# Trajectory Optimization using Mixed-Integer Linear Programming

by

Arthur George Richards

Master of Engineering
University of Cambridge, 2000

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2002

Author . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Department of Aeronautics and Astronautics
May 5, 2002

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Jonathan P. How
Associate Professor
Thesis Supervisor

Accepted by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Wallace E. Vander Velde
Chairman, Department Committee on Graduate Students

# Trajectory Optimization using Mixed-Integer Linear Programming

by

Arthur George Richards

Submitted to the Department of Aeronautics and Astronautics
on May 5, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science

## Abstract

This thesis presents methods for finding optimal trajectories for vehicles subjected to avoidance and assignment requirements. The former include avoidance of collisions with obstacles or other vehicles and avoidance of thruster plumes from spacecraft. Assignment refers to the inclusion of decisions about terminal constraints in the optimization, such as assignment of waypoints to UAVs and the assignment of spacecraft to positions in a formation. These requirements lead to non-convex constraints and difficult optimizations. However, they can be formulated as *mixed-integer linear programs* (MILP) that can be solved for global optimality using powerful, commercial software.

This thesis provides several extensions to previous work using MILP. The constraints for avoidance are extended to prevent *plume impingement*, which occurs when one spacecraft fire thrusters towards another. Methods are presented for efficient simplifications to complex problems, allowing solutions to be obtained in practical computation times. An approximation is developed to enable the inclusion of aircraft dynamics in a linear optimization, and also to include a general form of waypoint assignment suitable for UAV problems. Finally, these optimizations are used in *model predictive control*, running in real-time to incorporate feedback and compensate for uncertainty.

Two major application areas are considered: spacecraft and aircraft. Spacecraft problems involve minimum fuel optimizations, and include ISS rendezvous and satellite cluster configuration. Aircraft problems are solved for minimum flight-time, or in the case of UAV problems with assignment, waypoint values and vehicle capabilities are included. Aircraft applications include air traffic management and coordination of autonomous UAVs. The results in this thesis provide a direct route to globally-optimal solutions of these non-convex trajectory optimizations.

Thesis Supervisor: Jonathan P. How
Title: Associate Professor

## Acknowledgments

My thanks to my advisor, Professor Jon How, for his support and guidance through the course of this degree, and to all my friends and colleagues at MIT for their good-humoured comradeship, both in the office and out.

*I dedicate this thesis to my mother and wish her a very happy birthday.*

The research was funded in part under Draper Laboratory grant DL-H539057, Air Force grant # F49620-01-1-0453 and DARPA contract # N6601-01-C-8075.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Problem Definition

This section defines the class of problems to be considered in this thesis, and explains the issues that make them complex. Examples are then given of real problems in this class. All involve planning optimal trajectories for vehicles. These are therefore optimizations subject to dynamics constraints. The particular feature of the problems in this thesis is the inclusion of other, *non-convex* constraints. These arise from two requirements: avoidance and assignment. *Avoidance* is the requirement to remain outside certain regions of the solution space, such as those which would lead to collisions. The general term "avoidance" is used because more complicated requirements also fall into this category, such as the avoidance of thruster plumes from spacecraft. *Assignment* refers to the inclusion of variable boundary conditions (initial or terminal), subject to logical constraints. For example, the desired layout of a cluster of spacecraft may be specified, but the assignment of particular spacecraft to locations is free, and therefore to be chosen for minimum fuel use. Assignment problems involve discrete decisions, such as "go to point A or point B", and therefore lead to non-convex constraints.

Their non-convexity makes these problems intrinsically difficult to solve. Path-planning in the presence of non-convex obstacles has been shown to be $\mathcal{NP}$-complete [9]. Assignment problems are essentially combinatorial and therefore can lead to very large solution spaces. Furthermore, the combined problems of path-planning and assignment are tightly coupled: the assignment is strongly dependent on the paths taken, and to decouple the problems would require computing paths for all possible assignments, which is prohibitive.

### 1.1.1 Spacecraft Problems

The configuration of a cluster of spacecraft leads to a problem of this class. Autonomous formation flying of satellite clusters has been identified as an enabling technology for many future NASA and U.S. Air Force missions [1, 2, 3, 4]. Fig. 1-1 shows an artist's impression of the proposed TechSat21 mission [4, 68], using separated spacecraft for Earth observation using space-based radar. The use of fleets of small satellites, instead of a single monolithic satellite, offers improved science return through longer baseline observations, enables faster ground track repeats, and provides a high degree of redundancy and reconfigurability in the event of a single vehicle failure. These benefits can only be achieved at the expense of more stringent requirements on fleet coordination, high-level mission management, and fault detection [5, 6]. In some formation flying scenarios [4, 7], the vehicles will be arranged around a *passive aperture*, which are short-baseline, periodic formation configurations that provide good, distributed Earth-imaging while reducing the tendency of the vehicles to drift apart [16, 27, 28, 29]. Changing the viewing mode of the fleet could require a change in the formation configuration, but only the relative alignment of the spacecraft is critical. This leads to an assignment problem for the reconfiguration. Collision avoidance is also a constraint on such maneuvers. Another concern is *plume impingement*: if one spacecraft fires it thrusters at another, the plume can damage the impinged spacecraft. The high-energy particles can cause pitting or deposition on sensitive instruments, such as mirrors [26, 4].

Autonomous rendezvous is another spacecraft problem of significant interest. Programs are underway to develop autonomous rendezvous capability for missions to the International Space Station (ISS) [63, 65]. Similar capabilities are required for future on-orbit servicing systems [64, 66]. Again, collision avoidance and plume impinge-

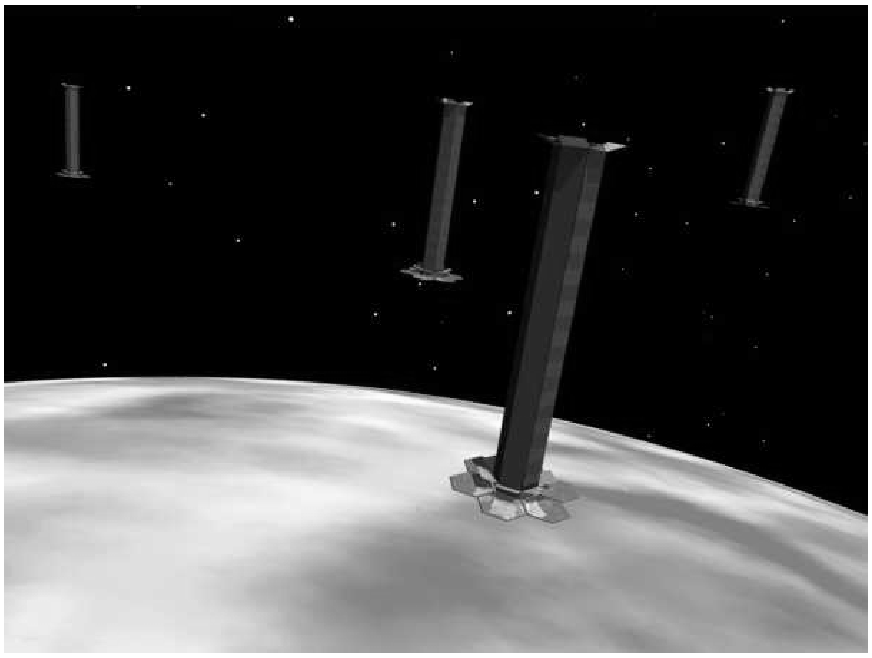


Figure 1-1: Artist's Impression of the TechSat21 Mission [68]

ment are major concerns. In particular, structural loading due to plume impingement during docking was a design driver for ISS solar panels [23].

### 1.1.2 Aircraft Problems

Two major applications drive this research: air traffic management and autonomous Unmanned Aerial Vehicles (UAVs). Future air traffic concepts involve "free-flight," in which flight-planning and conflict resolution are performed on-board [41]. Both areas require path-planning methods for multiple vehicles, avoiding obstacles and each other.

UAV problems often involve the additional complexity of waypoint assignment [42, 43, 55]. Small groups of vehicles operate autonomously, with the high-level goal of visiting a set of waypoints. The allocation of waypoints to UAVs is to be determined according to capability and timing constraints. This assignment is required to optimize some metric for the mission, such as minimum time, maximum reward, or minimum risk. All of these objectives are highly dependent on the paths taken, since this determines the order of events, the flight time and the risk exposure. This makes the two problems of assignment and path-planning strongly coupled.

## 1.2 Solution Concepts

This thesis considers three major steps in the solution of the problems discussed in Section 1.1. The first, discussed in Section 1.2.1 is the representation of non-convex trajectory optimizations as integer programs. The second, discussed in Section 1.2.2, is the development of efficient approximations to enable these optimizations to be solved quickly. Finally, Section 1.2.3 discusses the use of the optimizations in a real-time, Model Predictive Control scheme to compensate for uncertainty, such as noise and disturbance.

### 1.2.1 Mixed-Integer Linear Programming

The approach presented in this thesis formulates the problem as a *mixed-integer linear program* (MILP). This is a modification to a linear program (LP) in which some variables are constrained to take only integer values. In particular, we use binary variables, taking only the values 0 or 1. Constraints on such variables enable the inclusion of discrete decisions in the optimization [10, 11], encoding the non-convexity of the problem. Both avoidance and assignment constraints can be considered in terms of such decisions. For collision avoidance, a vehicle must either be "left" or "right" of an obstacle, each leading to a convex sub-problem. Assignment can be expressed as discrete choices of destinations. Constraints on the binary variables are used to include logical requirements on the decisions, such as compatible assignments.

In general, MILPs are also $\mathcal{NP}$-complete [12], indicating that the MILP representation retains the inherent complexity of the problem. However, in many instances, MILPs can be solved using a branch-and-bound algorithm, exploiting their relaxation to LP form to accelerate the solution process. The MILP form of the trajectory optimization problems is linear by definition, so the method is immune to issues of local minima and globally-optimal solutions can be found. Highly-optimized, commercial software is available for this process. These codes were developed to solve MILPs in the field of operations research, such as airline scheduling [13]. The CPLEX optimization software [14] is used to solve the MILPs in this thesis, although various other options exist. CPLEX implements the branch-and-bound algorithm in conjunction with many adjustable heuristics, allowing quite large problems to be solved in practical computation times.

There are two major drawbacks to the MILP approach. The first is the intensive nature of the computation, which is centralized and scales poorly with problem size. The second is the restriction to linear problems. However, this thesis demonstrates that the method can solve realistic problems. Solutions can be obtained in practical computation times, and linear constraints provide good approximations to the systems of interest.

### 1.2.2 Approximations for Solving MILP Problems

MILP representations of trajectory problems can involve many binary variables, typically thousands. Depending on the nature of the trajectory problem, various techniques can be used to simplify the MILP and accelerate the solution process. In this thesis, methods are presented for using prior knowledge of the solution to identify redundant or inactive constraints before solving the problem.

If the solution is likely to have a "bang-off-bang" profile, plume constraints in middle of the time interval are likely to be inactive. An iterative scheme is developed in which plume constraints are first applied only at the start and end of the maneuver. The solution is post-analyzed for plume impingement, and repeated if necessary with additional plume constraints. It is often quicker to solve several simplified problems than the completely constrained problem. Similar iterative schemes have been successfully applied to scheduling problems in operations research [59].

In other problems, it is likely that adjoining time steps will have the same binary variable settings. A formulation is developed in which binary variables are 'shared' across adjacent time steps, reducing the complexity of the problem with very little change to the solution.

### 1.2.3 Model Predictive Control

Solving a single MILP optimization generates an optimal trajectory for the problem seen at a particular instant. In practice, this "snapshot" will be subject to uncertainty, and the actual problem to be solved will change as time progresses. Furthermore, the model of the system used in the optimization will be imperfect, and the system state will not evolve exactly as predicted. The single MILP is an open-loop solution and cannot account for these uncertainties.

For application to real-time control, the MILP trajectory optimizations are embedded within *Model Predictive Control* (MPC) [60]. This is a scheme in which trajectory optimization is performed at each time step, finding a solution to complete the problem from the current position. Only the first step of the resulting control sequence is implemented and the process is then repeated. This incorporates feedback in the control, allowing it to compensate for uncertainties such as model error, disturbances and noise. It also introduces new challenges into the problem: it is required to run in real-time, and the stability of the resulting controller must be demonstrated.

## 1.3 Survey of Previous Work

### 1.3.1 Avoidance

Many approaches have been investigated for solving the problem of trajectory optimization with collision avoidance for dynamic systems. All techniques for solving these problems involve some kind of simplification, aiming to capture certain key elements of the problem in a form suitable for computation. Potential functions have been employed in the fields of spacecraft [20], air traffic management [51] and UAV planning [54, 56]. This approach involves replacing avoidance constraints with proximity penalties in the objective function, allowing simpler optimization schemes, such as steepest descent, to be used. Such schemes offer fast operation, some with provable safety, but without optimality. Randomized searches [17, 57, 58] were developed to rapidly find feasible paths through fields of obstacles, again neglecting optimality. In

UAV problems, Voronoi diagrams [54, 42] have been used to find coarse paths between radar hazards, approximating the trajectories as joined line segments. In "reactive" schemes [18], vehicles fly a nominal trajectory and perform predetermined evasive maneuvers when conflicts are detected. For some aircraft problems, path-planning can be reduced to a single heading change decision [19, 44], which greatly simplifies the global trajectory optimization. An indirect method [40] performs iterative searches for solutions to necessary conditions for aircraft dynamics. Other approaches use splines [21] and lower-dimensional representations [22] of nonlinear systems to reduce the solution space before performing nonlinear optimization.

The MILP approach in this thesis simplifies the dynamics model to a linearized form, while retaining the full non-convex constraints for avoidance and obtaining globally optimal solutions.

### 1.3.2 Assignment

Most approaches decouple the problems of configuration selection and trajectory planning. Once the trajectory costs have been calculated for each configuration option and the corresponding paths, the selection is a linear assignment problem, which is readily solved using standard LP tools [31]. A method for spacecraft [7] computes the costs for many maneuvers and then selects the one that gives the lowest overall cost. Similar auction-based approaches [42, 52, 53] are used for UAV problems given cost approximations for trajectories or trajectory segments. Another approach to the spacecraft reconfiguration problem involved separating the reconfiguration into permutation maneuvers within groups of satellites [30].

An advantage of the MILP approach in this thesis is its retention of the inherent coupling of the assignment and path-planning sub-problems. They are solved in a single, centralized optimization.

### 1.3.3 Model Predictive Control

MPC has become popular in industry, particularly in the process industry, and has recently received considerable attention in the controls community [60, 61, 62]. In particular, Bemporad and Morari [47] have considered the theory for the use of MILP in MPC, with examples in process control. In the field of aerospace, Manikonda [69] applied MPC to spacecraft formation keeping, without considering avoidance constraints. Dunbar [70] used nonlinear optimization in MPC for a flight experiment.

This thesis presents the combination of MILP and MPC for spacecraft control, enabling the inclusion of avoidance constraints in the real-time controller.

## 1.4 Layout of Thesis

The main body of this thesis is divided into three chapters. Chapter 2 deals with spacecraft applications. Using these examples, the fundamentals of the MILP approach are demonstrated. This chapter includes two of the major contributions of this work: the use of prior knowledge to develop efficient approximations for solving trajectory optimizations; and the extension of avoidance constraints to prevent plume impingement. Chapter 3 shows the method of including aircraft dynamics, which are fundamentally nonlinear, in a linear optimization, another of the key developments of this work. This chapter also includes work on forms of assignment constraints suitable for UAV problems. Chapter 4 presents work on MPC, returning to the context of spacecraft rendezvous.

# Chapter 2

# Spacecraft Applications

This chapter demonstrates the MILP approach with application to spacecraft maneuvering problems. These are typically minimum-fuel, fixed-time maneuvers, and may include collision and plume avoidance constraints. These examples are used to introduce the formulation for avoidance using mixed-integer constraints. Furthermore, methods of simplification are introduced to deal with the large number of binary variables that arise in avoidance problems, reducing computation time to practical periods.

Sections 2.2 and 2.3 are reviews, of the minimum-fuel LP problem and collision avoidance, respectively. Section 2.3 also includes an introduction to the principle of using MILP for avoidance and the details of the solution procedure. Section 2.4 shows the extensions to prevent plume impingement. Section 2.5 discusses the computational intensity of the problems and introduces methods to reduce solution times. Finally, Section 2.6 presents modified initial and terminal constraints to include assignment in the problem.

## 2.1 Nomenclature for Spacecraft Problems

The following variable names are used in this chapter:

|  |  |
|---|---|
| $\mathbf{x}$ | vehicle state |
| $\mathbf{u}$ | control input |
| $T$ | Number of time-steps |
| $V$ | Number of vehicles |
| $N$ | Number of dimensions |
| $G$ | Number of global configurations available for end states |
| $M$ | Large number for logical constraints |
| $P$ | Plume length |
| $W$ | Plume width |

The following subscripts are used:

|  |  |
|---|---|
| $i$ | time-step |
| $p$, $q$ | vehicles |
| $n$, $m$ | axes in some orthogonal co-ordinate frame |
| $l$ | obstacle |
| $g$ | global configuration for final states |
| $r$ | position within final configuration |

## 2.2 Linear Program for Minimum Fuel Spacecraft Problems

The core of the optimization is to choose discrete state values $\mathbf{x}_{\mathrm{ip}}$ for each vehicle $p$ and time-step $i \in [0 \ldots T]$ and the corresponding input values $\mathbf{u}_{\mathrm{ip}}$. The state at the first time point is constrained to be the specified starting conditions

$$\mathbf{x}_{0\mathrm{p}} = \mathbf{x}_{\mathrm{S_p}} \tag{2.1}$$

where $\mathbf{x}_{\mathrm{S_p}}$ is the initial state vector for the $p^{\mathrm{th}}$ vehicle. Similarly, the state at the final time point is fixed at the specified end conditions

$$\mathbf{x}_{\mathrm{Tp}} = \mathbf{x}_{\mathrm{F_p}} \tag{2.2}$$

where $\mathbf{x}_{\mathrm{F_p}}$ is the final state vector for the $p^{\mathrm{th}}$ vehicle. Later problems will demonstrate more flexible terminal constraints to include higher level mission requirements. The states at intermediate points in time must be consistent with the system dynamics

$$\mathbf{x}_{\mathrm{(i+1)p}} = \mathbf{A}\mathbf{x}_{\mathrm{ip}} + \mathbf{B}\mathbf{u}_{\mathrm{ip}} \tag{2.3}$$

where $\mathbf{A}$ and $\mathbf{B}$ are a discretized form of the continuous system dynamics. For simplicity, the constraints show the same dynamics for all vehicles, but it is straightforward to modify the formulation to account for heterogenous vehicles. For the problems of interest in this chapter, in which the spacecraft are in close proximity on similar orbits, the most common approximation for spacecraft dynamics are the linearized Hill's equations [15]

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{m} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{2.4}$$

where the $x$-coordinate is in the radial direction, the $y$-coordinate is in the in-track direction and the $z$-component is in the out-of-plane direction [15]. The spacecraft mass is $m$ and the natural frequency of the reference orbit is $\omega$. The corresponding state and input vectors are $\mathbf{x} = [x\ y\ z\ \dot{x}\ \dot{y}\ \dot{z}]^{\mathrm{T}}$ and $\mathbf{u} = [u_x\ u_y\ u_z]^{\mathrm{T}}$. These equations are discretized resulting in the form in Eqn. 2.3. This can be done by assuming that the thrust $\mathbf{u}_{\mathrm{ip}}$ is applied continuously throughout the time-step or impulsively at the beginning of each step. Also, other linearized models of the relative dynamics exist and can also be used in this optimization framework [32].

The state and input vectors are confined to lie within specified, symmetrical limits

$$-x_{\mathrm{max_n}} \leq x_{\mathrm{ipn}} \leq x_{\mathrm{max_n}} \tag{2.5}$$

$$-u_{\mathrm{max_n}} \le u_{\mathrm{ipn}} \le u_{\mathrm{max_n}} \tag{2.6}$$

where $x_{\mathrm{ipn}}$ and $u_{\mathrm{ipn}}$ denote the $n^{\mathrm{th}}$ component of the state and thrust vectors respectively for the $p^{\mathrm{th}}$ vehicle at the $i^{\mathrm{th}}$ time-step. The input limit is typically used to represent the limited force available from each thruster. The velocity constraints can represent safety limits, such as a maximum maneuvering speed in the proximity of another spacecraft. The position limits have no such significance, but their presence ensures that the problem is bounded.

The objective is to minimize the total fuel consumption of all vehicles in the problem. For a total of $V$ vehicles moving in $N$-dimensional space over $T$ time-steps, the cost function is

$$J = \sum_{i=0}^{T-1} \sum_{p=1}^{V} \sum_{n=1}^{N} |u_{\mathrm{ipn}}| \tag{2.7}$$

This piecewise linear function can be converted to a linear function using slack variables [33].

In summary, the minimum-fuel path-planning linear program, neglecting avoidance so far, is

$$\min_{\mathbf{u},\mathbf{x}} J = \sum_{i=0}^{T-1} \sum_{p=1}^{V} \sum_{n=1}^{N} |u_{\mathrm{ipn}}| \tag{2.8}$$

such that

$$\begin{array}{lrcccll}
\forall p \in [1 \dots V], & & \mathbf{x}_{\mathrm{(i+1)p}} & = & \mathbf{A}\mathbf{x}_{\mathrm{ip}} + \mathbf{B}\mathbf{u}_{\mathrm{ip}} & \forall i \in [0 \dots T-1] & \\
\text{and} & & \mathbf{x}_{\mathrm{0p}} & = & \mathbf{x}_{\mathrm{S_p}} & & \\
\text{and} & & \mathbf{x}_{\mathrm{Tp}} & = & \mathbf{x}_{\mathrm{F_p}} & & \\
\text{and} & -x_{\mathrm{max_n}} \le & x_{\mathrm{ipn}} & \le & x_{\mathrm{max_n}} & \forall i \in [0 \dots T-1] & \forall n \in [1 \dots N] \\
\text{and} & -u_{\mathrm{max_n}} \le & u_{\mathrm{ipn}} & \le & u_{\mathrm{max_n}} & \forall i \in [0 \dots T-1] & \forall n \in [1 \dots N]
\end{array} \tag{2.9}$$

All of the problems considered in this chapter involve a linear program of the form shown above, subject to additional mixed-integer constraints to enforce various forms of avoidance.

## 2.3 Collision Avoidance for Spacecraft

This section introduces the constraints for avoidance, describes the implementation of MILP-based path-planning and shows some examples of collision avoidance maneuvers. This is a review of work by Schouwenaars [35]. It is presented here as a recap, since it forms the basis for the developments in this thesis.

### 2.3.1 Obstacle Avoidance Constraints

This section presents the additional constraints on the linear program to avoid static obstacles [34, 35]. The obstacles can be modeled in this framework as convex polygons of any number of sides, but, to simplify the presentation, the results in this thesis only use rectangles. In spacecraft applications, "obstacle" refers to any region that the vehicles must avoid that is not under our control. For example, when planning a rendezvous maneuver with a space station, the station would be modeled as a collection of fixed obstacles, rather than another vehicle. Collisions are prevented by ensuring that the vehicle trajectories lie outside the obstacles at each of the discrete time points. Note that it is feasible for the trajectory to cut into obstacles between the discrete time points. It is therefore necessary to enlarge the obstacle models and select the time-step length such that these incursions cannot intersect the real obstacles.

For visualization, the constraints are first derived for the two dimensional case. The location of the rectangular obstacle is defined by its lower left-hand corner $(x_{\min}, y_{\min})$ and its upper right-hand corner $(x_{\max}, y_{\max})$. The point $(x, y)$ must lie in the area outside of the obstacle. This requirement can be formulated as the set of conditions

$$\begin{array}{rcl} x & \leq & x_{\min} \\ \text{or } x & \geq & x_{\max} \\ \text{or } y & \leq & y_{\min} \\ \text{or } y & \geq & y_{\max} \end{array} \tag{2.10}$$

These constraints can be transformed into a mixed-integer form by introducing binary variables $a_{\mathrm{k}}$ [10, 12]. Let $M$ be an arbitrary positive number, larger than any other distance in the problem. The constraints in Eqn. 2.10 are represented by the following mixed-integer constraints

$$\begin{array}{rrcrcl} & x & \leq & x_{\min} & + & Ma_1 \\ \text{and} & -x & \leq & -x_{\max} & + & Ma_2 \\ \text{and} & y & \leq & y_{\min} & + & Ma_3 \\ \text{and} & -y & \leq & -y_{\max} & + & Ma_4 \\ \text{and} & \sum_{k=1}^{4} a_{\mathrm{k}} & \leq & 3. & & \end{array} \tag{2.11}$$

Note that if $a_{\mathrm{k}} = 0$, then the $k^{\mathrm{th}}$ constraint from Eqn. 2.10 is enforced. However, if $a_{\mathrm{k}} = 1$, then that constraint is relaxed, because the $M$ term moves the upper bound beyond the solution space. The last *and*-constraint in Eqn. 2.11 ensures that no more than three constraints from Eqn. 2.10 are relaxed, and hence at least one of the original *or*-constraints is satisfied.

These constraints are then enforced at every time step, and extended to a general number of dimensions (in practice, $N = 2$ or $N = 3$), vehicles and obstacles. The position of vehicle $p$ at time-step $i$ is the vector $\mathbf{x}_{ip} = [x_{\mathrm{ip1}} \ldots x_{\mathrm{ipN}}]^{\mathrm{T}}$. The vertex of obstacle $l$ with the minimum value of each coordinate is at position $\mathbf{L}_l$ (this is the bottom left hand corner in the 2-D case). Its vertex with the maximum of each coordinate is at $\mathbf{U}_l$. The binary variables $a_{iplk}$ are the switches, with $k \in [1 \ldots 2N]$, corresponding to being on one of two sides of the obstacle in each of $N$ dimensions. The complete formulation is

$$\begin{array}{rrcll} \forall p, \forall l, \forall i \in [1 \ldots T-1]: & x_{\mathrm{ipn}} & \geq & U_{\mathrm{ln}} - Ma_{\mathrm{ipln}} & \forall n \\ \text{and} & x_{\mathrm{ipn}} & \leq & L_{\mathrm{ln}} + Ma_{\mathrm{ipl(n+N)}} & \forall n \\ \text{and} & \sum_{k=1}^{2N} a_{\mathrm{iplk}} & \leq & 2N-1 & \end{array} \tag{2.12}$$

These become additional constraints on the linear program in Eqs. 2.8 and 2.9. The binary variables $a$ become extra decision variables in the problem. The new constraints in Eqn. 2.12 are linear in the decision variables, so the new problem is a *mixed-integer linear program.*

### 2.3.2 Collision Avoidance Constraints

This section derives constraints to avoid collisions between different vehicles [34, 35]. Every pair of vehicles must be at least a specified distance apart in each direction at each time-step. This corresponds to the enforcement of a rectangular exclusion region around each vehicle.

