variations]* - jahresbericht der deutschen mathematiker-vereinigung, 1930. [Vol. 39, 2nd section ,pg 44-48].

[7] Ho Y. C Bryson, A. E. Jr. *Applied Optimal Control: Optimization, Estimation and Control* - rev., hemisphere, new york, 1975. [chapter 6 & pg 77-80].

[8] Matthew R Jardin. Neighboring Optimal Aircraft Guidance in Winds Introduction. 24(4):2–7, 2001.

[9] Matt R Jardin. Analytical Solutions for Minimum-Time Neighboring Optimal Aircraft Guidance in Winds. *Control Solutions*, 0(August):1–11, 2008.

[10] Matt R Jardin and The Mathworks. Methods for Computing Minimum-Time Paths in Strong Winds. *Cycle*, (August), 2010.

[11] Irons R. P. Zagalsky, N. R. and R. L. Schultz. *Energy State Approximation and Minimum Fuel Fixed-Range Trajectories* - journal of aircraft,, 1971. [Vol. 8, June 1971, pp; 488-490].

[12] R. L. Schultz and N. R. Zagalsky. *Aircraft Performance Optimization* - journal of aircraft,, 1972. [Vol. 9, Feb. 1972, pp. 108-114].

[13] J. L Speyer. *On the Fuel Optimality of Cruise* - journal of aircraft,, 1973. [Vol. 10, Dec. 1973, pp. 763-765].

[14] J. L Speyer. *Nonoptimality of the Steady State Cruise for Aircraft* - aiaa journal,, 1976. [Vol. 14, Nov. 1976, pp. 1604-1610].

[15] R. L. Schultz. *Fuel Optimality of Cruise* - journal of aircraft,, 1974. [Vol. 11, Sept. 1974, pp. 586-587].

[16] P K A Menon. Study of Aircraft Cruise. 12(5), 1989.

[17] Gottfried Sachs and Rainer Mehlhornf. Enhancement of Endurance Performance by Periodic Optimal Camber Control. 16(3), 1993.

[18] Robert H. Chen and Jason L. Speyer. Improved Endurance of Optimal Periodic Flight. *Journal of Guidance, Control, and Dynamics*, 30(4):1123–1133, July 2007.

[19] W Grimm and K H Wellt. Periodic Control for Minimum-Fuel Aircraft Trajectories. 9(2):169–174, 1986.

[20] Abhijit Chakravarty. *Four-Dimensional Fuel-Optimal Guidance in the Presence of Winds* - aiaa guidance and control. (1):16–22.

[21] Frank Neuman. for Jet Transports. *Fuel*, 8(5):650–657.

[22] Heinz Erzberger and Homer Lee. Constrained Optimum Trajectories with Specified Range. *Control*, 3(1):78–85.

[23] John-Paul Clarke, John Brown, Kevin Elmer, Nhut Ho, Liling Ren, Kwok-On Tong, and Joseph Wat. Continuous Descent Approach: Design and Flight Test for Louisville International Airport. *Journal of Aircraft*, 41(5):1054–1066, September 2004.

[24] Mead R. Coppenbarger, R. and D. Sweet. *Field Evaluation of the Tailored Arrivals Concept for Datalink-Enabled Continuous Descent Approach* - aiaa-2007-7778,, 2007. [7th AIAA ATIO Conf, Belfast, Northern Ireland, Sep. 18-20].

[25] Anthony Warren, Kwok-on Tong, Boeing Air, and Traflc Management. Development of continuous descent approach concepts for noise abatement. pages 3–6, 2002.

[26] Kevin Elmer, Joseph Wat, Belur Shivashankara, Daniel Mcgregor, and David Lambert. AIAA 2002-5869 AIAA 2002-5869. (October):5869–5869, 2002.

[27] Frizo Vormer, Max Mulder, J.a. Mulder, and Rene Paassen. Optimization of Flexible Approach Trajectories Using a Genetic Algorithm. *Journal of Aircraft*, 43(4):941–952, July 2006.

[28] Eric Hoffman, Peter Martin, Thomas Pütz, Aymeric Trzmiel, and Karim Zeghal. Airborne Spacing: Flight Deck View of Compatibility with Continuous Descent Approach (CDA). *Interface*, (September):1–12, 2007.

[29] G.L. Slater. *Study on variations in vertical profile for CDA descents* - aiaa-2007-7778,, 2009. [9th AIAA Aviation Technology, Integration, and Operations Conference 21 - 23 September 2009].

[30] John E. Robinson III Keenan Roach. *A Terminal Area Analysis of Continuous Ascent Departure Fuel Use at Dallas/Fort Worth International Airport* - aiaa-2007-7778,, 2010. [10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference 13 - 15 September 2010, Fort Worth, Texas].

[31] Francois Le Sellier. Discrete Real-Time Flight Plan Optimization. 1999.

[32] W.a. Kamal and I. Postlethwaite. Real Time Trajectory Planning for UAVs Using MILP. *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 3381–3386, 2005.

[33] Arthur Richards and Jonathan P How. Collision Avoidance Using Mixed Integer Linear Programming 1. *Time.*

[34] Hong Yang and Yiyuan Zhao. Trajectory Planning for Autonomous Aerospace Vehicles amid Known Obstacles and Conflicts. *Journal of Guidance, Control, and Dynamics*, 27(6):997–1008, November 2004.

[35] Michael R. Jackson, Rhonda a. Slattery, and Yiyuan J. Zhao. Sensitivity of Trajectory Prediction in Air Traffic Management. *Journal of Guidance, Control, and Dynamics*, 22(2):219–228, March 1999.

[36] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, December 1959.

[37] Wikipedia. Dijkstra's algorithm — Wikipedia, the free encyclopedia, 2011. [Online; accessed 17-April-2011].

[38] N.X Vinh. *Introduction to Aircraft Performace, Selection and Design* - john wiley & sons, 1984. [Sections 4-2 to 4-5, Chap 10].

[39] Clark F. Olson. Parallel algorithms for hierarchical clustering. *Parallel Computing*, 21(8):1313 – 1325, 1995.

[40] Michael L. Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34(3):596–615, July 1987.