As in the previous section, the constraints are first shown in two dimensions for clarity. Let the safety distances in the $X-$ and $Y-$directions be denoted by $r_{\mathrm{x}}$ and $r_{\mathrm{y}}$ respectively. Denote the positions of two different vehicles $p$ and $q$ as $(x_{\mathrm{p}}, y_{\mathrm{p}})$ and $(x_{\mathrm{q}}, y_{\mathrm{q}})$, respectively. The constraints to ensure safe separation between vehicles $p$ and $q$ are written as

$$\begin{aligned} & x_{\mathrm{p}} - x_{\mathrm{q}} \geq r_{\mathrm{x}} \\ \text{or} \quad & x_{\mathrm{q}} - x_{\mathrm{p}} \geq r_{\mathrm{x}} \\ \text{or} \quad & y_{\mathrm{p}} - y_{\mathrm{q}} \geq r_{\mathrm{y}} \\ \text{or} \quad & y_{\mathrm{q}} - y_{\mathrm{p}} \geq r_{\mathrm{y}} \end{aligned} \tag{2.13}$$

As in the previous section, these can be converted to the more useful *and*-constraints by introducing binary variables, giving

$$\begin{aligned} & x_{\mathrm{p}} - x_{\mathrm{q}} && \geq && r_{\mathrm{x}} - Mb_1 \\ \text{and} \quad & x_{\mathrm{q}} - x_{\mathrm{p}} && \geq && r_{\mathrm{x}} - Mb_2 \\ \text{and} \quad & y_{\mathrm{p}} - y_{\mathrm{q}} && \geq && r_{\mathrm{y}} - Mb_3 \\ \text{and} \quad & y_{\mathrm{q}} - y_{\mathrm{p}} && \geq && r_{\mathrm{y}} - Mb_4 \\ \text{and} \quad & \sum_{k=1}^{4} b_{\mathrm{k}} && \leq && 3 \end{aligned} \tag{2.14}$$

where $M$ is the same large number used in the previous section.

This is extended to the general case using the same notation as before. The safety avoidance distance in direction $n$ is $r_n$. The condition $q > p$ avoids duplication of the constraints on the positions of pairs of vehicles.

$$\begin{array}{rrcll} \forall p, q | q > p : \forall i \in [1 \ldots T-1] : & x_{\mathrm{ipn}} - x_{\mathrm{iqn}} & \geq & r_n - M b_{\mathrm{ipqn}} & \forall n \\ \text{and} & x_{\mathrm{iqn}} - x_{\mathrm{ipn}} & \geq & r_n - M b_{\mathrm{ipq(n+N)}} & \forall n \\ \text{and} & \sum_{k=1}^{2N} b_{ipqk} & \leq & 2N - 1 & \end{array} \tag{2.15}$$

These constraints can also be added to the linear program of Eqs. 2.8 and 2.9. The binary variables $b$ become additional decision variables in the optimization problem.

### 2.3.3 Implementation Details

The global optimization problem is solved by a MILP solver, based on the branch-and-bound algorithm, implemented in the CPLEX software package [14]. The AMPL language[38] is used as the interface to CPLEX. Implementing the constraints in AMPL is straightforward, requiring minimal translation from the form shown in the equations. The problem formulation and constraints are defined in a model file, while the parameter values are in a separate data file. As a result, changes to the problem can be made without rebuilding the constraint expressions. AMPL combines the model and data files into a suitable format before invoking CPLEX to solve the problem. A combination of Matlab and AMPL scripts enables the path-planning problem to be initiated by a single command and then conveniently combined with simulation and plotting utilities.

### 2.3.4 Example: 2-D Vehicles with Collision Avoidance

These simple examples demonstrate the effect of the avoidance constraints. The vehicles involved are modeled as point masses, moving in 2-D with thruster actuation in both axes. Fig. 2-1 shows an avoidance maneuver designed using the constraints in Eqn. 2.12. A single vehicle is required to move from the start point to the end point without hitting the obstacles between. The figure shows that the designed trajectory is outside the obstacles at all the time points.

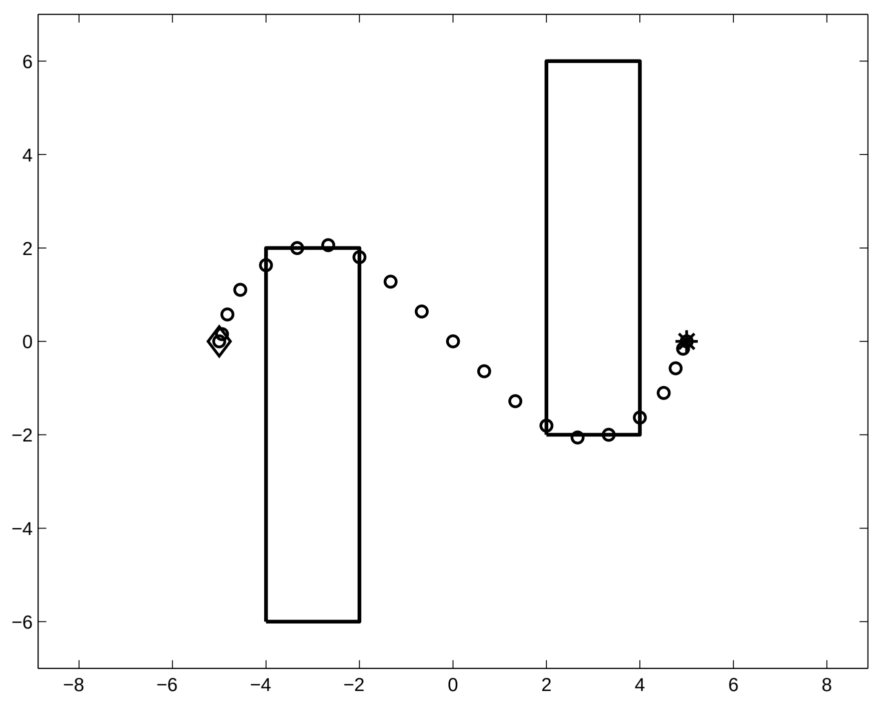


Figure 2-1: Example Obstacle Avoidance Maneuver in 2-D

In the second example, three vehicles are required to move between the positions given in Table 2.1. The start and end positions are all on the $X$-axis, but the order of the vehicles is reversed. The LP solution, which would involve each vehicle following a straight line from is start to its end point, would clearly lead to a collision between all three vehicles at the half-way point.

Table 2.1: Start and End Positions for 2-D Collision Avoidance Example

| Vehicle | Start Position | End Position |
|---|---|---|
| 1 | (-2,0) | (8,0) |
| 2 | (0,0) | (6,0) |
| 3 | (2,0) | (4,0) |

Fig. 2-2 shows the result of the MILP optimization that includes the collision avoidance constraints from Eqn. 2.15 with a safety distance of one unit. Vehicles 2

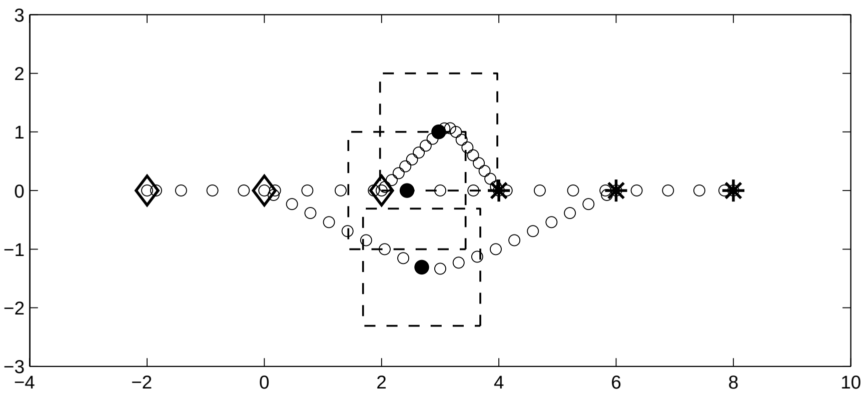


Figure 2-2: Example Collision Avoidance Maneuver in 2-D

(lower) and 3 (upper) move in the $Y$-direction off the $X$-axis to allow vehicle 1 to follow a straight-line path. The heavy dots indicate the positions of the vehicles at the tenth time-step, with the corresponding exclusion boxes shown dashed. Vehicles 1 and 3 are separated by exactly the safety distance in the $Y$-direction. At earlier time-steps, vehicles 1 and 2 also move along the edges of the exclusion regions. This shows that collision avoidance is efficiently implemented by this MILP formulation.

These simple examples show that the MILP constraints enforce avoidance effectively and efficiently. Even in simple cases, the optimal solutions to avoidance problems are rarely obvious, but can be found directly using MILP. Also, the solutions found use exactly the avoidance margins specified.

### 2.3.5 Example: ISS Remote Camera with Collision Avoidance

This problem involves a micro-satellite being used for external inspection of the International Space Station (ISS). The satellite is required to move between specified start and end points on opposite sides of the station without colliding with the structure or firing its thrusters at the station. The collision avoidance part of this problem was addressed by Roger and McInnes [20] using potential functions.

The dynamics are the Hill's equations for a 90 minute orbit, as in the previous

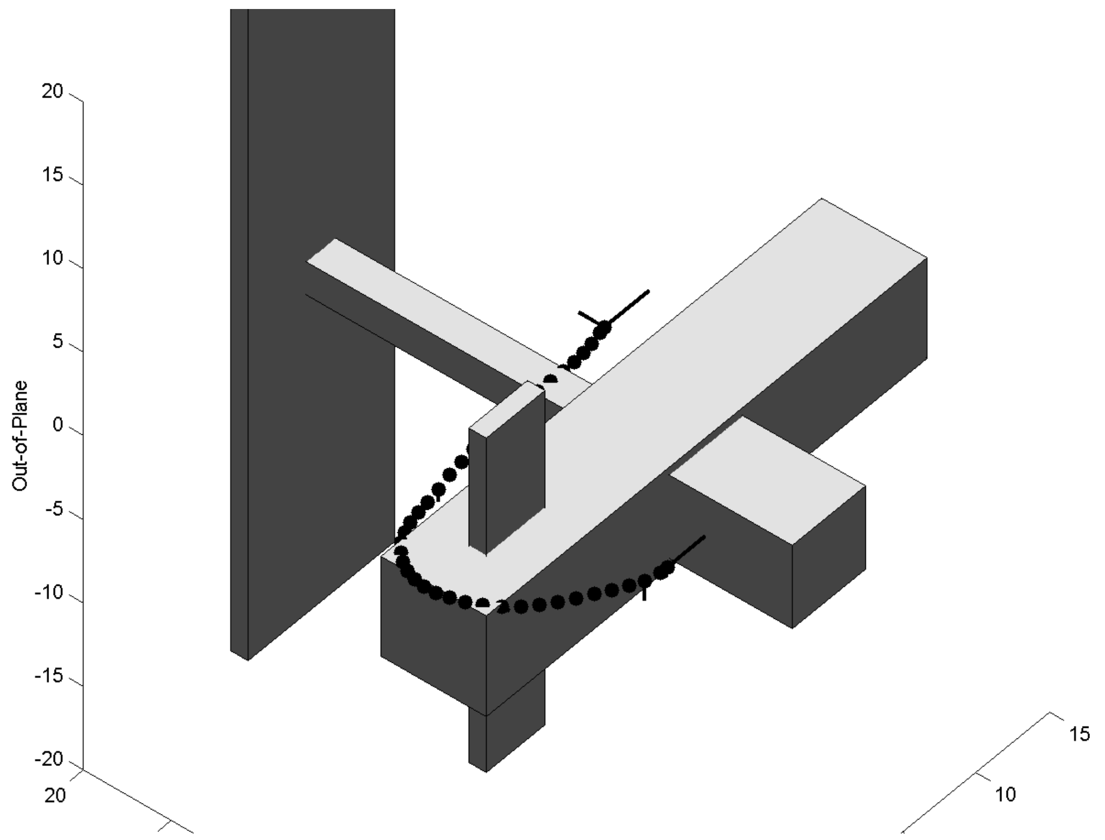


Figure 2-3: ISS Remote Camera Maneuver with Collision Avoidance. All dimensions are in meters.

example. The camera satellite is modeled as a mass of 5 kg with thrusters giving up to 1 mN in each direction. The ISS is modeled as a collection of boxes as shown in Fig. 2-3. The maneuver lasts for 4000 seconds and is discretized into 40 time-steps.

This problem was solved with collision avoidance constraints. The designed trajectory is shown in Fig. 2-3. The total fuel use is equivalent to a $\Delta V$ of 0.236 m/s. Section 2.4.5 extends this example to include plume avoidance constraints. The computational complexity of this problem is discussed in Section 2.5.

## 2.4 Plume Impingement Avoidance

This section develops the formulation to prevent plume impingement by one spacecraft upon another [36]. The plumes extend in discrete directions from the vehicle, which assumes that the thrusters are aligned with the axes. The inclusion of vehicle attitude variation within this formulation would complicate it considerably, and is the subject of ongoing research. As in the case of obstacle avoidance, the plume region could be represented by any convex polygon, but a rectangular shape is used here for simplicity. Two forms of plume constraints are presented: those for impingements between vehicles under our control, and those for impingements by vehicles upon fixed obstacles. As discussed in Section 2.3.1, vehicles not under our control and large structures are modeled as fixed obstacles in the Hills-frame space.

### 2.4.1 Plume Avoidance Constraints for Vehicles

All other vehicles are required to remain outside the plume region while the corresponding thruster is firing. Conversely, a spacecraft cannot fire a thruster if the resulting plume would impinge upon another vehicle. Once again, the constraints are first developed in two dimensions to simplify the visualization. Figure 2-4 (a) shows the modeled impingement region extending in the $-X$-direction. The vehicles shown by $\circ$ are clear of the plume since they are outside the impingement region. The vehicle shown by $\times$ will be impinged if the thruster is firing in the $-X$-direction, generating thrust in the $+X$-direction, but could still escape impingement if the thruster is not firing. Thus for the plume shown, there are five ways to avoid impingement: either the thruster is not firing, or the target vehicle is clear of the box in any of the four directions. These can be represented by the *or*-constraints

$$\begin{array}{rrcl} & u_{x\mathrm{p}} & \leq & 0 \\ \text{or} & x_\mathrm{p} - x_\mathrm{q} & \geq & P \\ \text{or} & x_\mathrm{q} - x_\mathrm{p} & \geq & 0 \\ \text{or} & y_\mathrm{p} - y_\mathrm{q} & \geq & W \\ \text{or} & y_\mathrm{q} - y_\mathrm{p} & \geq & W \end{array} \tag{2.16}$$

where $W$ is the plume half-width, $P$ is the plume length and $u_{x\mathrm{p}}$ is the thrust from vehicle $p$ in the $X$-direction. So, for the situation shown in Fig. 2-4, vehicle $p$ is

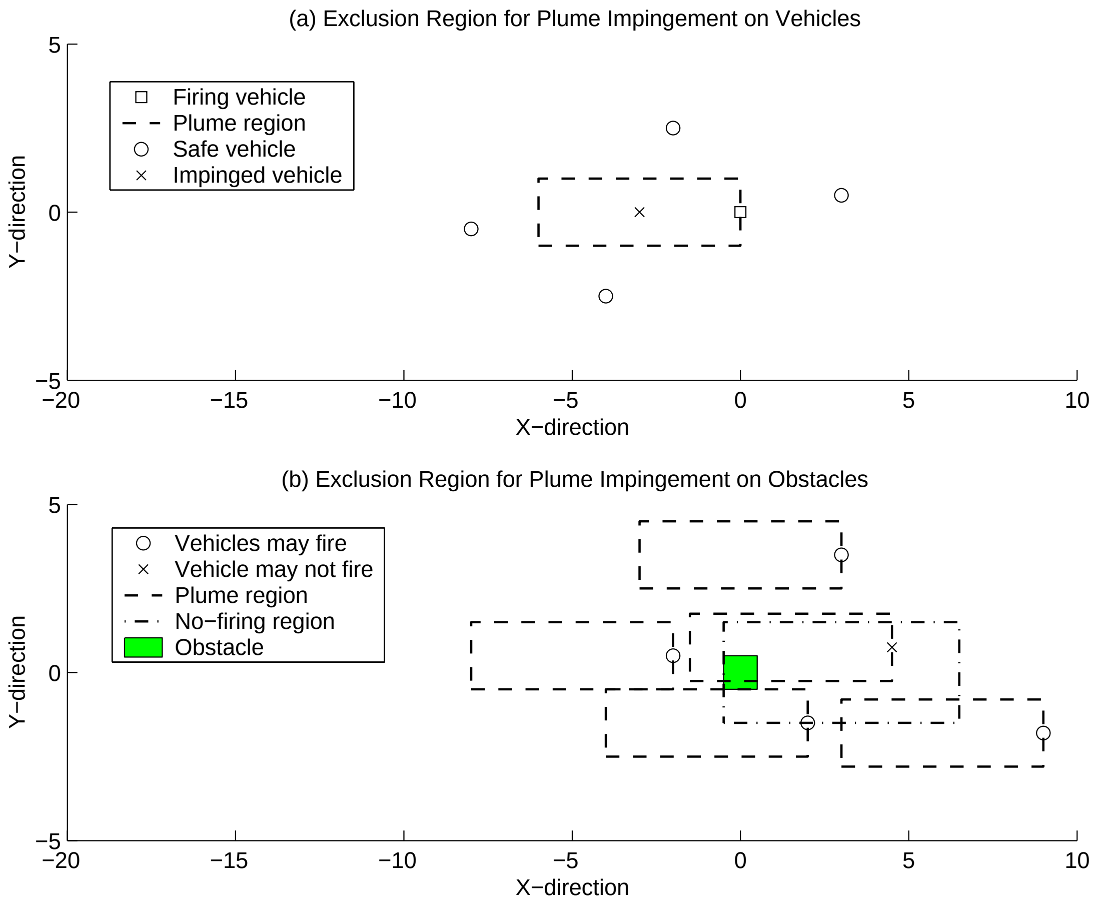


Figure 2-4: Examples of Plume Impingement Regions in 2-D

marked by the □ and any of the others may be vehicle $q$. The vehicles marked by $\circ$ each satisfy one of the last four constraints. The vehicle marked by $\times$ satisfies none of the last four constraints because it is inside the plume region, but it will not be impinged if the first constraint is satisfied.

As shown previously, these constraints can be converted to the more convenient *and*-form using binary variables

$$\begin{array}{rrcrcl} & u_{xpi} & \leq & & & Mc_0 \\ \text{and} & x_p - x_q & \geq & P & - & Mc_1 \\ \text{and} & x_q - x_p & \geq & & - & Mc_2 \\ \text{and} & y_p - y_q & \geq & W & - & Mc_3 \\ \text{and} & y_q - y_p & \geq & W & - & Mc_4 \\ \text{and} & \sum_{k=0}^{4} c_{\mathrm{k}} & \leq & 4 & & \end{array} \tag{2.17}$$

To extend the constraints to the most general case of all time steps, dimensions and vehicles, denote the thrust vector for vehicle $p$ at time-step $i$ as $\mathbf{u}_{ip} = [u_{\mathrm{ip1}} \ldots u_{\mathrm{ipN}}]^{\mathrm{T}}$. The formulation for vehicle $q$ to avoid the plumes from forward (positive thrust) thrusters of vehicle $p$ is

$$\begin{array}{rrcccl} \multicolumn{6}{l}{\forall p, q | q \neq p : \forall n : \forall i \in [0 \ldots T-1] :} \\ & -u_{ipn} & \geq & & -Mc^{+}_{ipqn0} & \\ \text{and} & x_{ipn} - x_{iqn} & \geq & P & -Mc^{+}_{ipqnn} & \\ \text{and} & x_{iqn} - x_{ipn} & \geq & & -Mc^{+}_{ipqn(n+N)} & \\ \text{and} & x_{ipm} - x_{iqm} & \geq & W & -Mc^{+}_{ipqnm} & \forall m | m \neq n \\ \text{and} & x_{iqm} - x_{ipm} & \geq & W & -Mc^{+}_{ipqn(m+N)} & \forall m | m \neq n \\ \text{and} & \sum_{k=0}^{2N} c^{+}_{ipqnk} & \leq & 2N & & \end{array} \tag{2.18}$$

Similarly, the constraints for avoiding plumes from reverse thrusters are

$$
\begin{array}{llclll}
\multicolumn{6}{l}{\forall p, q | q \neq p : \forall n : \forall i \in [0 \ldots T-1] :} \\
 & u_{ipn} & \geq & & -Mc^{-}_{ipqn0} & \\
\text{and} & x_{ipn} - x_{iqn} & \geq & & -Mc^{-}_{ipqnn} & \\
\text{and} & x_{iqn} - x_{ipn} & \geq & P & -Mc^{-}_{ipqn(n+N)} & \\
\text{and} & x_{ipm} - x_{iqm} & \geq & W & -Mc^{-}_{ipqnm} & \forall m | m \neq n \\
\text{and} & x_{iqm} - x_{ipm} & \geq & W & -Mc^{-}_{ipqn(m+N)} & \forall m | m \neq n \\
\text{and} & \sum_{k=0}^{2N} c^{-}_{ipqnk} & \leq & 2N & &
\end{array}
\qquad (2.19)
$$

When these constraints are added to the linear program in Eqs. 2.8 and 2.9, the variables $c$ become additional decision variables.

## 2.4.2 Plume Avoidance Constraints for Obstacles

This section derives the constraints to prevent vehicles from firing their thrusters when they would impinge upon fixed obstacles. Fig. 2-4 (b) shows how vehicles may avoid impinging on an obstacle in 2-D. The vehicles marked by $\circ$ are all free to fire as shown since their plumes will not contact the obstacle. The vehicle marked by $\times$ will impinge on the obstacle if it fires in the direction shown. It is clear from this example that there is a region around the obstacle in which the vehicles cannot emit plumes in the $-X$-direction without impinging. As before, there are five ways to avoid impinging: to be outside the box, or not to fire. These can be written as the following *or*-group using the corners of the obstacle, as shown in the obstacle avoidance section.

$$
\begin{array}{rrcrcl}
 & & & u_{x\mathrm{p}} & \leq & 0 \\
\text{or} & x_{\mathrm{p}} & - & x_{\max} & \geq & P \\
\text{or} & x_{\min} & - & x_{\mathrm{p}} & \geq & 0 \\
\text{or} & y_{\mathrm{p}} & - & y_{\max} & \geq & W \\
\text{or} & y_{\min} & - & y_{\mathrm{p}} & \geq & W
\end{array}
\tag{2.20}
$$

As before, binary variables are added to convert to *and*-form and extend to the general case. The constraints for forward thrust are

$$
\begin{array}{l}
\forall p : \forall l : \forall n : \forall i \in [0 \ldots T-1] : \\
\begin{array}{rrcrclll}
 & & & -u_{ipn} & \geq & & -Md^{+}_{ipqn0} & \\
\text{and} & x_{ipn} & - & U_{ln} & \geq & P & -Md^{+}_{ipqnn} & \\
\text{and} & L_{ln} & - & x_{ipn} & \geq & & -Md^{+}_{ipqn(n+N)} & \\
\text{and} & x_{ipm} & - & U_{lm} & \geq & W & -Md^{+}_{ipqnm} & \forall m|m \neq n \\
\text{and} & L_{lm} & - & x_{ipm} & \geq & W & -Md^{+}_{ipqn(m+N)} & \forall m|m \neq n \\
\text{and} & & & \sum_{k=0}^{2N} d^{+}_{ipqnk} & \leq & 2N & &
\end{array}
\end{array}
\tag{2.21}
$$

and similarly for reverse thrust

$$
\begin{array}{l}
\forall p : \forall l : \forall n : \forall i \in [0 \ldots T-1] : \\
\begin{array}{rrcrclll}
 & & & u_{ipn} & \geq & & -Md^{-}_{ipqn0} & \\
\text{and} & x_{ipn} & - & U_{ln} & \geq & & -Md^{-}_{ipqnn} & \\
\text{and} & L_{ln} & - & x_{ipn} & \geq & P & -Md^{-}_{ipqn(n+N)} & \\
\text{and} & x_{ipm} & - & U_{lm} & \geq & W & -Md^{-}_{ipqnm} & \forall m|m \neq n \\
\text{and} & L_{lm} & - & x_{ipm} & \geq & W & -Md^{-}_{ipqn(m+N)} & \forall m|m \neq n \\
\text{and} & & & \sum_{k=0}^{2N} d^{-}_{ipqnk} & \leq & 2N & &
\end{array}
\end{array}
\tag{2.22}
$$

Again, these can be appended to the linear program Eqs. 2.8 and 2.9, the binary variables $d$ becoming extra decision variables in the optimization.

### 2.4.3 Example: 2-D Vehicles with Plume Avoidance

Fig. 2-5 shows an example of a maneuver with plume avoidance between vehicles. The maneuver from Fig. 2-1 is redesigned with the additional constraints from Eqs. 2.18 and 2.19 to prevent plume impingement. Observe that in Fig. 2-1, the two vehicles that move away from the X-axis do so with some velocity in the X-direction, implying that their thrusters fired to the left and impinged upon other vehicles. In the redesigned maneuver of Fig. 2-5, the second vehicle from the left moves away in the Y-direction only, to avoid firing upon the other two. The heavy dots (●) mark the vehicle positions at the third time step, and the dashed line marks the plume region due to firing at that time. Vehicle # 2 fires as soon as possible without impinging on Vehicle # 1, which is just on the edge of the plume region, and vehicle # 3 has remained at its starting position. It does not move until both other vehicles are out of its plume region.

Fig. 2-6 shows a simple example involving the constraints for plume impingement on obstacles. A vehicle has to approach and stop next to an obstacle. In Fig. 2-6(a), the maneuver is designed without the plume constraints, and the final braking thrust impinges upon the obstacles. In Fig. 2-6(b), the same maneuver is designed with plume constraints in place. To avoid impinging with the braking thrust, the vehicle diverts from the direct path, and its final approach direction is such that the braking thrust does not impinge.

These simple examples demonstrate the effect of the plume constraints and verify that they perform as expected.

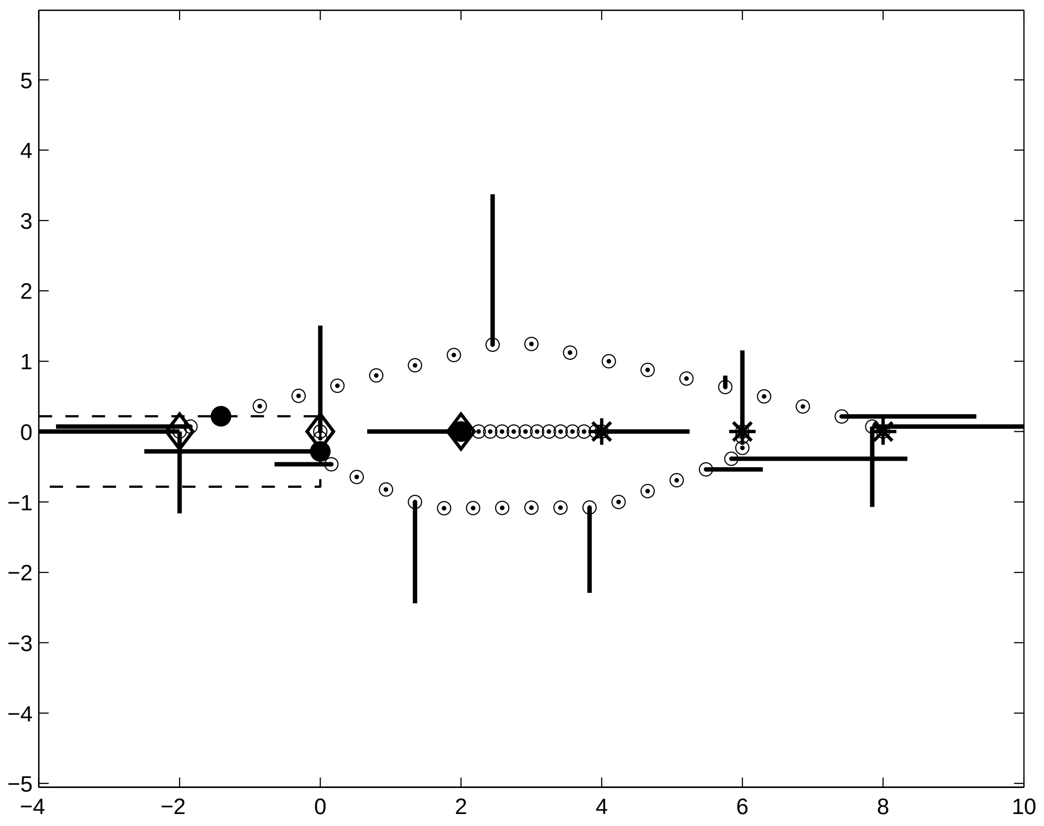
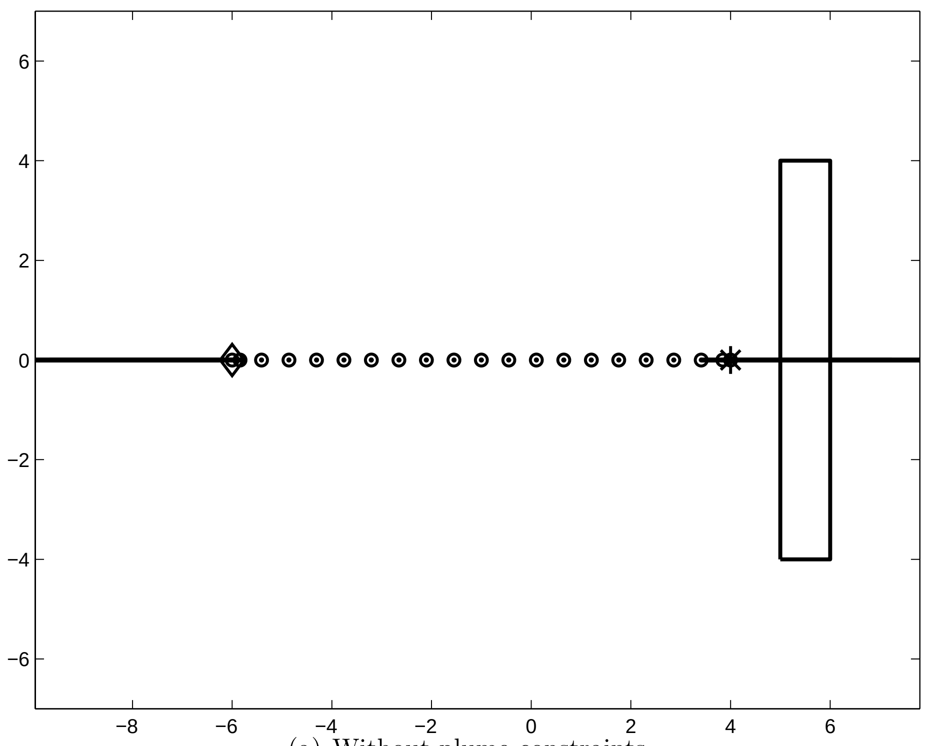


Figure 2-5: Example Plume Avoidance Maneuver in 2-D. The lines mark thruster firing, with length proportional to the thrust demanded.


(a) Without plume constraints

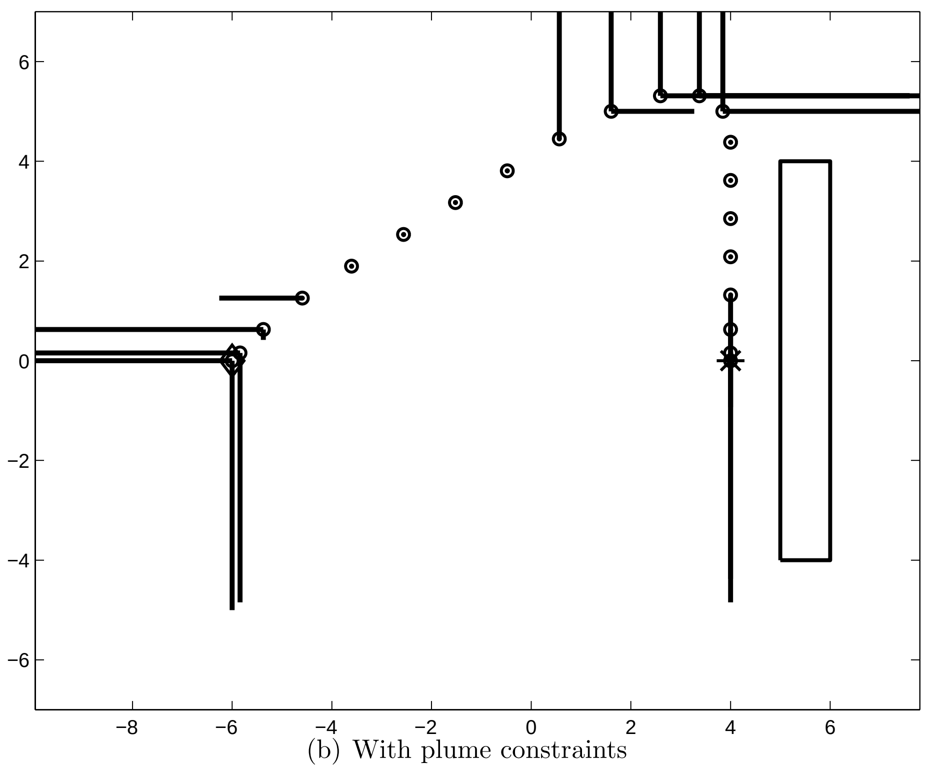

(b) With plume constraints

Figure 2-6: Demonstration of Plume Constraints in 2-D Obstacle Problem

### 2.4.4 Example: Multi-Satellite Plume Impingement Avoidance

Section 2.4.3 showed the effect of plume constraints on simple vehicles. This section shows their application to a spacecraft problem with relative orbital dynamics. Three identical spacecraft of mass 30 kg are separated along a line in-track. They are required to reconfigure onto one of the passive apertures, which in this case is a triangle in the plane of the orbit. The maneuver must be done in 9 minutes, which is one tenth of a 90 minute orbit. This artificially-short time scale makes in-track firing a favorable option where available, so the effect of the plume impingement constraints is clearly demonstrated. Each spacecraft has thrusters providing up to 0.2 N in each direction. The plume avoidance regions for the thrusters are 50 m wide and extend 120 m from the spacecraft. Collision avoidance is also enforced with a safety distance of 10 m. The problem was discretized into 20 time-steps.

The left-hand plot in Fig. 2-7 shows the trajectory designed without plume constraints. Note that the designed trajectories remain in the orbital plane, even though the full three-dimensional model from Eqn. 2.4 was used in the problem. As expected, substantial in-track firing is used to complete the problem in the time available. Fig. 2-8 shows details of the start and end of the maneuver. In Fig. 2-8(a), showing the first firing step, vehicle 1 ($\bigtriangledown$) impinges on both other vehicles, while vehicle 2 ($\square$) fires on vehicle 3 ($\bigcirc$) as well. Fig. 2-8(b) shows the final firing step, at which vehicle 3 fires upon both other vehicles.

The right-hand plot in Fig. 2-7 shows the trajectories redesigned to prevent plume impingement, with details of key steps shown in Fig. 2-9. Considerable deviations from the previous case are evident. Fig. 2-9(a) shows the first step, for comparison with Fig. 2-8(a). Vehicles 1 and 2 fire only radially, avoiding impingement. Note that vehicle 2 moves away in the opposite direction to the previous result. At time step 3, shown in Fig 2-9(b), vehicle 2 has moved in the radial direction and is now able to fire in-track without impinging upon vehicle 3. At step 6, shown in Fig. 2-9(c), vehicles 2 and 3 have become separated by exactly the width of the plume region,

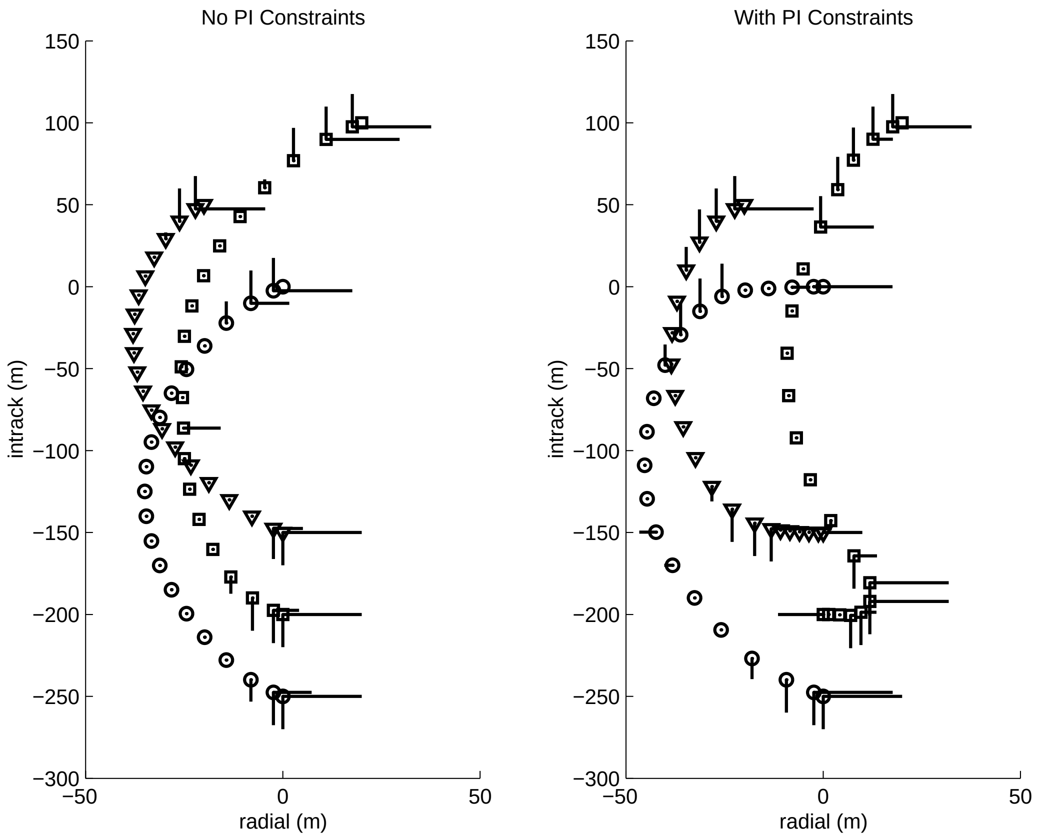


Figure 2-7: Reconfiguration Maneuver without Plume Impingement Constraints (Left) and With Plume Impingement Constraints (Right). The trajectory of Vehicle #1 is marked by $\bigtriangledown$, vehicle #2 by $\square$, and vehicle #3 by $\bigcirc$. The lines indicate the direction of the firing plumes. Their lengths are proportional to the thrust demanded.

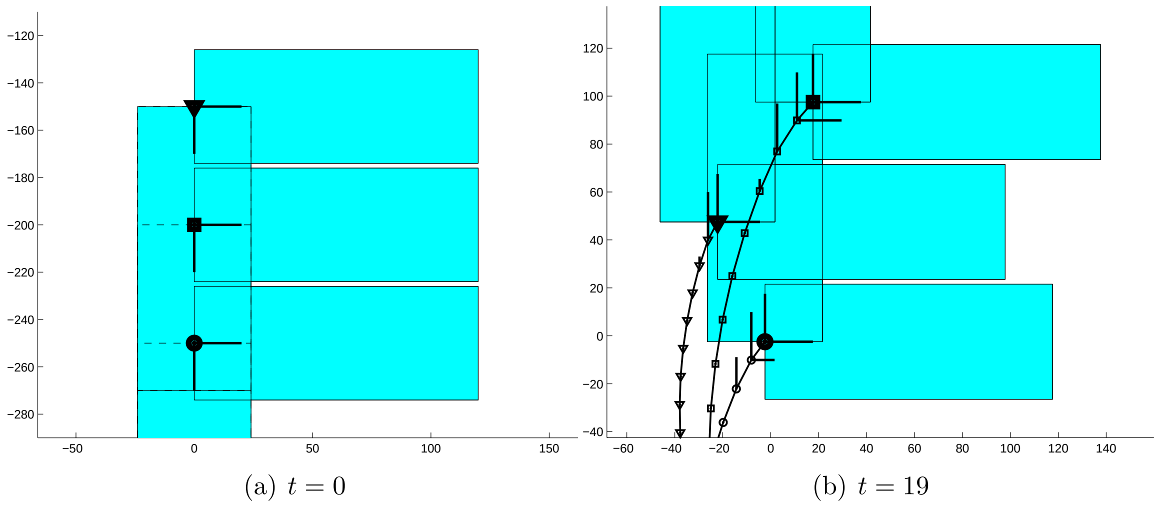


Figure 2-8: Detail of Spacecraft Maneuver without Plume Constraints. Vehicle #1 is marked by $\bigtriangledown$, vehicle #2 by $\square$, and vehicle #3 by $\bigcirc$. The lines indicate the direction of the firing plumes. Their lengths are proportional to the thrust demanded. The shaded areas are the plume avoidance regions.

and vehicle 1 fires into the space between them.

Fig. 2-9(d) shows the 13th time step, as the vehicles approach the end of the maneuver. After significant firing in the early steps, vehicle 3 has moved ahead of vehicle 1, and is able to fire in-track for braking without impinging. At step 15, shown in Fig. 2-9(e), vehicle 3 makes its final in-track braking burn, just clear of vehicle 2. Note that vehicle 1 will pass into the plume region at the very next step. Fig. 2-9(f) shows the final firing step. Unlike the situation in Fig. 2-8(b), the approach direction of vehicle 3 is such that it does not need to fire in-track, impinging upon the other vehicles.

This example has shown many complicated interactions between vehicles, with positions and firing carefully co-ordinated between the group to avoid impingement throughout the maneuver. The MILP approach offers a direct route to these solutions.

### 2.4.5 Example: ISS Remote Camera with Plume Avoidance

This section demonstrates the effect of adding plume constraints to the ISS camera example from Section 2.3.5. When solved for collision avoidance only, as shown in Fig. 2-3, the final braking thrust impinges upon the station. Fig. 2-10 shows the trajectory redesigned to prevent plume impingement, using a plume length $P = 10$m and half-width $W = 1$m. The final stages of this maneuver are shown in Fig. 2-11, seen in a larger scale from a different angle. Since the final position is in a corner formed by two adjacent modules, the camera satellite must approach from the side to prevent its braking thrust from impinging on the station. The figure shows the satellite making the necessary adjustment to its course by firing while still clear of the station, and its final approach leaves it requiring a braking thrust in the only available direction. The total fuel use for this maneuver is equivalent to a $\Delta V$ of 0.269 m/s, compared to 0.236 m/s for collision avoidance only. The small increase in firing is needed to achieve the final approach direction. The computational complexity of this problem is discussed in Section 2.5.

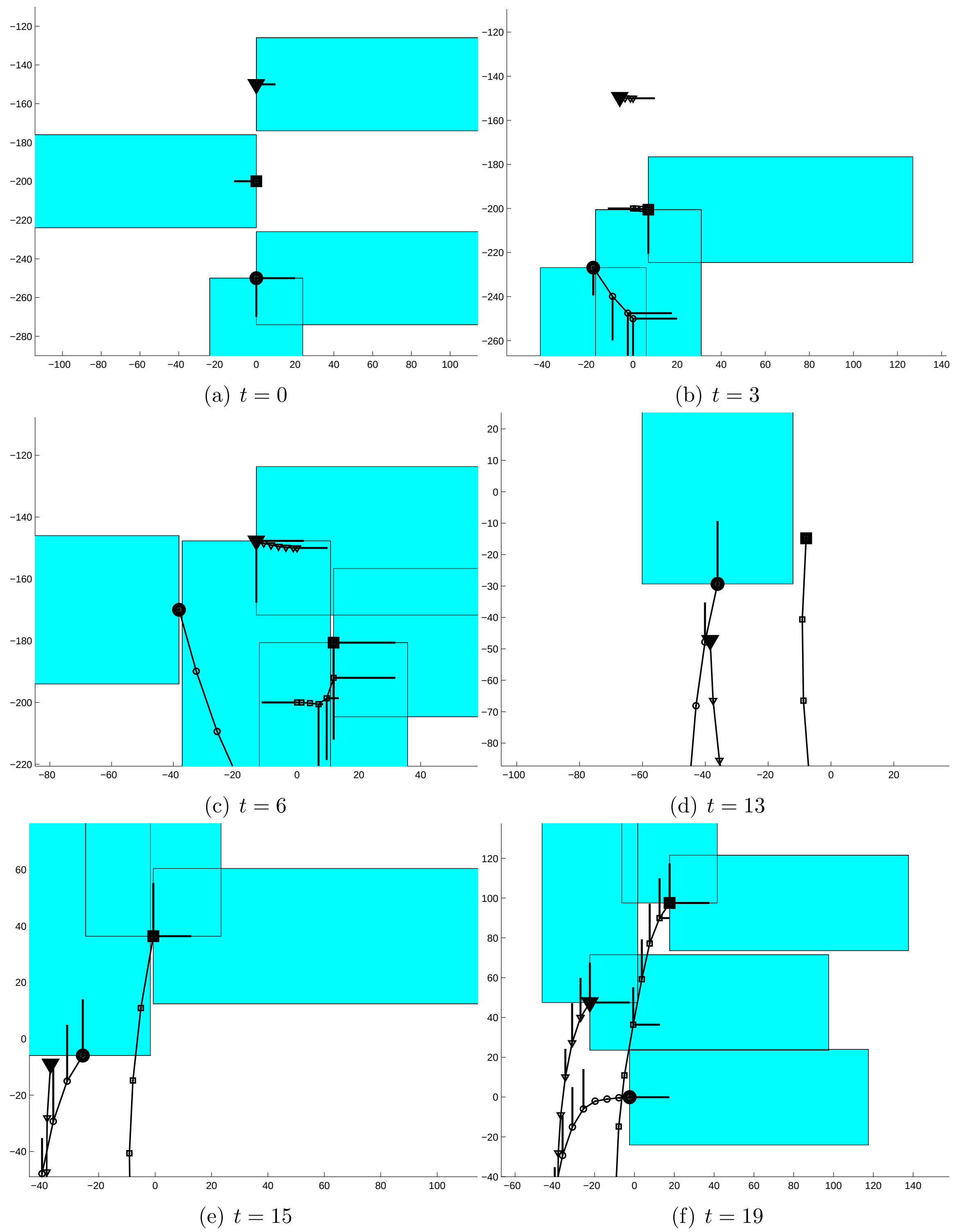

(a) $t = 0$ (b) $t = 3$ (c) $t = 6$ (d) $t = 13$ (e) $t = 15$ (f) $t = 19$

Figure 2-9: Detail of Spacecraft Maneuver with Plume Constraints. Vehicle #1 is marked by $\bigtriangledown$, vehicle #2 by $\square$, and vehicle #3 by $\bigcirc$. The lines indicate the direction of the firing plumes. Their lengths are proportional to the thrust demanded. The shaded areas are the plume avoidance regions.

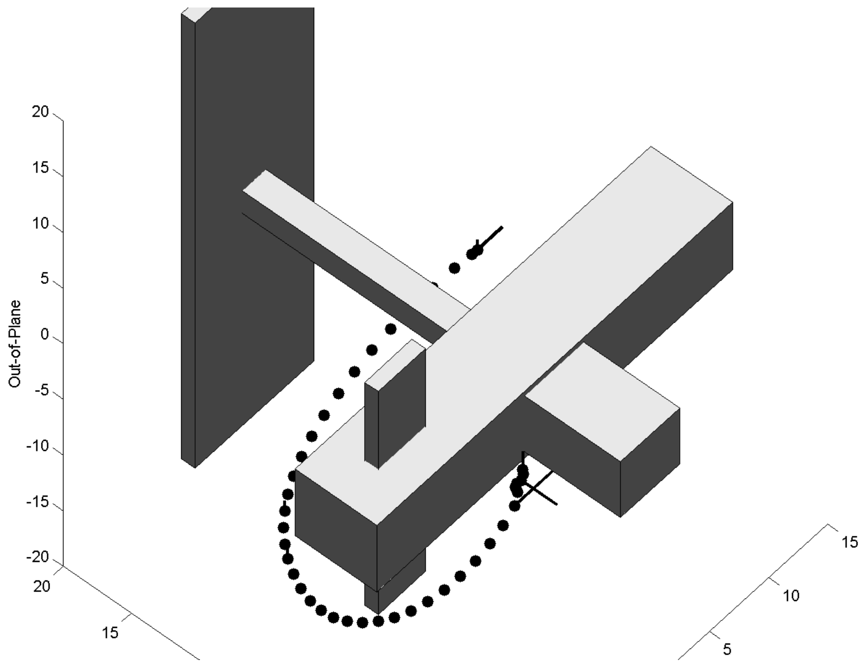


Figure 2-10: ISS Remote Camera Maneuver with Plume Impingement Constraints. Start and end positions are the same as in Fig. 2-3.

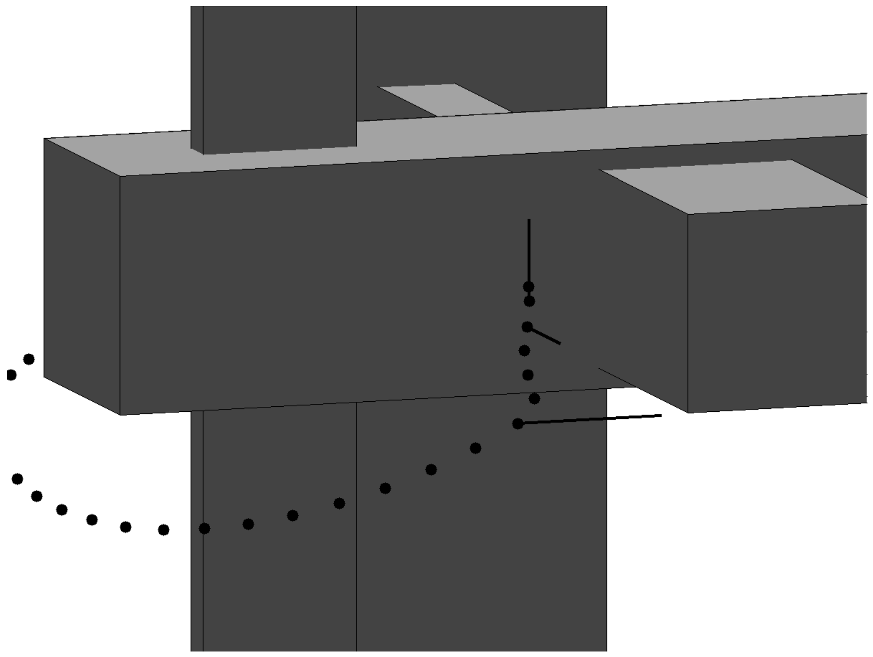


Figure 2-11: Final Stages of Maneuver from Fig. 2-10, shown in Close-Up from Below.

## 2.5 Computational Issues

The constraints for collision and plume avoidance can introduce large numbers of binary variables into a problem. The number of time steps is usually fixed by the size of the obstacles: if the time step is too long, a vehicle might pass right through an obstacle and still satisfy avoidance constraints on either side. This can lead to large increases in computation time in some cases. This section describes approximations that can be used to simplify the problems in order to reduce the solution time. The most useful approach is to use prior knowledge to identify redundant or inactive constraints. These can then be removed from the problem, which typically leads to a faster solution time. Two strategies are presented for identifying these removable constraints, each well-suited to a particular class of problems.

### 2.5.1 Normalization

The constraints shown so far can involve quantities of very different magnitudes. For example, separation distances can be on the order of hundreds of meters while thrust inputs may be a few micro-Newtons. To improve the numerical conditioning of the problem, the variables are normalized. New normalized states $\hat{\mathbf{x}}$ and inputs $\hat{\mathbf{u}}$ are related to the full-scale quantities by

$$\mathbf{x} = \mathbf{X}\hat{\mathbf{x}} \tag{2.23}$$

$$\mathbf{u} = \mathbf{U}\hat{\mathbf{u}} \tag{2.24}$$

where $\mathbf{X}$ and $\mathbf{U}$ are diagonal matrices of scaling factors equal to the original upper bounds in Eqs. 2.5 and 2.6. The new state and input bounds are given by

$$-1 \le \hat{x}_{\text{ipn}} \le 1 \tag{2.25}$$

$$-1 \le \hat{u}_{\text{ipn}} \le 1 \tag{2.26}$$

Other position quantities such as the obstacle definitions $\mathbf{L}$ and $\mathbf{U}$ and the plume scales $P$ and $W$ are scaled by the same factors. As a result, all of the decision variables are of the same order of magnitude, because the binary variables lie between 0 and 1, by definition. This modification has been found to make significant improvements in computation time in some cases, but no difference in others. While the reason for the improvement is not clear, it appears to be a result of an improvement in the efficacy of the heuristics employed by the CPLEX software to select branching nodes and directions.

### 2.5.2 Removal of Plume Constraints During Coast

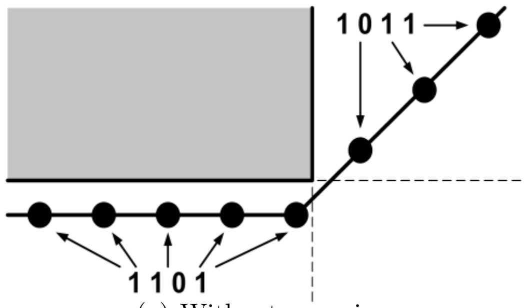
In constrained-input, minimum-fuel problems of the type in Eqn. 2.9, the optimal solution can be shown to consist of firing at the beginning and end of the trajectory, separated by a coasting phase. This is more commonly known as a "bang-off-bang" trajectory [37]. In certain problems, where the maneuvering space is large compared to the avoidance regions, it is possible to predict that the optimal solution including avoidance constraints will still be of a "bang-off-bang" form. Therefore, during the coasting phase, the plume constraints would be inactive.

This prior knowledge can be exploited by omitting some of the plume constraints during the anticipated coasting phase before solving the problem. This reduces the number of binary variables and typically leads to a faster solution time. However, it is then necessary to verify that no plume impingement occurred at the steps where the constraints were removed. This can be done quickly, and if no impingement is found, the result is also the optimal solution to the completely constrained problem. Note that this behavior exemplifies an $\mathcal{NP}$-complete problem: the feasibility of a candidate solution can be verified in polynomial time, but the global optimum cannot necessarily be found in polynomial time. If impingement is found to occur, it would then be necessary to include some of the removed constraints and solve again. This leads to an iterative solution process, but this is often still faster than solving the complete, global problem. Similar concepts have been applied to scheduling problems in Operations Research [59] using MILP and Constraint Programming (CP) in iterative schemes.

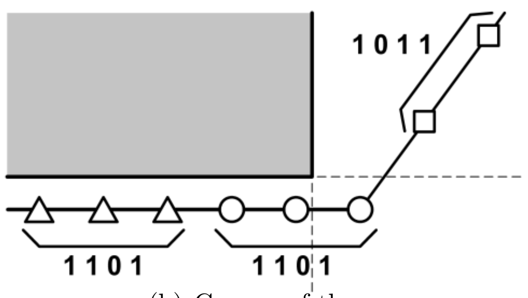
### 2.5.3 Time-Step Grouping.

For problems where the avoidance regions are not small compared to the maneuvering space, there is likely to be extensive interaction between vehicles, and it is not likely that the trajectory will be of the "bang-off-bang" type. However, as vehicles and obstacles move past each other, the interactions typically last for several time-steps at least, due to the comparatively large avoidance regions. Consequently, the binary variable settings are likely to be equal for sequences of time steps. To make use of this prediction, binary variables are "shared" across small groups of adjacent time-steps. This can be viewed as the inclusion of additional constraints, equating the binary variables in the groups. This is expected to increase the cost of the maneuver. However, since the binary variables were likely to be equal in the original problem, the additional constraints are likely to be satisfied by the solution to the original problem, hence the cost penalty is expected to be small.

Fig. 2-12 illustrates the effect of grouping time steps for obstacle avoidance. Fig. 2-12(a) shows part of the solution to a problem with individual sets of binary variables for each time step. The settings for the binary variables at each time step for avoiding the obstacle, corresponding to the variables $a_{\mathrm{k}}$ in Eqn. 2.11, are marked on the figure. Notice that sequences of time steps have equal settings, since the steps are small compared to the obstacle. Fig. 2-12(b) shows the new solution found with time step grouping. The group marked by $\triangle$ had the same settings in the original solution, and are therefore unchanged. The group marked by $\bigcirc$ runs across the join of two sequences from the original problem. The right-most point in the group is moved, such that it satisfies the same avoidance constraint as the other two. Since the time step is small compared to the obstacle, this change is likely to be small in terms of the whole maneuver. Therefore, it is possible to obtain a solution very close to the original, ungrouped problem with a third the number of binary variables. The effect of this reduction on computation is shown in Section 2.5.4



(a) Without grouping



(b) Groups of three

Figure 2-12: Illustration of Time Step Grouping. Time steps in the same group have the same symbol. The binary settings ($a_1a_2a_3a_4$) from Eqn. 2.11 for each step or group are shown.

### 2.5.4 Example: ISS Remote Camera - Computation

The ISS camera example, seen in Section 2.3.5, is extremely demanding in terms of computation. Since some of the obstacles are thin panels, it is necessary to use a short time-step to ensure that the vehicle cannot "jump" straight through an obstacle between time-steps and at least 40 time-steps are needed. Although there is only one vehicle, collision avoidance in three dimensions considering five obstacles requires some 30 binary variables per time-step. To solve for collision avoidance alone involves roughly 1200 binary variables, but this problem can be solved in approximately 8.0 seconds. Adding plume impingement requires a further 210 binary variable per time-step forming a problem with 9600 binary variables. Since the binary variable search space is so large, it is impractical to compute the global optimal solution to the full problem.

This problem is well-suited to the "time-step grouping" technique described in Section 2.5.3. The camera satellite moves between 0.7 and 1 units per time-step, while the obstacles are roughly tens of units across. Table 2.2 compares the results using groups of different sizes. The top row shows the results for the original problem without grouping. It can be seen that using groups of three time-steps reduces the computation time by a factor of at least 30 at the expense of only a 2% increase in fuel use. This additional conservatism was expected, due to the more constrained

Table 2.2: Results for ISS Problem with Plume Constraints

| Time-step grouping size | Computation time (seconds) | Fuel cost as $\Delta V$ (m/s) |
|---|---|---|
| None | 1800 | 0.2692 |
| 2 | 190 | 0.2727 |
| 3 | 54 | 0.2746 |
| 4 | 67 | 0.2864 |

nature of the problem when grouping is used. Note that increasing the group size beyond three actually causes a slight increase in the solution time.

Further experiments have been performed to find the variation of the solution time with the particular problem instance, and also to investigate the effect of normalization, as discussed in Section 2.5.1. Problems were solved with 60 different, randomly-generated start and end positions around the station, using time-step groups of length three. Fig. 2-13 shows the cumulative distribution of computation times. It can be seen that normalization gives a significant improvement in computation time, and that most problems can then be solved in under 100 seconds.

### 2.5.5 Example: ISS Rendezvous using Iterative Scheme

This section demonstrates the technique of removing the plume constraints during the expected coasting period, leading to an iterative solution procedure. The example involves an autonomous supply spacecraft performing a rendezvous with the ISS. Unlike the remote camera in the previous example, this maneuver starts much further away from the station. Therefore, the action of the plume constraints is expected to occur only at the end of the maneuver when the chaser spacecraft gets close to the station. Fig. 2-14 shows the designed trajectory for approaches from a particular point. Note that the approach is designed to avoid impinging on the final braking thrust. The starting point is on a neighboring orbit to the station, following a closed-form ellipse in the Hill's frame. The trajectory is designed to reach the point marked by the cross, at which point a different control scheme would execute the final docking

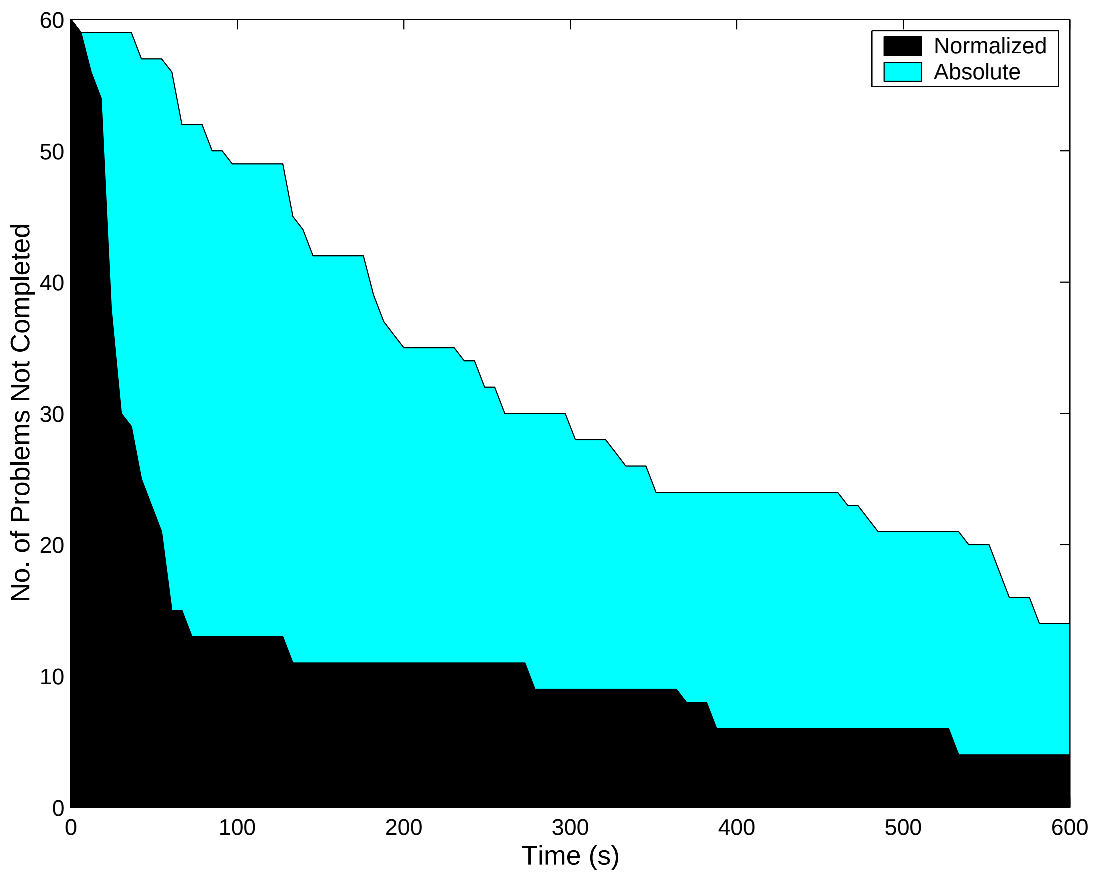


Figure 2-13: Distribution of Solution Times for Random ISS Camera Problems

maneuver.

Fig. 2-15 compares solution times using the iterative procedure versus a single optimization with all the constraints in place from the start. The times for the iterative procedure include that for checking trajectories and repeated solutions. Trajectories were designed for approach from different starting points on the ellipse shown in Fig. 2-14. For most of the cases, the iterative procedure is much faster than the single optimization, and the majority of problems are solved in under a second.

Further investigation was performed into those cases which proved difficult to solve by either method. Problems starting from around $150^o$ or $330^o$ take up to 30 seconds when the rest are all solved in under ten. Fig. 2-16 shows the designed trajectories for points on either side of each of these spikes. The spikes of slow computation occur where there is a cross-over from one approach “strategy” to another, corresponding to different binary variable settings. For example, when approaching from $120^o$, the

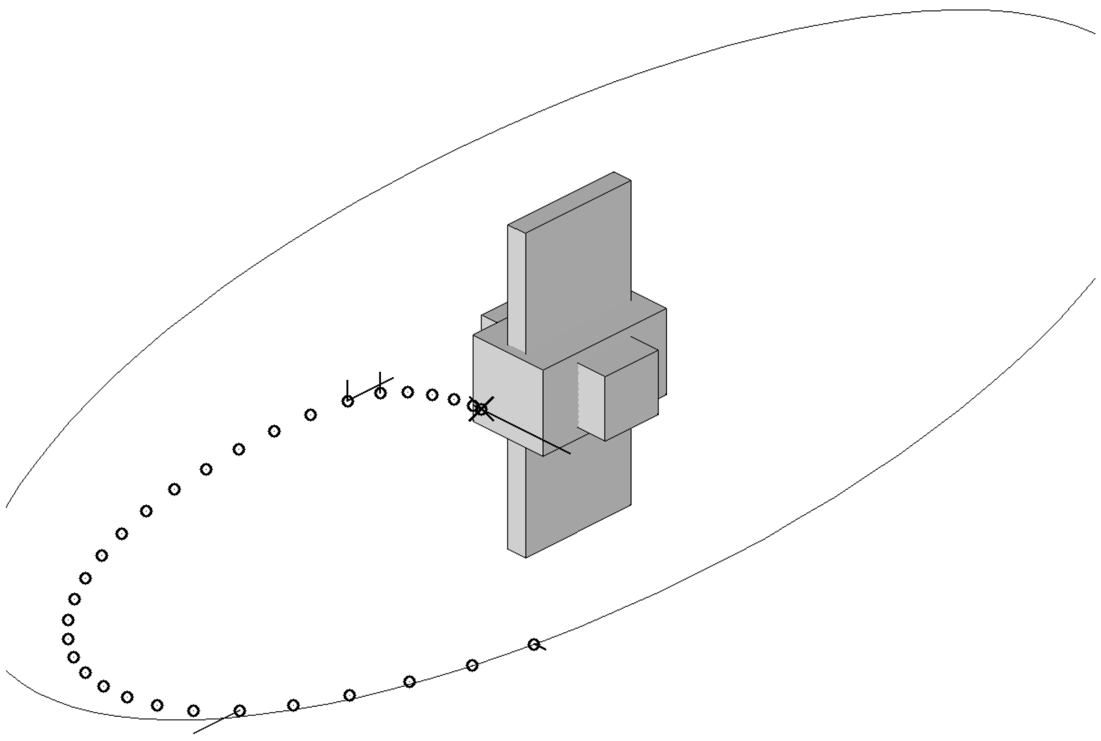


Figure 2-14: ISS Rendezvous Maneuver with Plume Avoidance

chaser craft always remains “in front” of the station, as seen in the plot, but from $170^o$, it passes behind and approaches its target in the opposite direction. Somewhere between these two starting angles will be a point from which each strategy has the same cost, and the branch-and-bound algorithm must fully evaluate both trajectories before terminating. Moving away from this point, one strategy will become more favorable, and the algorithm can discount the other option earlier, when the lower bound of its cost shows it to be unfavorable. A similar change in strategy can be seen between $300^o$ and $350^o$.

This section includes two key demonstrations. First, that iterative schemes based on prior knowledge offer substantial savings in computation time. Second, that some problems with multiple solutions having similar costs are harder to solve than other problems.

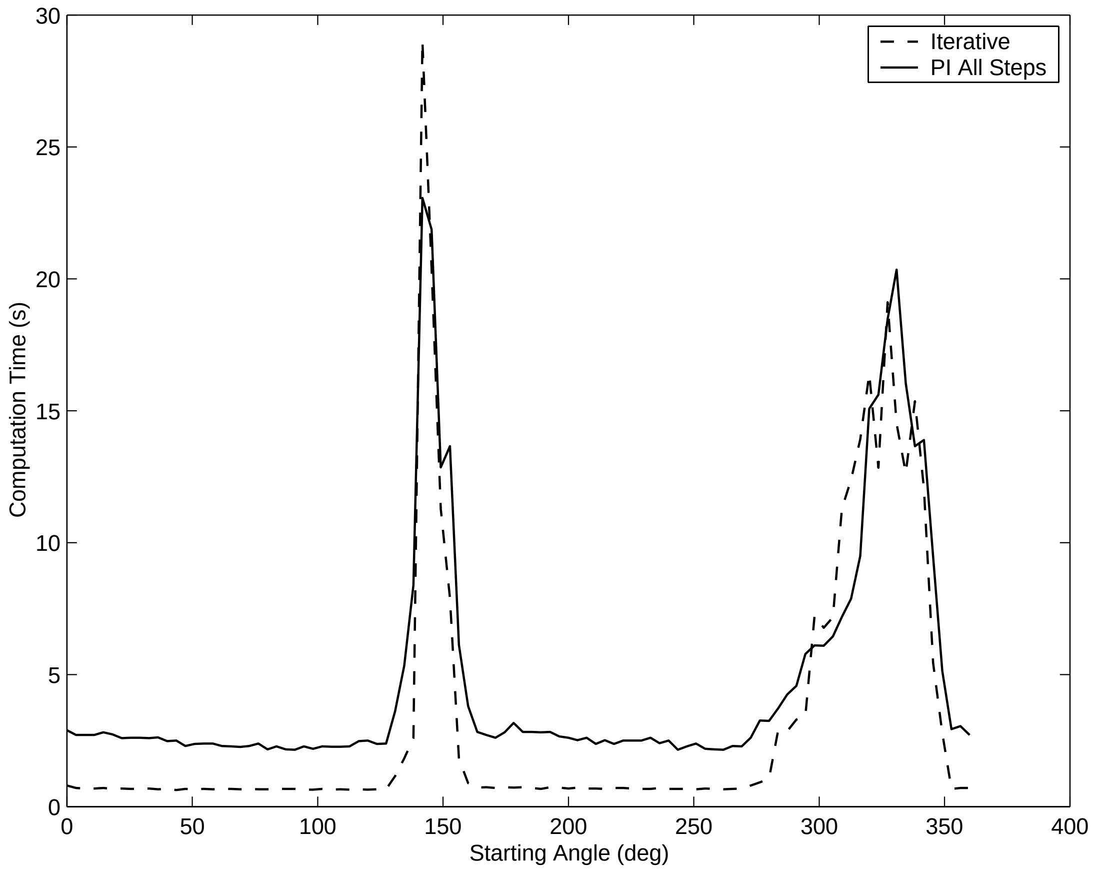
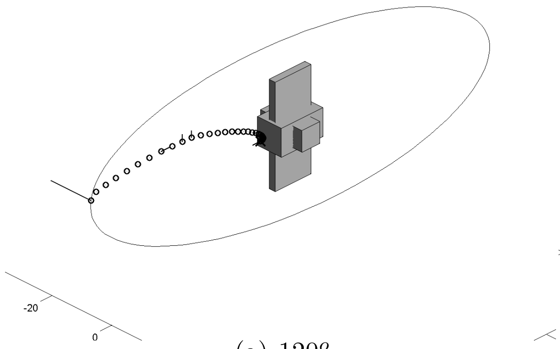
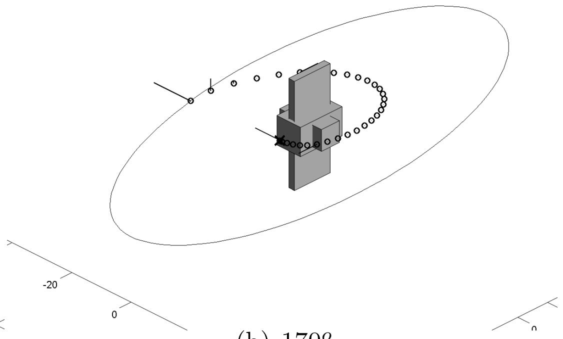


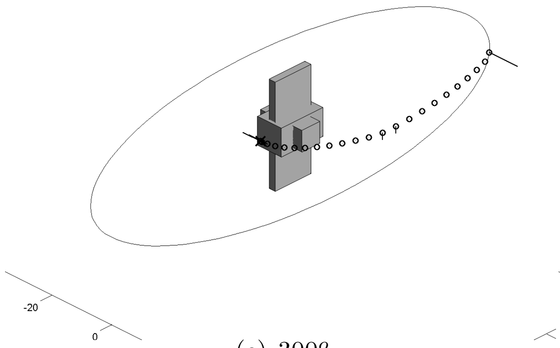
Figure 2-15: Variation of Computation Time for Rendezvous Maneuver with Starting Point. Iterative solution and single optimization technique are compared.



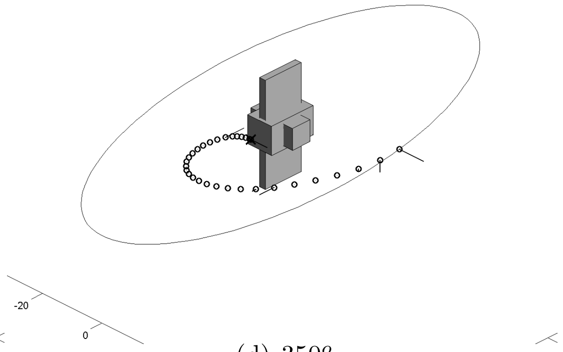
(a) $120^o$



(b) $170^o$



(c) $300^o$



(d) $350^o$

Figure 2-16: ISS Rendezvous Maneuvers from Various Starting Angles

## 2.6 Selection of Start and Finish Conditions

In this section, the boundary conditions for the maneuvers, either the terminal or initial constraints, are modified to include some form of assignment in the problem. Typically, the aim is to choose the boundary conditions and design the path for the minimum-fuel maneuver. The difficulty is that the assignment and path-planning problems are intrinsically coupled. If the costs for all possible assignments were known, the assignment would be straightforward, but the cost computation would be excessive. Other approaches to these problems, discussed in the Introduction Section 1.3.2, use approximate methods to generate costs of many possible maneuvers and then perform a simple assignment. In contrast, the method presented in this section captures both the path-planning and assignment in a single optimization, retaining their inherent coupling.

### 2.6.1 Final Configuration Selection Constraints

The constraint in Eqn. 2.2 enforces a fixed final state for each vehicle. This section generalizes that constraint to the case where each vehicle is assigned a specific final state from a set of possible alternatives [35]. A subset of final states, known as a "global configuration" is selected and spacecraft are assigned to positions within that subset. The selection and assignment are performed within the trajectory optimization to achieve the lowest overall fuel cost. For example, it might be required that the satellites reconfigure so that they are evenly spaced around a given ellipse, forming a passive aperture for a particular interferometry observation [16]. If the spacecraft are assumed to be identical, their ordering around the ellipse is not important. In addition, the rotation of the whole formation around the ellipse is not important. In the MILP formulation, the ellipse is discretized into a set of possible rotation angles for the formation. Each of these is entered as a global configuration, containing the final state for each spacecraft with the formation at that angle. When the resulting MILP is solved, the formation angle and the assignment of spacecraft around the formation are selected within the optimization to give the minimum fuel use in the reconfiguration maneuver.

The final state constraints for this formulation can be thought of as an extensive *or*-expression, in which the final states of the vehicles must be one of the available global configurations $g$ and the vehicles can be distributed in any one of the possible permutations across the terminal states. In more concise terms

$$\forall p: \quad \mathbf{x}_{\mathrm{Tp}} = \mathbf{x}_{\mathrm{F}_{\mathrm{r}}^{\mathrm{g}}} \text{ for some } g \in [1 \ldots G] \tag{2.27}$$

where $r$ is the unique position within the formation assigned to vehicle $p$. Using binary variables, these constraints can be expressed as

$$\forall p : \mathbf{x}_{\mathrm{Tp}} = \sum_{\mathrm{g=1}}^{\mathrm{G}} \sum_{\mathrm{r=1}}^{\mathrm{V}} \mathbf{x}_{\mathrm{F}_{\mathrm{r}}^{\mathrm{g}}} f_{\mathrm{pgr}} \tag{2.28}$$

where binary variable $f_{\mathrm{pgr}} = 1$ if vehicle $p$ takes the $r^{\mathrm{th}}$ position within the $g^{\mathrm{th}}$ global configuration and 0 otherwise. It is then necessary to place the following logical constraints upon these variables.

$$\begin{array}{rcccl} \forall p & : & \sum_{g} \sum_{r} f_{\mathrm{pgr}} & = & 1 \\ \forall g, \forall p & : & \sum_{r} f_{\mathrm{pgr}} & = & \sum_{r} f_{\mathrm{rgp}} \\ \forall g & : & \sum_{p} \sum_{r} f_{\mathrm{pgr}} & = & V \sum_{r} f_{1gr} \end{array} \tag{2.29}$$

The first constraint ensures that every satellite $p$ chooses exactly one position. The second constraint ensures that different satellites move to different positions $r$. The third constraint then ensures that all the chosen positions belong to the same global configuration $g$. The right-hand side of this equation equals 0 or $V$, the number of satellites.

## 2.6.2 Example: Satellite Formation Reconfiguration

In this example, a group of satellites is reconfigured to be evenly spaced around an ellipse of a given size and inclination. In an interferometry application, this would correspond to a change in aperture, as discussed previously. The satellites are identical, so the assignment within the formation and the overall rotation of the ellipse can be selected in the optimization using the terminal constraints from Eqs. 2.28 and 2.29. This example will also be used to demonstrate the computation times involved with various numbers of vehicles and different constraint forms. Results are shown for cases involving two, three and four vehicles, with and without plume impingement.

The vehicles are initially arranged along a line in-track. Each satellite is modeled as a point mass of 50 kg, with Hill's equations as the relative dynamics using a 90 minute orbit. The final configuration selection is discretized such that the ellipse is divided into 30 possible global configurations, each containing one position for each spacecraft. The maneuver must be performed in 1000 seconds, equivalent to just over 15 minutes or 1/6 of an orbit. The maneuver is discretized into 25 time-steps each of 40 seconds. The plume exclusion box is 100 m long and is 20 m by 20 m square in cross-section ($P = 100, W = 10$).

Fig. 2-17 shows the designed trajectories for the two-, three- and four-vehicle cases, with and without plume constraints. The ellipse associated with the final aperture is also shown. Note that in the three- and four-vehicle cases, the chosen assignment is changed when plume constraints are included. This demonstrates that the inclusion of path-planning and assignment in a single optimization can fully model the coupled problem and offer fuel savings. Table 2.3 shows the computation time and fuel use in each case. Adding plume constraints on all steps causes a considerable increase in computation time, as shown in the second row of Table 2.3. While the computation times for two- and three-vehicle cases are still shorter than the maneuver time, the four-vehicle case now takes over half an hour to compute, which is approximately twice as long as the maneuver itself. As in the ISS example, this complexity arises from the number of binary variables in the problem. For example, the four-vehicle case involves 900 binaries for collision avoidance, 480 for configuration selection, and 9000 for plume avoidance.

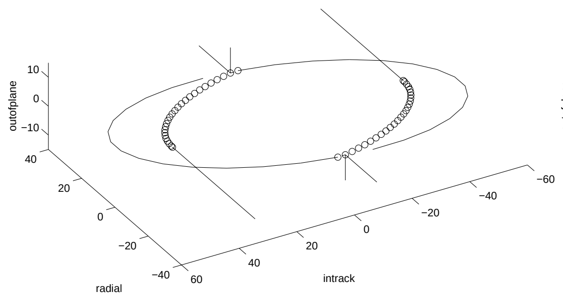
This issue with the computation time can be addressed by recognizing that this problem is likely to result in a "bang-off-bang" trajectory, since the vehicle avoidance

Table 2.3: Results for Reconfiguration Maneuver, showing computation time (in seconds) and fuel (as $\Delta V$ in m/s) and $V$ =Number of vehicles.

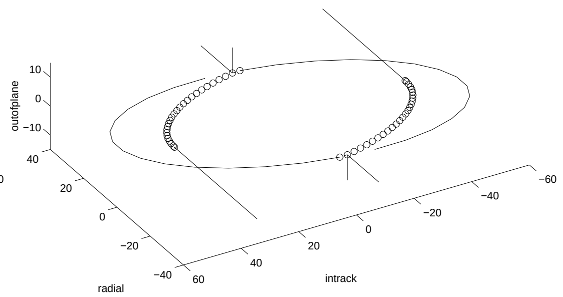
| | $V = 2$ | | $V = 3$ | | $V = 4$ | |
|---|---|---|---|---|---|---|
| | Time | Fuel | Time | Fuel | Time | Fuel |
| CA only | 1.0 | 4.67 | 7.8 | 9.82 | 28 | 13.18 |
| With PI | 6.8 | 4.67 | 170 | 10.20 | 1900 | 13.84 |
| PI on first ten steps | 2.0 | 4.67 | 33 | 10.20 | 640 | 13.84 |
| PI on all steps, groups of three | 4.8 | 4.67 | 100 | 10.35 | 2500 | 14.41 |

regions are small compared to the maneuvering distances. Therefore, the removal of plume constraints in the coast phase, as discussed in the computation section, can be used to reduce solution time. Also, the final positions are much further apart and not aligned with each other, so impingement is not expected to occur at the end of the maneuver. The third row of results in Table 2.3 were obtained by preventing plume impingement only on the first ten time-steps. A post-analysis of the trajectories showed that plume impingement did not occur at the steps from which the constraints had been removed. The significant reduction in solution time with the removal of redundant constraints demonstrates the benefit of this technique, as discussed in Section 2.5.2.

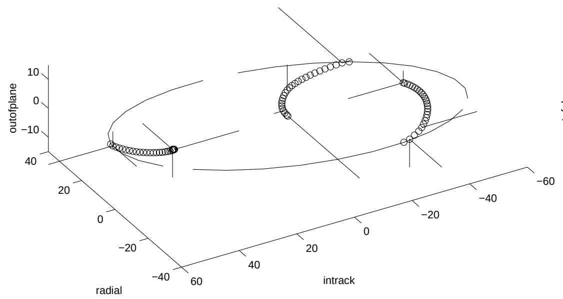
The fourth row shows the results from the application of the time-step grouping idea to this problem. This technique is not expected to be very effective here, since the avoidance regions are small compared to the maneuvering space and the interaction between vehicles is short-lived. In this case, plume impingement was prevented on all steps, but the binary variables were shared across groups of three adjacent time-steps. The fuel costs are slightly higher than those in the second row, as expected from the grouping method. The computation times are slightly reduced for the two- and three-vehicle cases, but increased in the four vehicle case. This contrasts with the ISS example, where grouping made the most complicated case solve more quickly. This demonstrates the difference between the two approximation techniques and shows that they are dependent on the problem characteristics. More generally, the inclusion of prior knowledge for simplification is problem-specific.



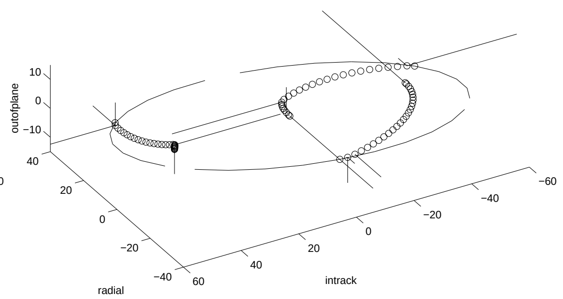
(a) $V = 2$, no PI



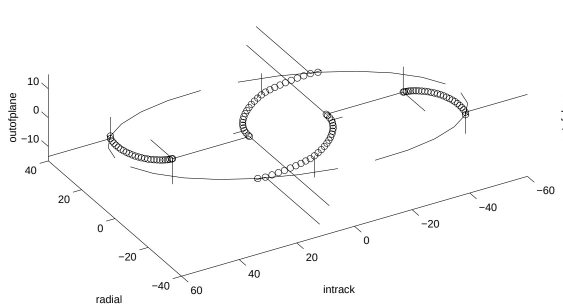
(b) $V = 2$, with PI



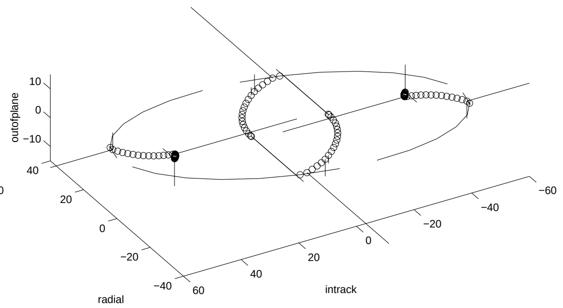
(c) $V = 3$, no PI



(d) $V = 3$, with PI



(e) $V = 4$, no PI



(f) $V = 4$, with PI

Figure 2-17: Reconfiguration Maneuvers with Varying Number of Vehicles $V$, with and without Plume Impingement Constraints (PI)

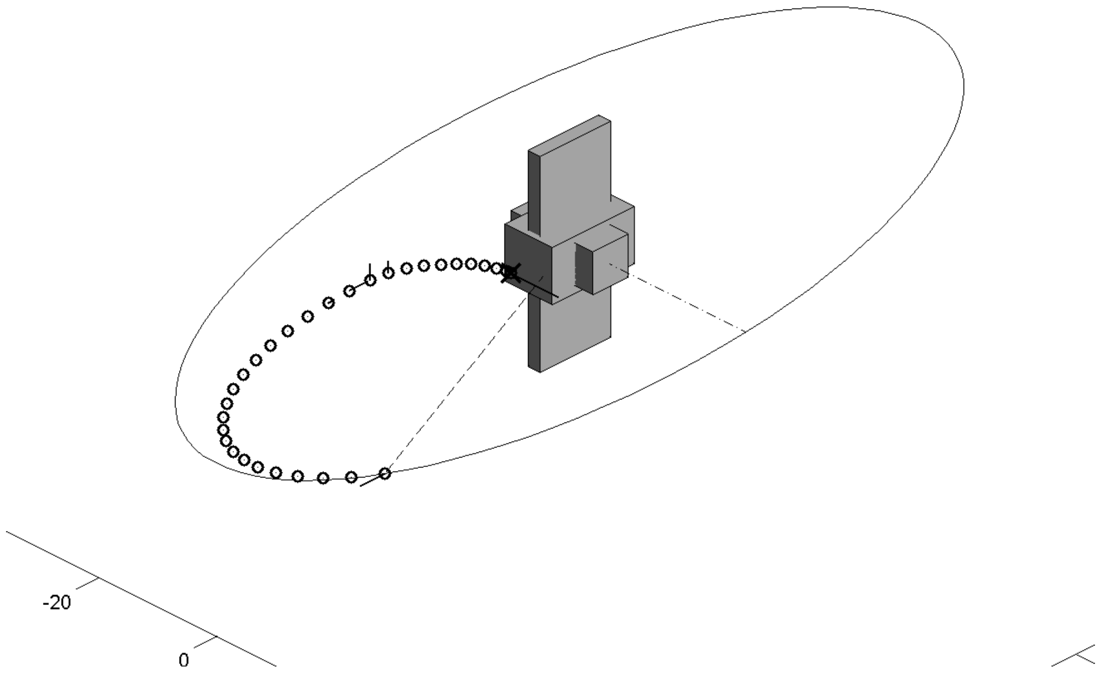


Figure 2-18: ISS Rendezvous Maneuver with Starting Point chosen to Minimize Fuel Use

### 2.6.3 Example: ISS Rendezvous with Variable Starting Point

This section demonstrates the use of variable starting positions. The formulation for these is analogous to the variable finishing states in Eqs. 2.28 and 2.29. The example is that of the ISS rendezvous spacecraft as introduced in Section 2.5.5. The vehicle starts on a closed-form ellipse around the space-station. In this section, the optimization may choose the starting state from a selection of points on that ellipse, such that the following rendezvous maneuver is accomplished with minimum fuel. This choice is equivalent to having a variable starting time for the rendezvous. The ellipse is discretized into 100 possible starting points, and the optimal maneuver is shown in Fig. 2-18.

Fig. 2-19 shows the variation in fuel use with starting point, found by solving the fixed starting point problem from each of the discrete points on the ellipse. The

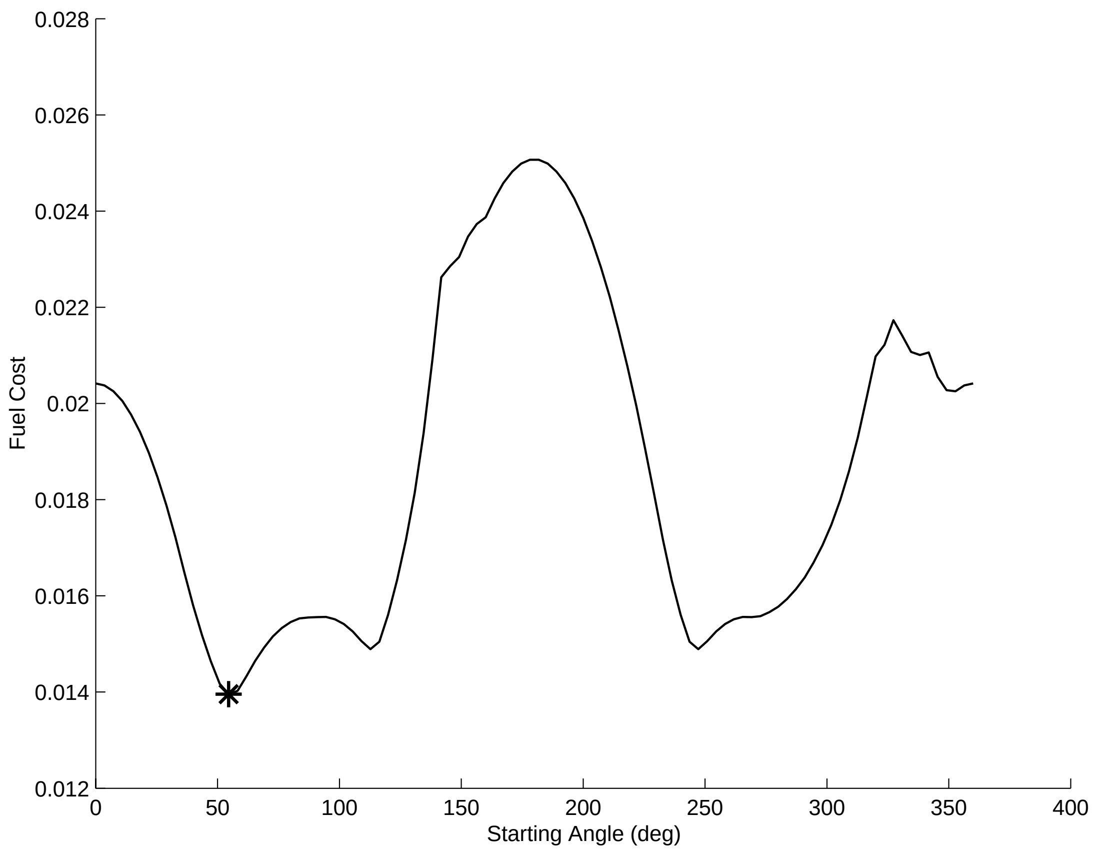


Figure 2-19: Variation of Fuel Use with Starting Point for ISS Rendezvous Maneuver. The starting point chosen by the optimization in Fig. 2-18 is marked by $*$

starting point is parameterized by the angle on the ellipse, with the zero-degree line marked by the dash-dot line in Fig. 2-18. The star marked on Fig. 2-19 indicates the starting point chosen by the optimization with variable initial conditions. As expected, it has chosen the global optimum.

## 2.7 Summary

This chapter has shown that various spacecraft avoidance problems can be solved for the minimum-fuel maneuvers using MILP. In particular, collision avoidance and plume avoidance have been enforced, involving multiple spacecraft and space stations. While the optimizations are complicated to solve, techniques have been presented to allow solutions to be obtained in practical times. These include grouping time steps for avoidance enforcement and using iterative schemes to exploit prior knowledge of the solution. The problems can be extended to include assignment, in which both the paths and destinations are chosen for minimum fuel use.

# Chapter 3

# Aircraft Applications

This chapter extends the work in Chapter 2 to solve problems involving aircraft, in particular multiple UAVs. Two major developments are involved: the first is the inclusion of a linear model of aircraft dynamics in the MILP; the second involves general forms of terminal constraints to include waypoint assignment in the optimization. These are considered in Sections 3.3 and 3.4 respectively. The final formulation is able to solve the combined path-planning assignment problem, capturing the inherent coupling of these two problems in a single optimization.

## 3.1 Nomenclature for Aircraft Problems

The following quantities define the size of problems in this Chapter

| | |
|---|---|
| $N_V$ | Number of vehicles |
| $N_W$ | Number of waypoints |
| $N_Z$ | Number of exclusion zones |
| $N_D$ | Number of time dependencies |

The following standard subscripts are used:

| | |
|---|---|
| $t$ | Time-step |
| $p, q$ | Aircraft |
| $j$ | Exclusion zone |
| $i$ | Waypoint |
| $k$ | Time dependency |

The following variables are used:

| | |
|---|---|
| $(x, y)$ | Position |
| $(\dot{x}, \dot{y})$ | Velocity |
| $\mathbf{s}$ | State vector |
| $\mathbf{f}$ | Force vector |

Table 3.1 shows the parameters used to specify the problems.

Table 3.1: Parameters of General Aircraft Problems

| Parameter | Size | Meaning |
|---|---|---|
| $\mathbf{v}_{\max}$ | $N_V \times 1$ | Maximum speeds of aircraft |
| $\omega_{\max}$ | $N_V \times 1$ | Maximum turn rates |
| $\mathbf{S}$ | $N_V \times 4$ | Initial states |
| $\mathbf{W}$ | $N_W \times 2$ | Waypoint positions |
| $\mathbf{Z}$ | $N_Z \times 4$ | Exclusion zones |
| $\mathbf{K}$ | $N_V \times N_W$ | Aircraft capabilities |
| $\mathbf{\Delta}$ | $N_D \times N_W$ | Time dependencies |
| $\mathbf{t}_D$ | $N_D \times 1$ | Time dependency intervals |
| *Additional Parameters for Scores and Penalties* | | |
| $\mathbf{V}$ | $N_V \times N_W$ | Waypoint scores (replaces $\mathbf{K}$) |
| $\mathbf{P}$ | $N_V \times N_Z$ | Exclusion zone penalties |

## 3.2 Problem Statement

This section gives a more detailed illustration of the problem statement for the aircraft case, motivating the formulations that follow. Fig. 3-1 illustrates a scenario for a typical UAV problem. In all cases, it is required to plan a trajectory for each vehicle, obeying its specified dynamics constraints. Three problems are solved for this scenario.

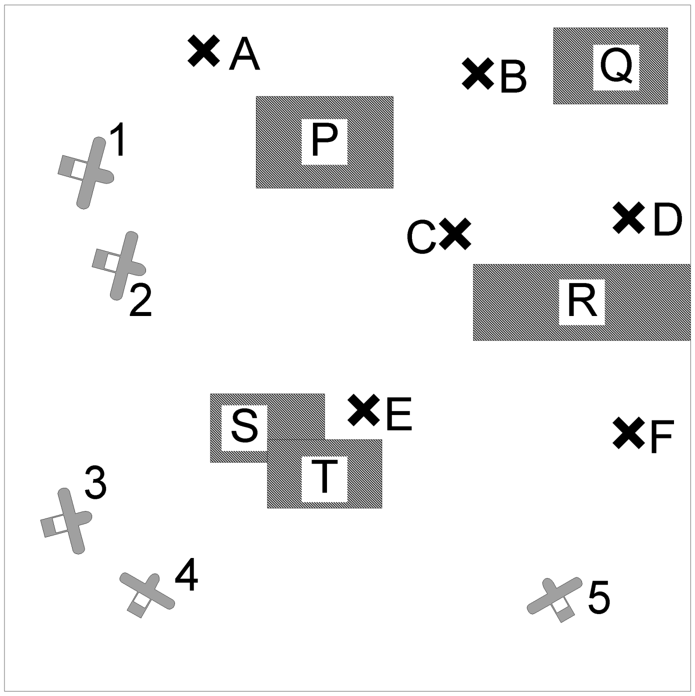


Figure 3-1: Example Scenario for Aircraft Problem

**(A)** Each vehicle has an assigned destination, *e.g.* vehicle 1 to point A, vehicle 2 to point B, and so on. The optimization must find the minimum-time solution for each vehicle avoiding the obstacles and other vehicles. This problem could represent conflict resolution in air traffic control.

**(B)** The vehicles are not given pre-assigned destinations. Each has specified capabilities, *e.g.* vehicle 1 may visit waypoints A, B and D, vehicle 2 may visit waypoints B, C and E, and so on. There may also be timing constraints, such as requiring waypoint A to be visited before D. The optimization seeks the solution for the minimum mission completion time. In addition, extra waypoints may be added such that there are more waypoints than vehicles, and then the optimization chooses the order of visits for those vehicles assigned more than one waypoint. This represents a planning problem for a fleet of UAVs.

**(C)** The final variation extends the assignment to include scores for each waypoint.

For example, vehicle 1 would score 20 points for visiting point A, 30 for B, none for C, and so on. Each 'point' is equivalent to the cost of flight for one time step, since flight time is also penalized. Also, the obstacles are relaxed to become 'penalty zones' representing some hazard. Vehicles incur penalties for entering the regions, *e.g.* vehicle 1 would lose 20 points for entering zone P, 100 points for entering Q, and so on. The optimization assigns the vehicles and designs the trajectories for the highest overall score. This represents a more general form of UAV mission planning including risks and values.

Section 3.3 develops the linear aircraft model and verifies it using examples of problem A. Section 3.4 extends the formulation for the scenarios in B and C.

## 3.3 Modeling Aircraft Dynamics in MILP Form

MILP captures the non-convexity of avoidance problems while enabling globally-optimal solutions to be computed. The expense is the necessity for the problem to be completely linear. This section develops a linearized model of aircraft dynamics that can be used in this framework.

This Chapter considers problems in which aircraft fly at constant altitude, resulting in planar motion. This is a common restriction in air traffic models, as air space is usually structured in layers [40]. Also, in UAV problems, altitude is often determined by mission constraints, such as sensor resolution or radar visibility, resulting in a 2-D guidance problem. The MILP formulation can readily be extended to 3-D problems, as shown in Chapter 2, but for simplicity, this Chapter will consider only 2-D problems.

For many cases of interest, an aircraft can be modeled as moving at constant speed. The rate of change of heading angle is limited by the maximum bank angle of the aircraft. Writing these constraints exactly results in nonlinear expressions, which cannot be handled in a MILP framework. Section 3.3.1 shows that the dynamics can be approximated by a point mass with limited speed and force actuation. Section 3.3.2

then shows the linear constraints used to include this model. Section 3.3.3 shows the inclusion of avoidance constraints from Chapter 2. Section 3.3.4 shows how to solve for the minimum time solution, which favors solutions remaining at the maximum speed and therefore returns the minimum distance solution.

### 3.3.1 Approximation of Vehicle Dynamics

This section shows that an aircraft flying at constant altitude can be approximated by a point mass, moving with limited speed and acted upon by a force of limited magnitude. The turning rate constraint is effected by a force magnitude limit. Consider a point mass $m$ traveling with speed $v$ subject to a force of magnitude $f$. The instantaneous turning rate $\omega$ will be greatest if the force is perpendicular to the velocity, causing the vehicle to follow a circular path. It is therefore limited by

$$\omega \leq \frac{f}{mv} \tag{3.1}$$

Furthermore, if the magnitude of the force is limited to $f_{\max}$ and the speed is a constant $v_{\max}$, the rate is limited throughout the problem by

$$\omega \leq \omega_{\max} = \frac{f_{\max}}{mv_{\max}} \tag{3.2}$$

To rigorously constrain speed to remain at $v_{\max}$ would require non-convex constraints in the velocity plane, complicating the problem considerably. In the linear approximation, only an upper bound is included, which can be approximated by linear constraints. In this model, it is feasible for the speed to fall below $v_{\max}$, allowing tighter turns than the bound in Eqn. 3.2. However, the optimization seeks the minimum time solution, making it favorable to remain at maximum speed and obey the specified turn rate limit.

For example, consider the situation shown in Fig. 3-2, in which an aircraft must turn through $90^o$ from one straight line on to another. Two of the possible paths are shown in this figure. Following the solid line, the aircraft remains at maximum

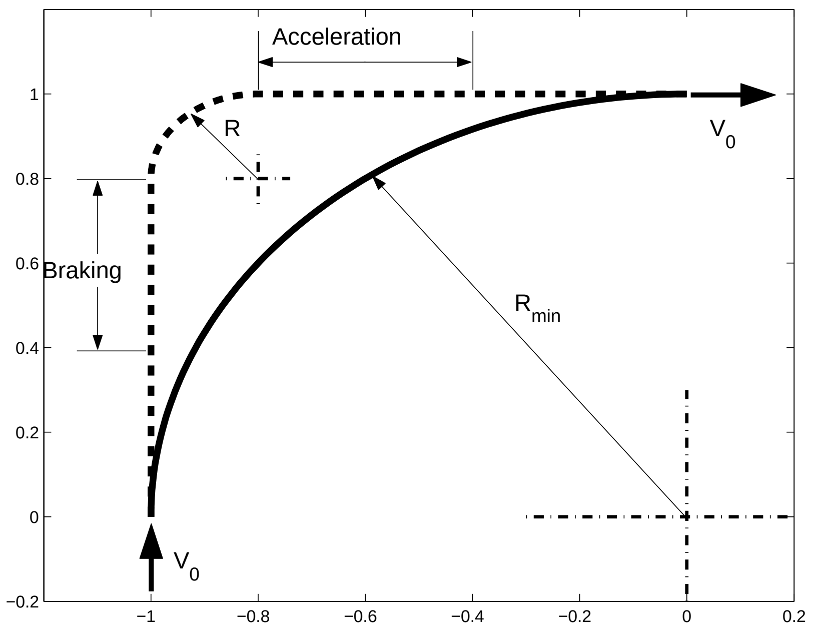


Figure 3-2: Feasible Cornering Paths for the Aircraft Model. $R_{\min}$ is the designed minimum radius of curvature. The dashed path is allowable within the constraints, but will always take longer than the solid path.

speed and turns at the prescribed maximum turning rate $\omega_{\max}$, therefore following the prescribed minimum radius $R_{\min}$. It is also feasible to follow the dashed path, decelerating first, then applying the maximum force to achieve a smaller radius of curvature, before accelerating back to maximum speed and rejoining the solid path. Although it is allowed in the linear model, this trajectory exceeds the nominal turning rate limit. Fig. 3-3 shows the variation in total maneuver time with the rate of turn used. This was found analytically by calculating the duration of each turn, the necessary deceleration and acceleration periods, and the adjoining periods of maximum speed travel. The figure shows that the fastest turning maneuver is achieved by remaining at maximum speed and obeying the nominal maximum turning rate $\omega_{\max}$. Using higher turning rates leads to a slower overall maneuver, due to the additional deceleration required. This result matches intuition, because the solid path in Fig. 3-2 is shorter in length than the dashed path and has a higher average speed. The optimization will return the solution following the solid line, which obeys the

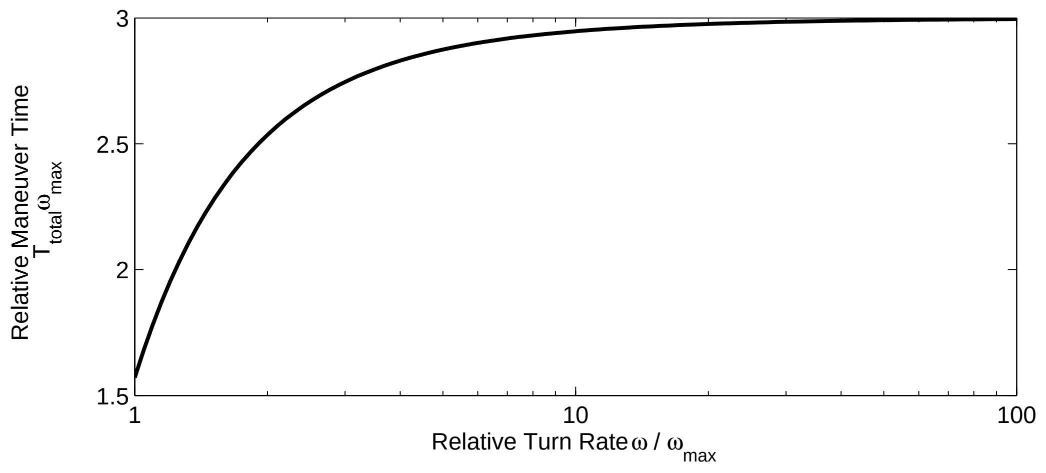


Figure 3-3: Variation of Duration with Turn Rate for the Maneuver in Fig. 3-2.

nominal turning rate limit and is therefore flyable for the real aircraft.

In conclusion, the optimization will always favor a path obeying the turn limit if such a path is available. When avoidance constraints are added, some arrangements of obstacles could cause the model to take a tighter turn. Therefore, it is necessary to post-analyze each trajectory to ensure it is flyable by the real aircraft. If not, the problem can be rerun with a lower force limit until an acceptable solution is found.

### 3.3.2 Dynamics Constraints

This section describes the constraints used to implement the model of aircraft dynamics described in Section 3.3.1. Let there be $N_V$ aircraft, each approximated as a unit point mass moving in 2-D free space. The position of aircraft $p$ at time-step $t$ is given by $(x_{tp}, y_{tp})$ and its velocity by $(\dot{x}_{tp}, \dot{y}_{tp})$, forming the elements of the state vector $\mathbf{s}_{tp} = (x_{tp}, y_{tp}, \dot{x}_{tp}, \dot{y}_{tp})^{\mathrm{T}}$. Each aircraft is assumed to be acted upon by control forces $(f_{x_{tp}}, f_{y_{tp}})$ in the $X$- and $Y$-directions respectively, forming the force vector $\mathbf{f}_{tp}$. The discretized dynamics of the point mass, for all $N_V$ vehicles at up to $N_T$ time-steps, can be written in the linear form

$$\begin{gathered} \forall p \in [1 \dots N_V] \, \forall t \in [0 \dots N_T - 1] \\ \mathbf{s}_{\mathrm{(t+1)p}} = \mathbf{A}\mathbf{s}_{\mathrm{tp}} + \mathbf{B}\mathbf{f}_{\mathrm{tp}} \end{gathered} \tag{3.3}$$

where **A** and **B** are the discretized system matrices. In all cases, the initial conditions are specified as

$$\mathbf{s}_{0\mathrm{p}} = \mathbf{s}_{\mathrm{p}}^{0} \tag{3.4}$$

where $\mathbf{s}_{\mathrm{p}}^{0}$ is the initial state of vehicle $p$ from the $4 \times N_V$ matrix $\mathbf{S} = [\mathbf{s}_1^0 \ldots \mathbf{s}_{N_V}^0]$.

As discussed in Section 3.3.1, both the velocity of the aircraft and the force acting upon the aircraft are subject to magnitude limits. The exact representation of these constraints would be nonlinear, but they can be approximated by linear inequalities. The true magnitude constraints enclose a circle on the $X$-$Y$ plane, as shown in Fig. 3-4. An arbitrary number of constraints ($M$) is used to approximate the circle. For both velocity and force constraints these are given by

$$\begin{aligned} &\forall t \in [0 \ldots N_T - 1]\, \forall p \in [1 \ldots N_V]\, \forall m \in [1 \ldots M] \\ &f_{\mathrm{xtp}} \sin\left(\frac{2\pi m}{M}\right) + f_{\mathrm{ytp}} \cos\left(\frac{2\pi m}{M}\right) \leq f_{\max} \end{aligned} \tag{3.5}$$

$$\begin{aligned} &\forall t \in [1 \ldots N_T]\, \forall p \in [1 \ldots N_V]\, \forall m \in [1 \ldots M] \\ &\dot{x}_{\mathrm{tp}} \sin\left(\frac{2\pi m}{M}\right) + \dot{y}_{\mathrm{ip}} \cos\left(\frac{2\pi m}{M}\right) \leq v_{\max} \end{aligned} \tag{3.6}$$

The sine and cosine values are passed to the optimization as a table of constants, and Eqs. 3.5 and 3.6 form a series of linear constraints. The feasible region formed by ten constraints ($M = 10$) is shown in Fig. 3-4, forming a good approximation to the circle. While this introduces many constraints, they only involve the continuous variables, so the computation time is not seriously affected.

### 3.3.3 Avoidance Constraints

Avoidance of both obstacles and other vehicles is enforced using the constraints from Section 2.3. They are presented here in 2-D form using the nomenclature of this Chapter.

For any pair of vehicles $p$ and $q$, let the safety distance for collision avoidance be

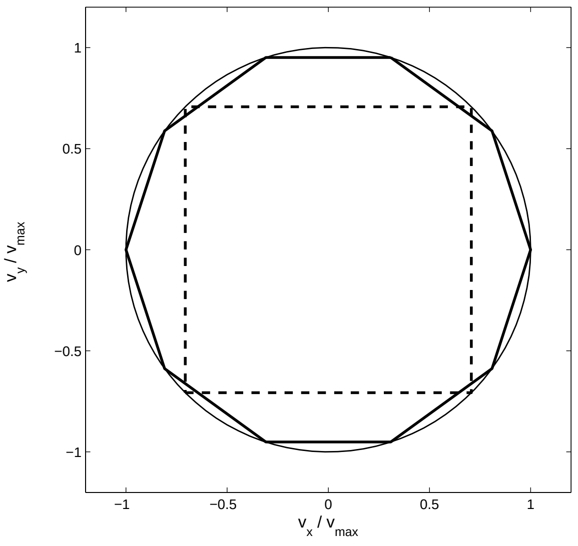


Figure 3-4: Approximations to Magnitude Limits for 2-D Vectors. The circle is the feasible region for true magnitude limits. The square and polygon are two ways of approximating these regions with linear constraints.

denoted by $r$, assumed to be the same in both $X$ and $Y$ directions for simplicity. The constraints for collision avoidance are

$$\begin{aligned} \forall t \in [1 \ldots N_T] \ \forall p, q \ | q > p \\ x_{\mathrm{tp}} - x_{\mathrm{tq}} &\geq r - Rc_{\mathrm{tpq1}} \\ \text{and} \quad x_{\mathrm{tq}} - x_{\mathrm{tp}} &\geq r - Rc_{\mathrm{tpq2}} \\ \text{and} \quad y_{\mathrm{tp}} - y_{\mathrm{tq}} &\geq r - Rc_{\mathrm{tpq3}} \\ \text{and} \quad y_{\mathrm{tq}} - y_{\mathrm{tp}} &\geq r - Rc_{\mathrm{tpq4}} \\ \text{and} \quad \sum_{k=1}^{4} c_{\mathrm{tpqk}} &\leq 3 \end{aligned} \tag{3.7}$$

where $c_{tpqk}$ are a set of binary variables (0 or 1) and $R$ is a positive number that is much larger than any position or velocity to be encountered in the problem. If $c_{tpqk} = 0$, there is at least $r$ distance between vehicles $p$ and $q$ in the $k^{\text{th}}$ direction (of the four directions $+X$, -$X$, $+Y$, -$Y$) at the $t^{\text{th}}$ time-step. If $c_{tpqk} = 1$, the constraint is relaxed. Eqn. 3.7 becomes an additional constraint on the trajectory optimization problem. The binaries $c_{\mathrm{tpqk}}$ become decision variables for the optimization.

Obstacles, or "exclusion zones" for UAV problems, are specified in the $(N_Z \times 4)$

matrix $\mathbf{Z}$, where $N_Z$ is the number of exclusion zones in the problem. $(Z_{\mathrm{j1}}, Z_{\mathrm{j2}})$ is the bottom left vertex of the $j^{\text{th}}$ zone and $(Z_{\mathrm{j3}}, Z_{\mathrm{j4}})$ is the top right vertex. The avoidance constraints can then be written as

$$\begin{array}{rrcl} & \multicolumn{3}{l}{\forall t \in [1 \ldots N_T] \ \forall p \in [1 \ldots N_V] \ \forall j \in [1 \ldots N_Z]} \\ & x_{\mathrm{tp}} - Z_{\mathrm{j3}} & \geq & -R d_{\mathrm{tpj1}} \\ \text{and} & Z_{\mathrm{j1}} - x_{\mathrm{tp}} & \geq & -R d_{\mathrm{tpj2}} \\ \text{and} & y_{\mathrm{tp}} - Z_{\mathrm{j4}} & \geq & -R d_{\mathrm{tpj3}} \\ \text{and} & Z_{\mathrm{j2}} - y_{\mathrm{tp}} & \geq & -R d_{\mathrm{tpj4}} \\ \text{and} & \sum_{k=1}^{4} d_{\mathrm{tpjk}} & \leq & 3 \end{array} \tag{3.8}$$

where $d_{tpjk}$ are a further set of binary variables and $R$ is the same large, positive number used in Section 3.3.3. Once again, if $d_{tpjk} = 0$, the $p^{\text{th}}$ vehicle is beyond the $j^{\text{th}}$ exclusion zone in the $k^{\text{th}}$ direction (of the four directions $+X$, -$X$, $+Y$, -$Y$) at the $t^{\text{th}}$ time step. Eqn. 3.8 becomes an additional constraint on the trajectory optimization problem. The binaries $d_{\mathrm{tpjk}}$ become decision variables for the optimization.

### 3.3.4 Solving for the Minimum Time Trajectory

This section describes the additional constraints and the cost function used to solve for the minimum-time solution. The optimization therefore returns the shortest flight path, and favors the solution with the widest turns, as discussed in Section 3.3.1. A concurrent development used a similar formulation for the off-line design of minimum-time regulators [49]. This section will deal only with problem A, as defined in Section 3.2, in which each vehicle has a single, pre-assigned destination. Later sections will extend this to a more general form including waypoint assignment.

In this problem, vehicle $p$ is required to reach its destination $(x_{\mathrm{F_p}}, y_{\mathrm{F_p}})$ at some time-step before the maximum $N_T$. The binary variables $b_{\mathrm{tp}}$ are introduced, which have a value of 1 if the $p^{\text{th}}$ vehicle reaches its destination at the $t^{\text{th}}$ time-step, and 0 otherwise. The necessary MILP constraints are

$$\begin{array}{rrcrl}\forall p & \in & [1\ldots N_V] & \forall t & \in\ [1\ldots N_T] \\ & x_{\mathrm{tp}} - x_{\mathrm{F_p}} & \leq & & R(1-b_{\mathrm{tp}}) \\ \text{and} & x_{\mathrm{tp}} - x_{\mathrm{F_p}} & \geq & - & R(1-b_{\mathrm{tp}}) \\ \text{and} & y_{\mathrm{tp}} - y_{\mathrm{F_p}} & \leq & & R(1-b_{\mathrm{tp}}) \\ \text{and} & y_{\mathrm{tp}} - y_{\mathrm{F_p}} & \geq & - & R(1-b_{\mathrm{tp}}) \end{array} \tag{3.9}$$

$$\forall p \in [1\ldots N_V] \sum_{t=1}^{N_T} b_{\mathrm{tp}} = 1 \tag{3.10}$$

where $R$ is the same large, positive number used in Eqn. 3.7. It can be seen that if $b_{tp} = 1$, Eqn. 3.9 forces the aircraft position to equal the destination position at time-step $t$. However, if $b_{\mathrm{tp}} = 0$, then the constraints are inactive. Eqn. 3.10 enforces the logic that each vehicle must reach its target at one time point, but does not require that the vehicles finish at the same time. The minimum time solution for each aircraft is sought by minimizing the sum of the finishing times for each vehicle

$$\min_{\mathbf{s},\mathbf{f},\mathbf{b},\mathbf{c},\mathbf{d}} J = \sum_{t=1}^{T} \sum_{p=1}^{N} t b_{\mathrm{tp}} \tag{3.11}$$

where $\mathbf{b}$ are the binary variables for finishing times in Eqn. 3.9. The binary variables for obstacles $\mathbf{c}$ and for collision avoidance $\mathbf{d}$ are also decision variables.

The cost function in Eqn. 3.11 leads to an inefficient formulation. Since time has been discretized, there can be multiple solutions finishing at each time-step. In addition, the states and inputs for time-steps after the selected finishing time have no effect on the cost. These redundancies do not change the optimal cost, but results show that they have a dramatic impact on solution time. The strength of the branch and bound algorithm lies in its ability to "prune" regions of the search space and avoid working through all the integer combinations. However, the ambiguities described mean that multiple solutions achieve the global minimum cost, and the algorithm must find them all before terminating. This problem can be remedied by adding a small force penalty to the cost function

$$\min_{\mathbf{s},\mathbf{f},\mathbf{b},\mathbf{c},\mathbf{d}} J = \sum_{p=1}^{N} \left( \sum_{t=1}^{T} t b_{\mathrm{tp}} + \epsilon \sum_{t=0}^{T-1} \left( |f_{x_{tp}}| + |f_{y_{tp}}| \right) \right) \tag{3.12}$$

where $\epsilon$ is a positive number, small enough to ensure that the fuel penalty never exceeds the value of one time-step. The complete problem is then to minimize Eqn. 3.12 subject to the constraints in Eqs. 3.3–3.6, 3.9 and 3.10. With the added fuel penalty, there is a unique optimal solution, and the algorithm performs more efficiently. Due to the complexity of the CPLEX software, the exact cause of this improvement is unclear, but later results will show that it has a significant effect on the solution times.

### 3.3.5 Example: Single Aircraft avoiding Obstacles

The example in this section demonstrates that the linear constraints form an acceptable approximation to aircraft dynamics. It involves a single aircraft with a fixed destination. The dynamics were discretized with a time-step of four seconds. The aircraft has a maximum speed of 225m/s and a maximum turn rate of $5^{\text{o}}$/s. The initial position was $(-9.5, 0)$ with a velocity $(0.22, 0)$. The destination was $(9, -1)$ (all co-ordinates in kilometers). Three obstacles blocked the route as shown in Fig. 3-5, and the designed trajectory can be seen to successfully avoid these obstacles. The computation time was 5.22 seconds.

Avoidance is enforced at each of the discrete time-steps, but some of the lines joining adjacent time-steps in Fig. 3-5 would ‘cut corners’ through the zone. The exclusion zone models must therefore be slightly larger than the real zones, to ensure that such incursions do not encroach on the real zones.

Fig. 3-6(a) shows the speed of the aircraft along this trajectory. It remains close to, but never exceeds, its specified maximum. Fig. 3-6(b) shows the rate of turn. The trajectory is shown to consist of straight lines joined by turns at the maximum turn rate. This satisfies the necessary conditions derived in [40]. Note that some of the turns exceed the nominal turning rate by a small margin. At these times, the speed

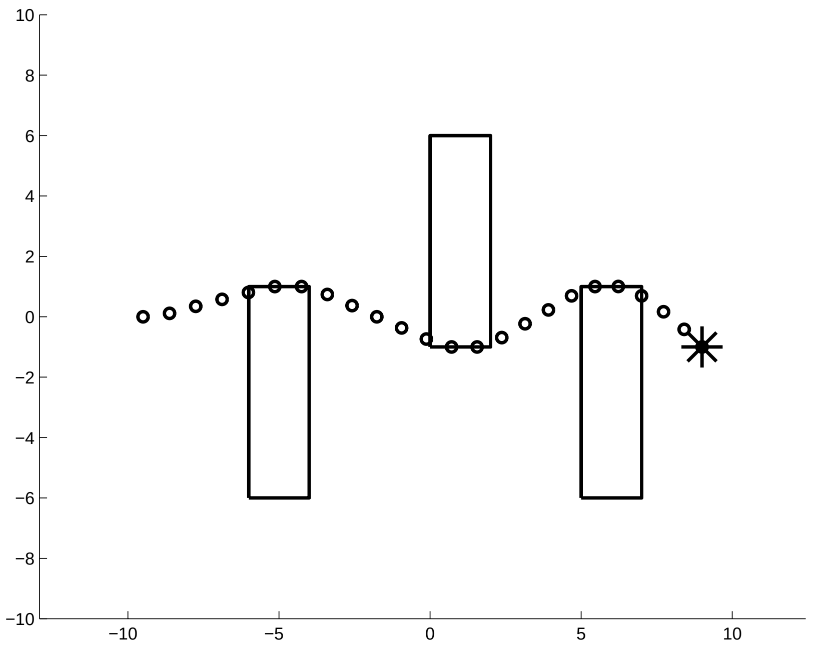


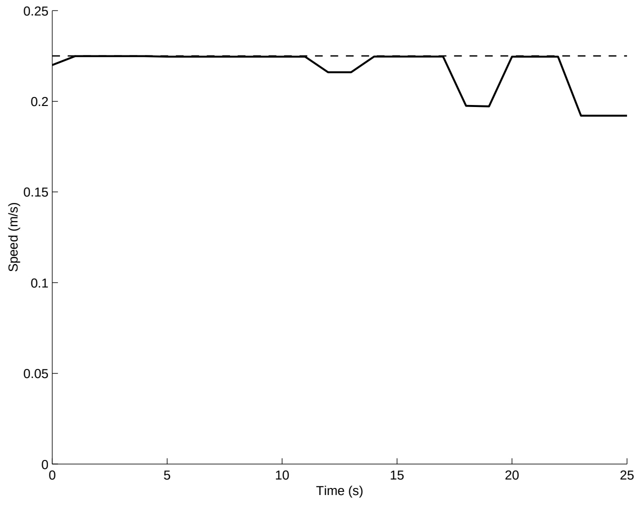
Figure 3-5: Designed Trajectory for Aircraft avoiding Obstacles. The star marks the destination point.

is below the maximum, hence the faster turn rate is possible.

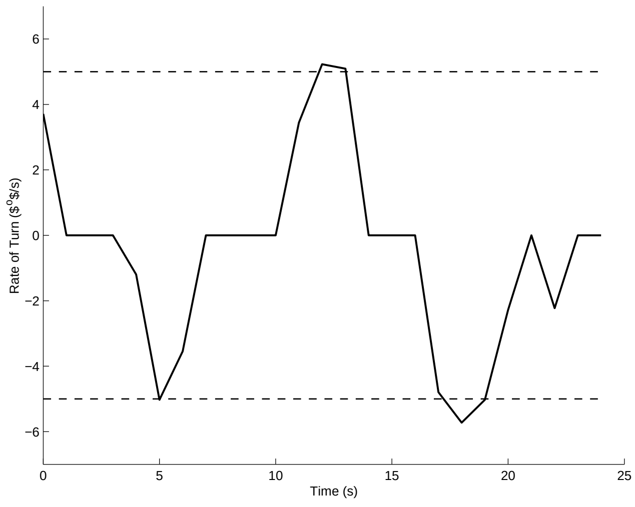
Further experiments have been performed to investigate the variation in computation time with particular problem instance. The single aircraft example described in this section was repeated with 100 different, randomly chosen layouts of obstacles. Each layout consisted of ten obstacles, each a square of side two units, placed in a square region extending seven units in each direction from the origin. Fig. 3-7 shows a histogram of the resulting computation times. Table 3.2 compares these results with those for the same computations done without the force penalty. It is shown that the inclusion of the penalty leads to faster solutions: in particular, the most difficult problems are solved much faster, leading to a substantial decrease in the maximum computation time.

Table 3.2: Comparison of Collision Avoidance Formulations applied to 100 Randomly-Generated Single Aircraft Problems

| Formulation | Computation Time (s) | |
|---|---|---|
| | Mean | Max. |
| Basic | 8.11 | 26.0 |
| Without force penalty | 14.7 | 78.4 |



(a) Speed



(b) Rate of turn

Figure 3-6: Speed and Turn Rate of Aircraft along Trajectory in Fig. 3-5

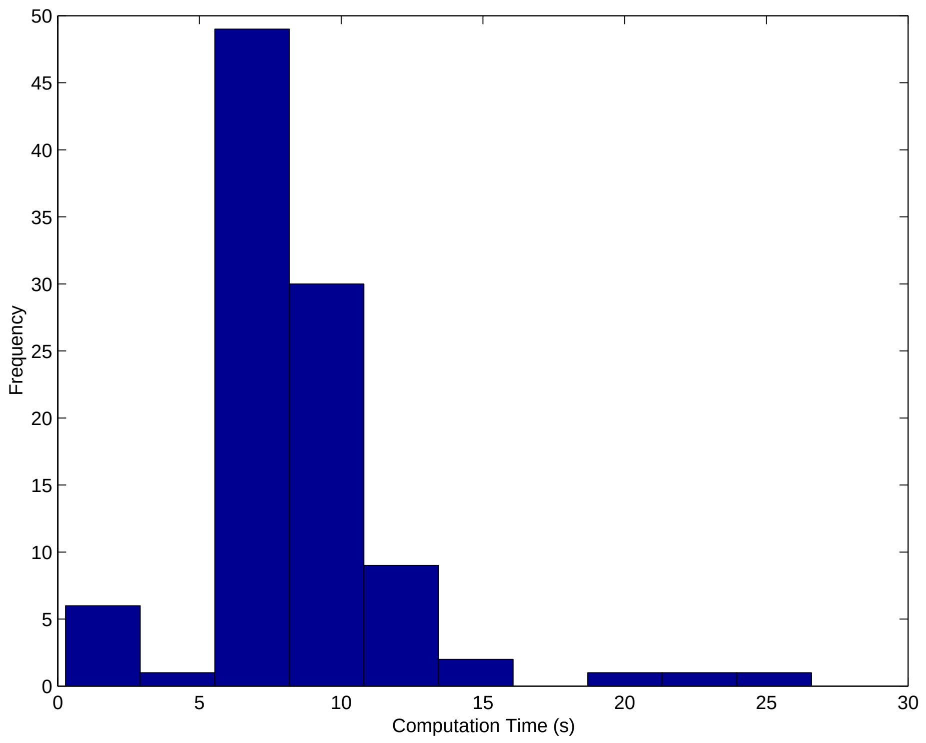


Figure 3-7: Histogram of Computation Times for 100 Randomly-Generated Single Aircraft Problems. The force penalty was included for these experiments.

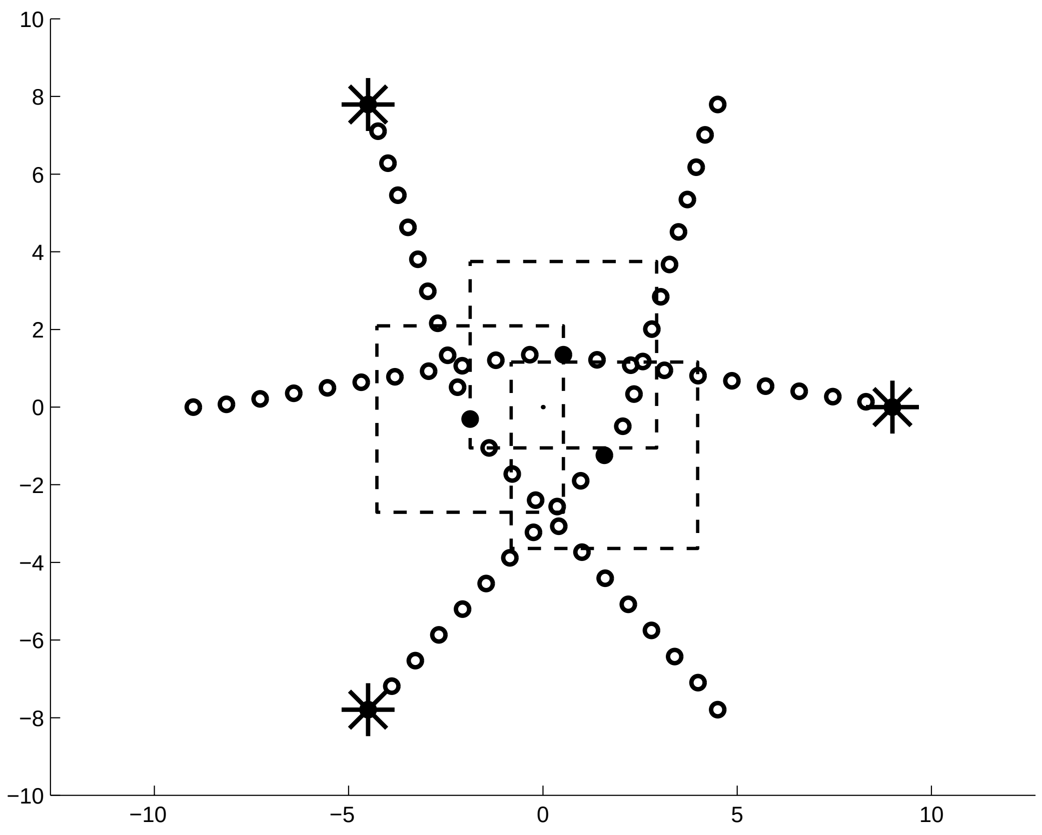


Figure 3-8: Designed Trajectories for Multiple Aircraft with Avoidance. The stars mark the target positions.

### 3.3.6 Example: Multiple Aircraft avoiding Collision

Having verified the model in the previous section, this example applies it to a collision avoidance problem involving multiple aircraft. Three aircraft, similar to those used in the previous example, are required to traverse different diameters of a circle, as shown in Fig. 3-8. Their destinations are marked by stars. The straight line paths to the destinations would clearly lead to a collision at the center. The designed trajectories form a 'roundabout' maneuver and successfully avoid collision with minimal deviation. The square exclusion region is 2.4km across. A similar result was shown in [40], found by an iterative process involving the necessary conditions for optimal avoidance. This example has repeated that result by direct optimization. The trajectories also demonstrate the cooperative nature of the solutions from this optimization method.

The heavy dots mark the positions of the aircraft at the 12th time-step. The

Table 3.3: Comparison of Computation Time Results for Different Numbers of Vehicles, solving 200 Randomly-Generated Problems for Each

| Number of vehicles | Computation Time (s) | | % solved in under 10 minutes |
|---|---|---|---|
| | Median | Max. | |
| 2 | 2.6 | 16.2 | 100 |
| 3 | 9.5 | 547.1 | 100 |
| 4 | 30.9 | 600.0 | 92 |
| 5 | 123.7 | 600.0 | 75 |
| 6 | 600.0 | 600.0 | 34 |

exclusion regions around these positions are shown by the dotted boxes. Observe that the vehicles are separated by exactly the safety distance in both directions, illustrating the efficiency of the formulation and the direct physical significance of the avoidance distance. This contrasts with penalty methods such as potential functions [50], in which the avoidance weighting is not as obviously related to the achieved distance and may need tuning.

Table 3.3 and Fig. 3-9 show the variation in computation times with the number of vehicles. For each number, 200 randomly-generated problems were solved. In each case, the vehicles started evenly spaced on the line between $(-5, -5)$ and $(-5, 5)$ (all distances in kilometers), each with velocity $(220, 0)$ m/s. The dynamics of each vehicle were the same as in Section 3.3.5. Their destinations were chosen randomly along the line between $(5, -8)$ and $(5, 8)$, leading to various combinations of interactions between vehicles during the maneuvers. For each problem, a computation time limit of ten minutes was applied, after which the search was terminated and the best available solution was returned.

In all cases, a feasible solution was found. For the cases computed in under ten minutes, this was known to be the global optimum. The majority of cases are solved quickly, but the median solution time grows exponentially with the number of vehicles. Meanwhile, the proportion of problems for which the global optimum can be found and verified in ten minutes decreases sharply with the number of vehicles.

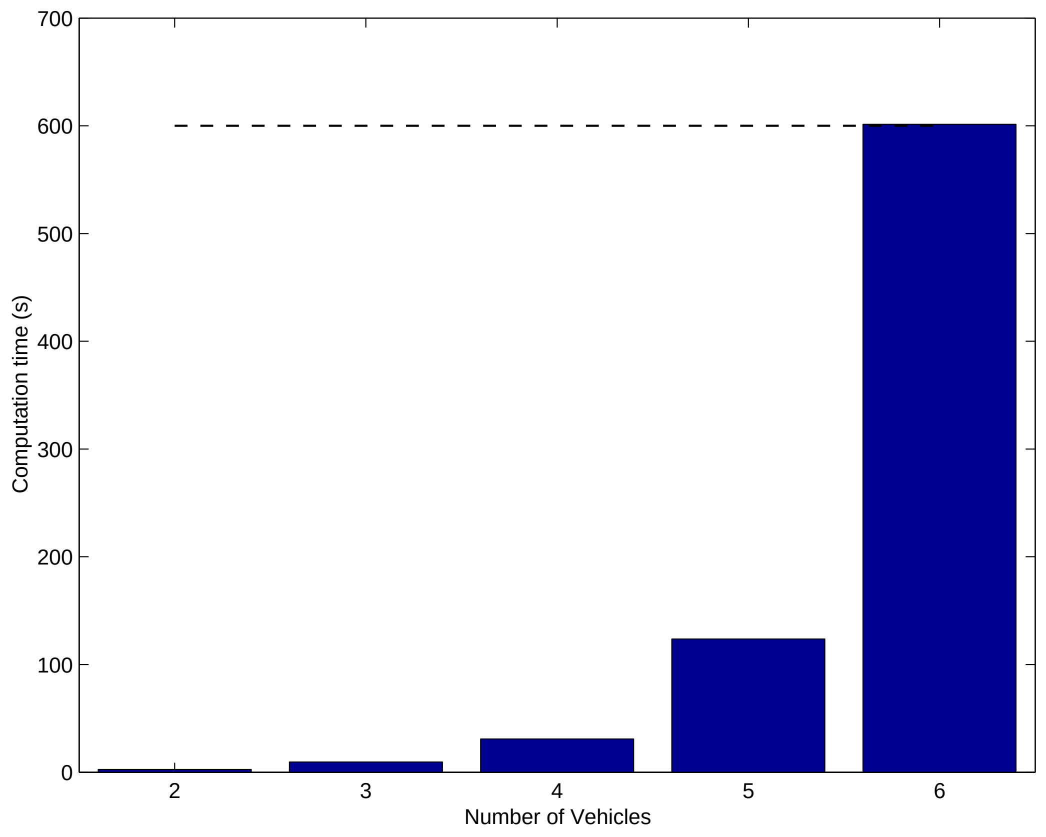


Figure 3-9: Variation of Computation Time with Number of Vehicles for Collision Avoidance Problem. The plot shows the median computation time, subject to a time limit of 600s.

### 3.3.7 Additional Avoidance Constraints

This section investigates potential modifications to the avoidance formulation. While the constraints in Section 3.3.3 have been shown to effectively enforce avoidance, there are further relationships that the solution must satisfy. These are found from the geometry of the problem, and are implicitly enforced by the existing constraints. For example, it is impossible to be both "left" and "right" of an obstacle at the same time. This section investigates if the inclusion of such additional constraints improves computation efficiency. The experiments described in Section 3.3.5 are repeated with the modified formulations to compare the solution times.

### Making the Problem *Completely Well-Posed*

Bemporad and Morari [47] discuss the notion of a *well-posed* problem in which binary variables are completely determined by the settings of the corresponding continuous variables. In the collision avoidance formulation in Eqn. 3.7, this is not the case. Consider an example in which the vehicle is both above and to the left of some obstacle at some time-step. The binary variables 2 and 3 are not uniquely defined: the logical constraint in the final line requires that one of them be zero, but it does not determine which. The addition of the following constraints removes this ambiguity making a *completely well-posed* problem [47].

$$\begin{array}{rrcl} \multicolumn{4}{c}{\forall t \in [1 \ldots N_T] \; \forall p \in [1 \ldots N_V] \; \forall j \in [1 \ldots N_Z]} \\ & x_{\mathrm{tp}} - Z_{\mathrm{j3}} & \leq & \epsilon + R(1 - d_{\mathrm{tpj1}}) \\ \text{and} & Z_{\mathrm{j1}} - x_{\mathrm{tp}} & \leq & \epsilon + R(1 - d_{\mathrm{tpj2}}) \\ \text{and} & y_{\mathrm{tp}} - Z_{\mathrm{j4}} & \leq & \epsilon + R(1 - d_{\mathrm{tpj3}}) \\ \text{and} & Z_{\mathrm{j2}} - y_{\mathrm{tp}} & \leq & \epsilon + R(1 - d_{\mathrm{tpj4}}) \end{array} \tag{3.13}$$

In the original constraints, if a binary variable was set to zero, that implied that the vehicle was on one side of a particular line. If the variable was one, the vehicle could have been either side of the line. These additional constraints are “complements” to the originals and enforce the opposite implication. If the binary variable is set to one, the vehicle is on the opposite side of the line. The margin $\epsilon$ is necessary to prevent ambiguity if the vehicle is exactly on the line.

### Additional Logic I

This subsection and the next describe two further observations leading to additional constraints. First, note that least two of the constraints must be in their ‘relaxed’ state, with the binary variables equal to one. For example, it is possible to be both to the right and above of an obstacle, but not possible to be to the right, above and to the left. This is captured in the following constraint

$$\begin{aligned} &\forall t \in [1 \dots N_T] \; \forall p \in [1 \dots N_V] \; \forall j \in [1 \dots N_Z] \\ &\text{and} \quad \sum_{k=1}^{4} d_{\mathrm{tpjk}} \;\geq\; 2 \end{aligned} \tag{3.14}$$

### Additional Logic II

A similar idea to that in the previous subsection leads to a different constraint: it is impossible to be simultaneously above and below a particular obstacle. Therefore, at least one of the third and fourth constraints in Eqn. 3.7 must be in its relaxed state, and similarly with the first and second. This is enforced by the following two constraints.

$$\begin{aligned} &\forall t \in [1 \dots N_T] \; \forall p \in [1 \dots N_V] \; \forall j \in [1 \dots N_Z] \\ &d_{\mathrm{tpj1}} \;+\; d_{\mathrm{tpj2}} \;\geq\; 1 \\ &d_{\mathrm{tpj3}} \;+\; d_{\mathrm{tpj4}} \;\geq\; 1 \end{aligned} \tag{3.15}$$

### Timing Results

Table 3.4 compares the timing results for the different formulations. The same 100 randomly generated problems used in Section 3.3.5 were repeated, with the modifications described. The results show that each modification led to an increase in computation time. These increases are small, suggesting that they might be due to the overhead of handling a larger number of constraints, which were effectively redundant. In conclusion, there is no computation advantage to be gained from making the problem completely well-posed or including additional logic in the formulation. The formulation in Section 3.3.3 efficiently captures the requirements for collision avoidance.

## 3.3.8 Using a Terminal Penalty

This section demonstrates the effect of relaxing the terminal constraints. In Eqn. 3.9, the last point of the trajectory was required to lie exactly on the destination point of the aircraft. Since the points are selected at discrete time intervals, it might be necessary to adjust the speed of the vehicle near the end of the trajectory to align

Table 3.4: Comparison of Collision Avoidance Formulations on 100 Randomly-Generated Single Aircraft Problems

| Formulation | Computation Time (s) | |
|---|---|---|
| | Mean | Max. |
| Original | 8.11 | 26.0 |
| Completely well-posed | 12.0 | 33.5 |
| Additional Logic I | 10.3 | 32.5 |
| Additional Logic II | 11.5 | 58.0 |

the discrete time point and the destination. Two such speed reductions can be seen in Fig. 3-6(a). In this section, the terminal constraint will be relaxed such that the final point has to be within the length flown in a time-step of the destination. The remaining distance is penalized in the cost function.

### Formulation

The terminal constraints in Eqn. 3.9 are replaced with the following constraints to find the distance of the chosen finishing point from the destination

$$\begin{array}{rrcrl} \forall p & \in & [1 \dots N_V] & \forall t & \in \; [1 \dots N_T] \\ & x_{\mathrm{tp}} - x_{\mathrm{F_p}} & \leq & x_{\mathrm{D_p}} & R(1-b_{\mathrm{tp}}) \\ \text{and} & x_{\mathrm{tp}} - x_{\mathrm{F_p}} & \geq & -\; x_{\mathrm{D_p}} & -\; R(1-b_{\mathrm{tp}}) \\ \text{and} & y_{\mathrm{tp}} - y_{\mathrm{F_p}} & \leq & y_{\mathrm{D_p}} & R(1-b_{\mathrm{tp}}) \\ \text{and} & y_{\mathrm{tp}} - y_{\mathrm{F_p}} & \geq & -\; y_{\mathrm{D_p}} & -\; R(1-b_{\mathrm{tp}}) \end{array} \tag{3.16}$$

where $x_{\mathrm{D_p}}$ is a decision variable representing the distance of the terminal point from the destination in the $X$-direction. The variable $y_{\mathrm{D_p}}$ represents the distance in the $Y$-direction. If $b_{\mathrm{tp}} = 1$, the terminal point for the $p^{\mathrm{th}}$ vehicle is at time-step $t$. The following additional constraint forces the terminal point to be within a box of size $2\ell \times 2\ell$ centered on the destination, where $\ell$ is a parameter representing the length flown in a time-step.

$$\begin{array}{l} \forall p \in [1 \dots N_V] \\ x_{\mathrm{D_p}} \;\leq\; \ell \\ y_{\mathrm{D_p}} \;\leq\; \ell \end{array} \tag{3.17}$$

This distance is penalized in the cost function by modifying Eqn. 3.12 as follows

$$\min_{\mathbf{s},\mathbf{f},\mathbf{b},\mathbf{c},\mathbf{d}} J = \sum_{p=1}^{N} \left( \alpha(x_{\mathrm{D_p}} + y_{\mathrm{D_p}}) + \sum_{t=1}^{T} t b_{\mathrm{tp}} + \epsilon \sum_{t=0}^{T-1} (|f_{x_{tp}}| + |f_{y_{tp}}|) \right) \tag{3.18}$$

where $\alpha$ is a weighting parameter. Since the time-step index $t$ is used as the time variable, the unit of time measurement in the cost is equal to the time-step for the discretized system. Therefore, the weighting should be such that a distance equivalent to one time-step of flight has a value of one unit in the cost function. This is approximately achieved by dividing by the maximum speed, but adjustment may be made to suit the accuracy of the one-norm measure of distance.

### Example

In this example, the terminal penalty formulation is applied to the problem in Section 3.3.5. Fig. 3-10 shows the designed trajectory, which is similar to the result from Fig. 3-5. Note that the terminal point is close to the destination point but not exactly on it.

Fig. 3-11 shows the speed and turn rate data, for comparison with Fig. 3-6. The speed now remains at the maximum value throughout. As a result, the turn rate limit is exactly obeyed. In conclusion, this formulation leads to better adherence to the desired dynamics restrictions of constant speed and limited turn rate.

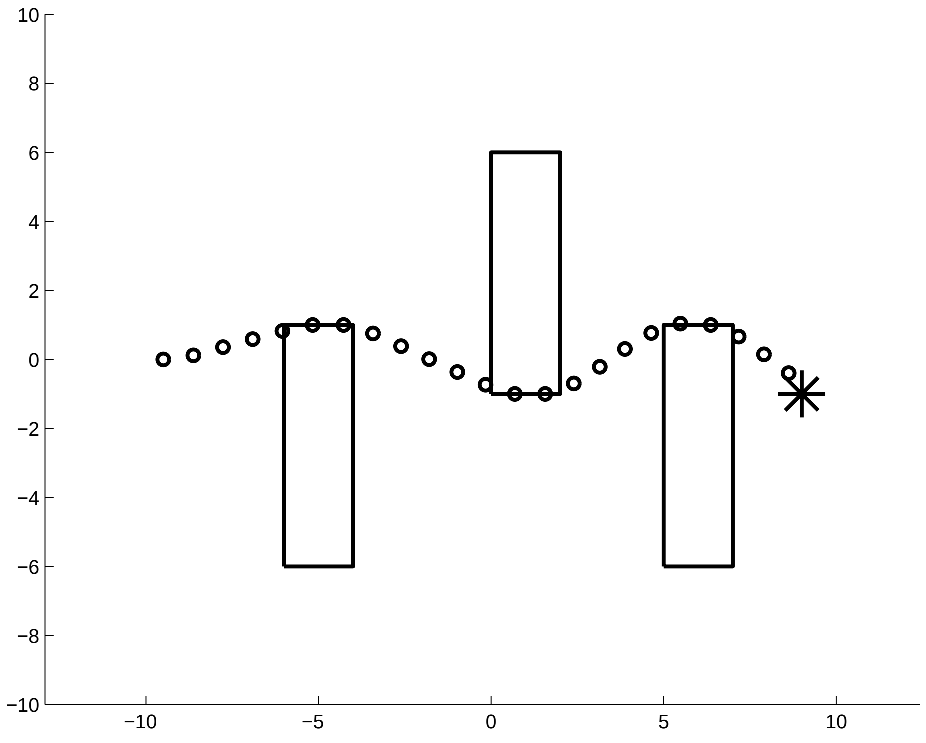
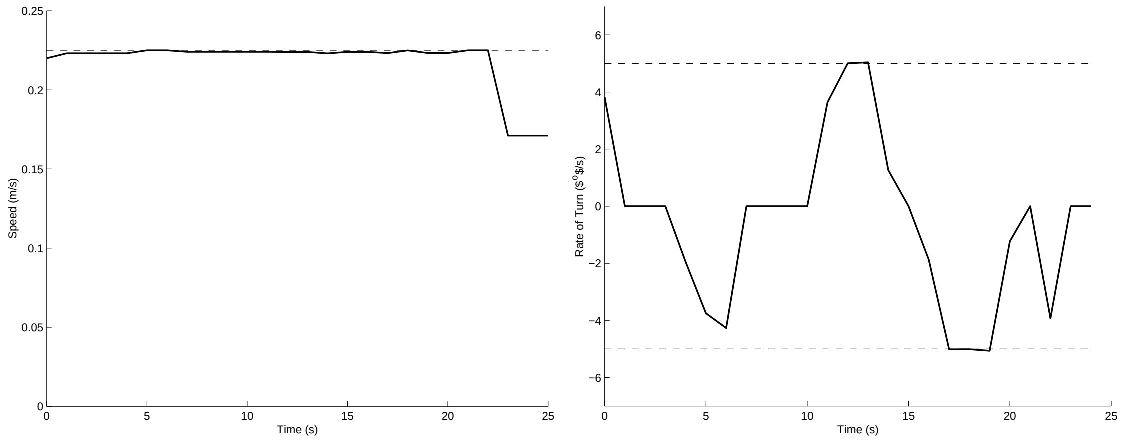


Figure 3-10: Designed Trajectory for Aircraft avoiding Obstacles using the Terminal Penalty. The star marks the destination point.



(a) Speed

(b) Rate of turn

Figure 3-11: Speed and Turn Rate of Aircraft along Trajectory in Fig. 3-10

## 3.4 Combined Assignment and Path-Planning

In this section, the constraints and cost function from Section 3.3 are extended to include multiple waypoint assignment. Instead of each vehicle having a single, fixed destination, a number of waypoints are specified in the problem. There can be more waypoints than vehicles, and the ordering of the visits need not be specified. The assignment of waypoints to vehicles, and the order in which the points are visited, are determined within the optimization. The assignment can be performed in two ways. The first, corresponding to problem B in Section 3.2, seeks the minimum overall mission time subject to capability and timing constraints. The second, corresponding to problem C, aims to maximize a 'score', including variable rewards for waypoints and penalties for entering certain zones.

### 3.4.1 Assignment for Minimum Mission Time

Capability constraints are included such that certain waypoints must be visited by certain vehicles. Also, time dependencies can be included to force specified waypoints to be visited before others.

Waypoint positions are specified in the $N_W \times 2$ matrix $\mathbf{W}$ where $(W_{\mathrm{i1}}, W_{\mathrm{i2}})$ is the position of the $i^{\mathrm{th}}$ waypoint and the total number of waypoints is $N_W$. The set of constraints to detect if a vehicle visits a waypoint can be written as

$$\forall p \in [1 \dots N_V]\ \forall t \in [1 \dots N_T]\ \forall i \in [1 \dots N_W]$$

$$\begin{array}{rrcl} & x_{\mathrm{tp}} - W_{\mathrm{i1}} & \leq & R(1 - b_{\mathrm{ipt}}) \\ \text{and} & x_{\mathrm{tp}} - W_{\mathrm{i1}} & \geq & -\ R(1 - b_{\mathrm{ipt}}) \\ \text{and} & y_{\mathrm{tp}} - W_{\mathrm{i2}} & \leq & R(1 - b_{\mathrm{ipt}}) \\ \text{and} & y_{\mathrm{tp}} - W_{\mathrm{i2}} & \geq & -\ R(1 - b_{\mathrm{ipt}}) \end{array} \tag{3.19}$$

where $b_{\mathrm{ipt}}$ is a binary decision variable and $R$ is the same large, positive number used in Eqn. 3.7. This equation is a generalized form of Eqn. 3.9, allowing all vehicles to visit all waypoints. It can be seen that $b_{\mathrm{ipt}} = 1$ implies that vehicle $p$ visits waypoint $i$ at time-step $t$. This binary variable can then be used in logical constraints for the assignment. Note that this formulation could easily be relaxed so that a vehicle 'visits' a waypoint if it passes within a specified distance of that point, using a formulation similar to that in Section 3.3.8.

Vehicle capabilities are specified in the matrix $\mathbf{K}$, in which $K_{pi} = 1$ if vehicle $p$ can visit the $i^{th}$ waypoint, and 0 otherwise. The matrix $\mathbf{K}$ has size $N_V \times N_W$. The following constraint enforces that each waypoint is visited exactly once by a vehicle with suitable capabilities.

$$\forall i \in [1 \dots N_W] \ \sum_{t=1}^{N_T} \sum_{p=1}^{N_V} K_{\mathrm{pi}} b_{\mathrm{ipt}} = 1 \tag{3.20}$$

Time dependencies, forcing one waypoint to be visited after another, separated by some interval, are included in the matrix $\boldsymbol{\Delta}$. Each row of the matrix represents a time dependency and it has a column for each waypoint. Thus if there are $N_D$ time dependencies, the matrix is $N_D \times N_W$. A dependency is encoded by $-1$ in the column corresponding to the first waypoint and $+1$ in the column for the second. The corresponding element in the vector $\mathbf{t}_D$ is the interval between the two visits. The constraints are then written as

$$\forall k \in [1 \dots N_D] \ \sum_{i=1}^{N_W} \Delta_{\mathrm{ki}} \sum_{t=1}^{N_T} \sum_{p=1}^{N_V} t \ b_{\mathrm{ipt}} \geq t_{D_{\mathrm{k}}} \tag{3.21}$$

in which the summations $\sum_{t=1}^{N_T} \sum_{p=1}^{N_V} t \ b_{\mathrm{ipt}}$ extract the time of visit for the $i^{\mathrm{th}}$ waypoint. Note that time is measured in units of time-steps, since the index $t$ is used as the measure of time at each step.

A modified cost function is required for the assignment formulation. The primary aim is to minimize the mission completion time. Small penalty weightings are included to help the numerical conditioning and accelerate the solution process. The first step is to extract the flight completion time $t_{\mathrm{p}}$ for the $p^{\mathrm{th}}$ vehicle, which is the time at which it visits its last waypoint

$$\forall p \in [1 \dots N_V] \; \forall i \in [1 \dots N_W] \; t_{\mathrm{p}} \geq \sum_{t=1}^{N_T} t \; b_{\mathrm{ipt}} \tag{3.22}$$

A similar set of constraints finds the overall mission completion time $\bar{t}$

$$\forall p \in [1 \dots N_V] \; \bar{t} \geq t_{\mathrm{p}} \tag{3.23}$$

The complete cost function is then

$$\min_{\mathbf{s},\mathbf{f},\mathbf{b},\mathbf{c},\mathbf{d}} J = \bar{t} + \epsilon_1 \sum_{p=1}^{N_V} [t_{\mathrm{p}} + \epsilon_2 \sum_{t=0}^{N_T-1} (|f_{x_{\mathrm{tp}}}| + |f_{y_{\mathrm{tp}}}|)] \tag{3.24}$$

where the decision variables are the forces **f** (which determine the state vectors **s**), and the binary variables **b**, **c** and **d**, for waypoint visit, collision avoidance and exclusion zone logic, respectively.

The weighting factors $\epsilon_1$ and $\epsilon_2$ are small positive numbers and are included to help the solution process. Their purpose is the same as that of the force penalty introduced in Section 3.3.4. The first weighting ensures that the minimum time path is chosen for all aircraft. If it were omitted, only the aircraft that finished last would be explicitly minimized, and those finishing earlier could select multiple paths without affecting the cost. The force weighting was discussed in Section 3.3.4. Together, the weightings force the problem to have a unique solution. Experience has shown that this greatly reduces the computation time.

### 3.4.2 Example: Assignment for Minimum Mission Time

This section shows a series of very simple examples demonstrating the effect of the assignment logic constraints, as developed in Section 3.4.1, including heterogenous vehicle capabilities and time dependencies. Obstacle avoidance is included where necessary, but collision avoidance constraints have been omitted for simplicity. The objective in these examples is to minimize the mission completion time.

Fig. 3-12(a) shows the designed trajectories for two vehicles visiting four waypoints. Both vehicles have the capability to visit all the waypoints, so every entry of the capability matrix is 1. There are no timing dependencies. As expected, each vehicle travels in a nearly straight path to the two nearest waypoints.

In Fig. 3-12(b), the scenario has been changed by removing the capability of vehicle 1 to perform the task at waypoint B, as it does in the solution of the first problem. Vehicle 2 is now the only vehicle with that capability, so it is required to visit point B. It would be feasible for vehicle 2 to follow the same trajectory as in the previous example, then visit point B at the end. However, by assigning vehicle 1 to point D, vehicle 2 can proceed straight from C to B, leading to an earlier mission completion.

Note that, in the plan shown in Fig. 3-12(b), vehicle 1 visits point A then point D. However, for the third problem, a timing constraint was added such that point D must be visited *before* point A. Clearly the previous trajectory is no longer feasible. In the optimal solution shown in Fig. 3-12(c), vehicle 2 goes almost directly to point D. Point C is on the way so it is visited in passing. Vehicle 1 moves slowly in order to arrive at point A just after vehicle 2 arrives at D. Finally, vehicle 2 is still required to visit point B due to the lack of capability of vehicle 1.

In the final variation on this problem, an obstacle is added to block the path from C to D taken by vehicle 2 in the previous design. Fig. 3-12(d) shows the new assignment and trajectories. It is still necessary that vehicle 2 visits point B, due to the lack of capability of vehicle 1, and that point D must be visited before point A. Therefore, vehicle 1 is sent directly to point D, while point A is visited by vehicle 2 on its way from C to B.

It can be seen that the design in each case satisfies the mission requirements, validating the formulation of the assignment constraints. The examples also demonstrate the complexity of the problem at hand: small changes in capability, timing constraints or obstacles can lead to completely different vehicle assignments, and a wide selection of permutations are used in even this simple example.

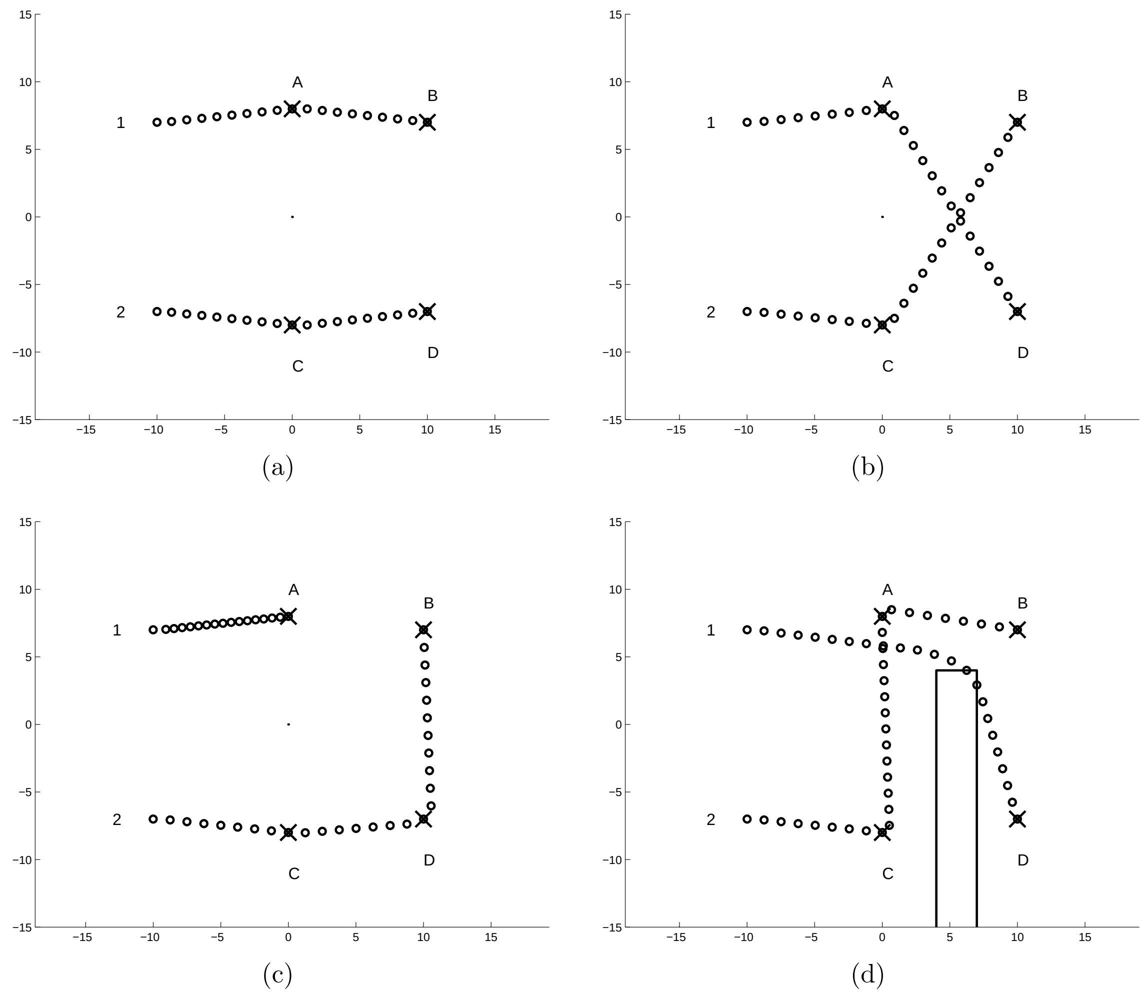


Figure 3-12: UAV Assignment Examples: (a) both vehicles have full capabilities; (b) as (a) but vehicle 1 cannot visit waypoint B; (c) as (b) but waypoint D must be visited before A; (d) obstacle added

### 3.4.3 Assignment for Maximum Score

In all the formulations shown so far, the primary objective has been to minimize flight time. In problems involving assignment of heterogenous UAVs to targets, there may be further considerations. In this section, the formulation will be modified to replace capabilities with scores for each vehicle-waypoint assignment. These scores may represent target values and probabilities of success for different vehicles. In addition, the exclusion zone formulation will be modified such that vehicles can enter exclusion zones, subject to penalties. These represent the risk to each vehicle of entering that region.

To include the scores for waypoint visits, a new variable $\bar{V}$ is introduced as the total score achieved. The following two constraints replace the capability-based constraint in Eqn. 3.20.

$$\bar{V} = \sum_{i=1}^{N_W} \sum_{t=1}^{N_T} \sum_{p=1}^{N_V} V_{\mathrm{pi}} b_{\mathrm{ipt}} \tag{3.25}$$

$$\forall i \in [1 \dots N_W] \ \sum_{t=1}^{N_T} \sum_{p=1}^{N_V} b_{\mathrm{ipt}} \leq 1 \tag{3.26}$$

where the parameter $V_{\mathrm{pi}}$ is the score achieved if the $p^{\mathrm{th}}$ vehicle visits the $i^{\mathrm{th}}$ waypoint. Eqn 3.26 prevents a waypoint being visited more than once, but allows waypoints to be neglected if their value is deemed insufficient.

The obstacle penalties are implemented by replacing the last line of Eqn. 3.8 with the following

$$\begin{aligned} &\forall t \in [1 \dots N_T] \ \forall p \in [1 \dots N_V] \ \forall j \in [1 \dots N_Z] \\ &\sum_{k=1}^{4} d_{\mathrm{tpjk}} \ \leq \ 3 \ + \ g_{\mathrm{pj}} \end{aligned} \tag{3.27}$$

If the additional binary variable $g_{\mathrm{pj}} = 0$, the constraints are the same as Eqn. 3.8 and the vehicle cannot enter the zone. If $g_{\mathrm{pj}} = 1$, all four of the conditions in Eqn. 3.8 can be in the relaxed state and the vehicle can enter the zone. These incursions are penalized to form the total penalty

$$\bar{P} = \sum_{j=1}^{N_Z} \sum_{p=1}^{N_V} P_{\mathrm{pj}} g_{\mathrm{pj}} \tag{3.28}$$

where $P_{\mathrm{pj}}$ is the penalty incurred if the $p^{\mathrm{th}}$ vehicle enters the $j^{\mathrm{th}}$ zone. Finally, the cost function from Eqn. 3.24 is modified to include the new variables and penalties.

$$\min_{\mathbf{s},\mathbf{f},\mathbf{b},\mathbf{c},\mathbf{d},\mathbf{g}} J = \bar{P} - \bar{V} + \sum_{p=1}^{N_V} [t_{\mathrm{p}} + \epsilon_2 \sum_{t=0}^{N_T - 1} (|f_{x_{\mathrm{tp}}}| + |f_{y_{\mathrm{tp}}}|)] \tag{3.29}$$

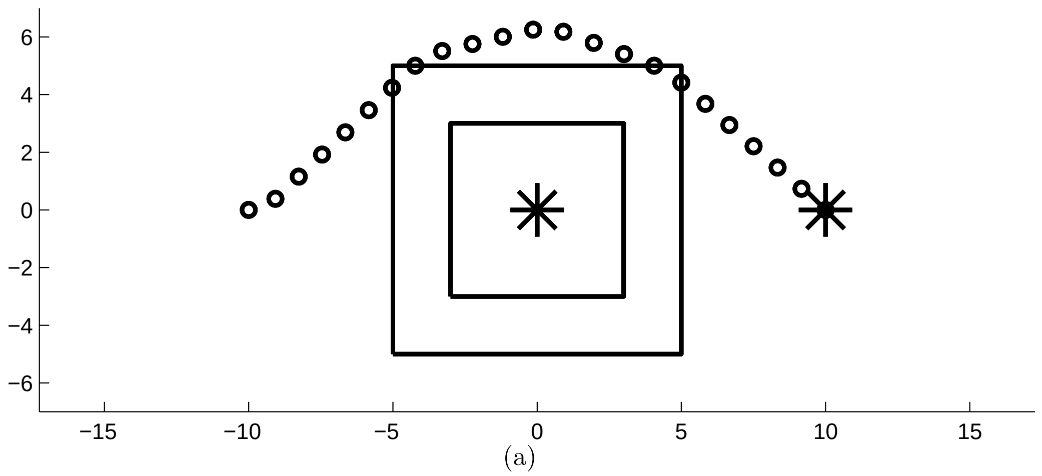
The individual finishing times are still penalized, now with unit weighting. This means that the penalties and scores are in units equivalent to the cost of one time-step of flight. The small weighting on force inputs is still necessary for the solver to work efficiently.

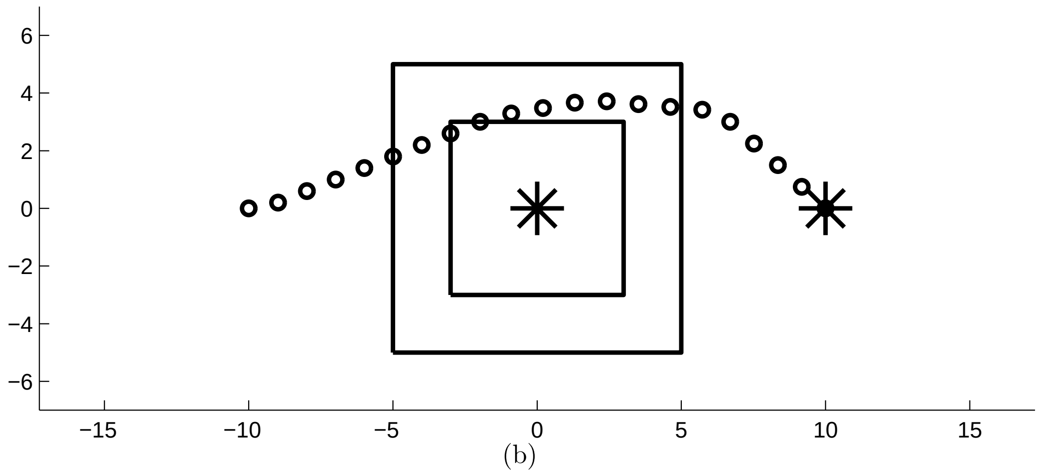
### 3.4.4 Example: Assignment for Maximum Score

The examples in this section demonstrate the use of scores for waypoint visits and penalties for exclusion zone incursions, implemented as described in Section 3.4.3. All three examples involve a single aircraft, with identical dynamics in each case. There are two exclusion zones and two waypoints. Only the exclusion zone penalties and waypoint scores are varied from case to case.

For the case in Fig. 3-13(a), the score for visiting each waypoint was 50. The penalty for the inner exclusion zone was 75 and for the outer zone 25. The vehicle remains outside both zones and visits only the waypoint on the right. In the case in Fig. 3-13(b), the penalty for the outer zone was reduced to 2. This was smaller than the additional cost to go around, so the vehicle enters the outer zone. It is still not worth entering the inner zone to visit the waypoint inside. Finally, in Fig. 3-13(c), the score for the waypoint inside the exclusion zones was increased to 100. This is now greater than the penalty for entering the exclusion zones, so the vehicle visits both waypoints and goes through both zones.

These examples have demonstrated that the formulation with scores and penalties performs as expected. The simplicity of these examples is intended to allow the solutions to be predicted by intuition and compared with the results by optimization. While more complicated examples can be solved, the arbitrary nature of the weightings makes the results difficult to predict.


(a)


(b)

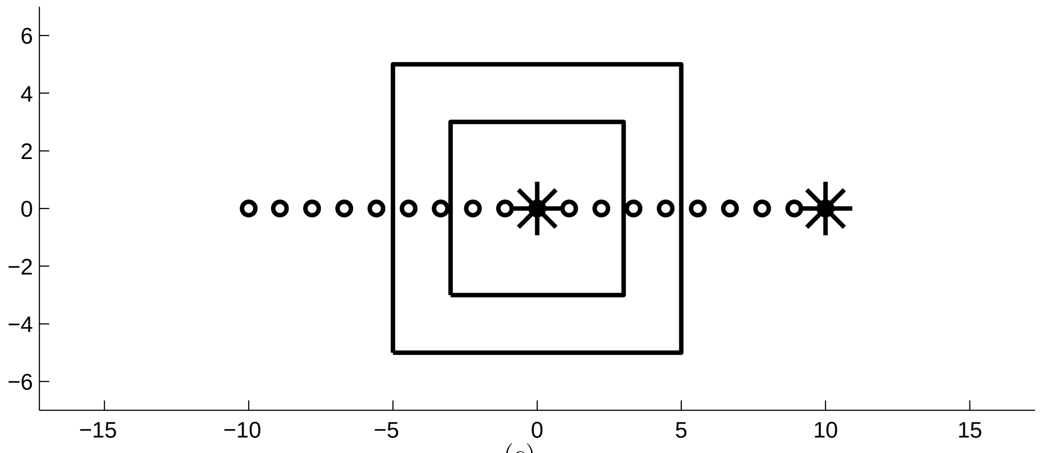

(c)

Figure 3-13: Examples of Assignment using Scores and Penalties

Table 3.5: Vehicle Capabilities in UAV Example

| Waypoint | Vehicle | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| A | X | X | |
| B | X | X | X |
| C | | X | X |
| D | X | X | |

Table 3.6: Vehicle-Waypoint Scores in UAV Example

| Waypoint | Vehicle | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| A | 50 | 50 | |
| B | 10 | 10 | 10 |
| C | | 50 | 50 |
| D | 50 | 50 | |

### 3.4.5 Example: UAV Problem

This section demonstrates the application of both assignment methods to a more complicated problem. It involves a fleet of three UAVs, required to visit four waypoints. It was first solved for the minimum mission time, with the vehicle capabilities shown in Table 3.5. There are no time dependencies. Fig. 3-14 shows the designed trajectories, found in just over eight minutes of computation. The total mission time for the designed solution is 23 time-steps (in this case, each step is four units: the scaling of this problem is arbitrary).

Fig. 3-15 shows the solution to a similar problem involving scores and penalties. Waypoint scores are shown in Table 3.6. The exclusion zone on the right has a penalty of 100, representing a mountain, while that on the left has a penalty of only 2. The optimization has found that it is not worth sending a vehicle to waypoint B, since its value is less than the cost for any vehicle to reach it, nor is it worth assigning vehicle 3 to any flight. Also, vehicle 2 flies through a penalty zone, since the penalty for entering is less than the flight time to go around.

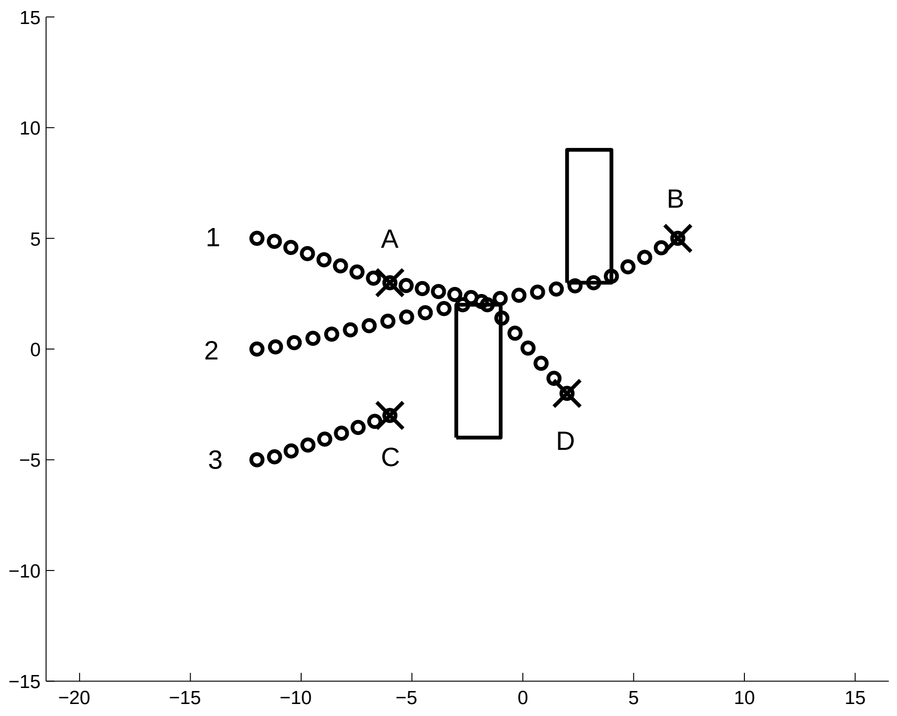


Figure 3-14: Solution of the UAV Example Problem for Minimum Mission Time

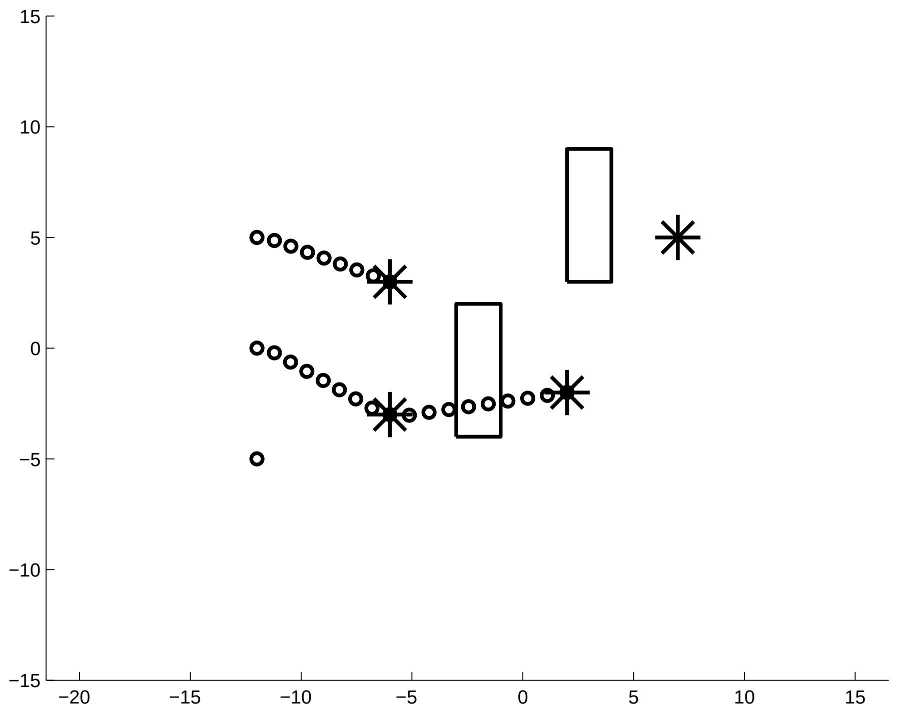


Figure 3-15: Solution of the UAV Example Problem for Maximum Score

## 3.5 Summary

This Chapter has shown that the MILP approach can be applied to problems involving aircraft. An approximate model allows aircraft dynamics to be included as linear constraints. A penalty is included to ensure that the optimization has a unique solution. This has been shown to improve solution time substantially. Furthermore, a single MILP can be used to solve the combined assignment and path-planning problems, capturing the inherent coupling between these. Assignment can be performed for minimum mission time or for an abstract 'score', and can include heterogenous vehicle capabilities and timing constraints.

The most general form of the minimum-time problem, that including waypoint assignment, is the minimization of the cost function in Eqn. 3.24 subject to the constraints in Eqs. 3.3–3.8, 3.19, 3.20, 3.22 and 3.23. Table 3.1 shows the list of parameters that completely specify the problem.

Note that the problem of a single, fixed destination for each vehicle is a special case of this formulation. In this case, the $p^{\text{th}}$ waypoint should be the destination of the $p^{\text{th}}$ vehicle and the capability matrix should be an identity $\mathbf{I}_{(N_W \times N_W)}$. Then each vehicle is required to visit its specified destination alone.

The problem with scores and penalties is solved with the cost function in Eqn. 3.29 subject to the constraints in Eqs. 3.3–3.7, 3.27, 3.28, 3.19, 3.25, and 3.26. Table 3.1 shows additional parameters for this problem.

The experimental results in this Chapter have shown that problems involving small numbers of vehicles and obstacles can be solved using a single MILP. Larger problems can lead to prohibitive computation times. Other research [52] suggests that effective approximations may exist for rapidly solving problems of this type, including many more vehicles and waypoints. The importance of this method is that it is guaranteed to find the globally optimal solution, since MILP problems are immune to issues of local minima. Therefore, it may be used as a "benchmark" against which approximate methods are evaluated for performance.

# Chapter 4

# Model Predictive Control

Chapter 2 showed the use of MILP to design spacecraft trajectories. The problems all involved solving single optimizations off-line to generate complete trajectories. This is an "open-loop" technique that cannot accommodate any uncertainty in the model or environment. In this chapter, the same optimizations are incorporated in a real-time scheme, solved on-line to compensate for the effect of uncertainty as the maneuver progresses.

The MILP optimizations are embedded in Model Predictive Control (MPC) [60]. This is a feedback scheme in which an optimization is solved online at each time-step. This optimization predicts the future behavior up to some horizon in time, using a model of the system dynamics, and designs the trajectory to minimize some cost functional over that period. It is also sometimes known as "receding horizon control" due to the way the prediction horizon 'recedes' ahead of the current time. Having found the optimal control series, only the first step is implemented, and the optimization is repeated for the new initial conditions.

This chapter will demonstrate MPC for spacecraft avoidance maneuvers using the MILP formulations from Chapter 2. Existing theoretical results are invoked to prove nominal stability of the resulting control scheme. Examples are used to show other aspects of MPC behavior, such as robustness and transients. This is not intended to be a through examination of these topics, rather a demonstration of the concepts. Related work by Bellingham [52] involves the application of the optimizations from

Chapter 3 to receding horizon control of aircraft.

**Nomenclature:** the nomenclature from Section 2.1 is also used in this Chapter.

## 4.1 Stable MPC for Spacecraft with Avoidance

This section shows the applicability of MPC to spacecraft maneuvers. Section 4.1.1 describes the MPC method, as it would be applied to spacecraft problems. Section 4.1.2 proves its stability using existing results. Section 4.1.3 demonstrates its application to the ISS rendezvous example.

### 4.1.1 MPC Overview

The MPC algorithm to be employed is as follows:

1. Solve a trajectory optimization using a formulation from Chapter 2, starting at current time $t$ and current state $\mathbf{x}$, and finishing at target equilibrium state $\mathbf{x}_{\mathrm{eq}}$ at time $t + H$. The prediction horizon is $H$ (the terminal point is always $H$ steps ahead, hence "receding horizon").

2. Implement the first step of the control sequence found by the optimization.

3. Repeat

The requirement that the final point be an equilibrium is necessary for stability. By implication, the horizon length must be at least long enough to complete the problem. This contrasts with other forms of MPC [52] in which only part of the trajectory is planned in detail, and a terminal cost is employed to represent the remainder of the problem. However, spacecraft applications are typically confined to proximity operations, and the requirement is therefore not overly restrictive in this case.

### 4.1.2 Stability of MPC

Bemporad and Morari [47] have shown that a general MILP optimization applied in an MPC framework is stable, subject to certain requirements being met. Let the system dynamics be denoted by

$$\mathbf{x}(t+1) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \tag{4.1}$$

where $t$ is the integer time-step index. Let $(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}})$ be an equilibrium pair such that $\mathbf{f}(\mathbf{x}_{\text{eq}}, \mathbf{u}_{\text{eq}}) = \mathbf{x}_{\text{eq}}$. This is the point about which the system is to be stabilized. At each time-step, an optimization is solved for the control and state sequences $\mathbf{u}(\cdot)$ and $\mathbf{x}(\cdot)$ as follows

$$V(t|t+H) = \min_{\mathbf{x},\mathbf{u}} \sum_{\tau=t}^{t+H} \ell(\mathbf{x}(\tau) - \mathbf{x}_{\text{eq}}, \mathbf{u}(\tau) - \mathbf{u}_{\text{eq}}) \tag{4.2}$$

where $V(t|t+H)$ is the minimum cost starting from time $t$ to complete the problem at time $t+H$, and $\ell(\cdot)$ is a positive definite value function of both its arguments. The optimization is solved subject to the following constraints

$$\mathbf{x}(\tau+1) = \mathbf{f}\left(\mathbf{x}(\tau), \mathbf{u}(\tau)\right) \ \forall \tau \in (t \ldots t+H-1) \tag{4.3}$$

$$[\mathbf{x}, \mathbf{u}](\tau) \in \mathcal{A} \quad \forall \tau \in (t \ldots t+H) \tag{4.4}$$

$$\mathbf{x}(t+H) = \mathbf{x}_{\text{eq}} \tag{4.5}$$

$$\mathbf{u}(t+H) = \mathbf{u}_{\text{eq}} \tag{4.6}$$

where $\mathcal{A}$ is the admissible set of states and controls. In our case, this set constraint includes the impingement and avoidance constraints. The first constraint ensures the dynamics model is satisfied and the final two constraints fix the horizon point at the target equilibrium.

Having implemented the first step of the control found from solving Eqn. 4.2, and assuming that the system obeys its dynamics model, it would then be feasible to complete the problem by implementing the rest of the control sequence, finishing at the end of the previous horizon. The minimum cost to complete this maneuver would be

$$V(t+1|t+H) = V(t|t+H) - \ell(\mathbf{x}(t) - \mathbf{x}_{\text{eq}}, \mathbf{u}(t) - \mathbf{u}_{\text{eq}}) \tag{4.7}$$

since the cost would be the same as found at the previous step, less that due to the step already performed. The next optimization starts from time $t+1$ and has an additional step, ending at time $t+1+H$. A feasible solution to this optimization would be to reach the target equilibrium at $t+H$, using the rest of the control sequence from the previous step and incurring the cost from Eqn. 4.7. For the last step, up to $t+1+H$, the system remains at the target, incurring no cost since $\ell(\cdot)$ is positive definite. This solution gives an upper bound for the second optimization cost

$$V(t+1|t+H+1) \leq V(t+1|t+H) \tag{4.8}$$

Combining this result with Eqn. 4.7 shows that the successive optimization cost results must be monotonically decreasing.

$$V(t+1|t+H+1) - V(t|t+H) \leq -\ell(\mathbf{x}(t) - \mathbf{x}_{\text{eq}}, \mathbf{u}(t) - \mathbf{u}_{\text{eq}}) \tag{4.9}$$

Since $V$ is a positive function and monotonically decreasing over time, it must eventually converge to a constant value, hence

$$\ell(\mathbf{x}(t) - \mathbf{x}_{\text{eq}}, \mathbf{u}(t) - \mathbf{u}_{\text{eq}}) \overset{t\to\infty}{\longrightarrow} 0 \tag{4.10}$$

and since $\ell(\cdot)$ is required to be positive definite, this implies the asymptotic stability result

$$\begin{aligned} \mathbf{x}(t) &\overset{t\to\infty}{\longrightarrow} \mathbf{x}_{\text{eq}} \\ \mathbf{u}(t) &\overset{t\to\infty}{\longrightarrow} \mathbf{u}_{\text{eq}} \end{aligned} \tag{4.11}$$

Two major differences exist between the optimizations solved in the Chapter 2 and the form presented in Eqs. 4.2–4.6. The first is the constraint in Eqs. 4.5 and 4.6 that the target point be an equilibrium for the system. In the examples that follow, the target points have been chosen to be equilibria, but this may be restrictive for future problems, and will be the subject of further research. This is just one form of the proof of MPC stability, and others [61] involve more general forms of terminal constraints.

The second difference is that the optimizations performed so far have only penalized fuel. In the context of the proof above, this implies that the value function $\ell(\cdot)$ is only positive definite for the control, and that the result in Eqn. 4.10 implies only that $\mathbf{u}(t) \to \mathbf{0}$. Thus, the stability guarantee is violated *only* if there exists some invariant trajectory, other than the target itself, in which $\mathbf{u}(t) = \mathbf{0}$ everywhere. Once the system reached such a trajectory, $\ell(\cdot)$ would converge to zero and no further control would ever be implemented. Invariant trajectories in the relative spacecraft problem are closed-form ellipses and in-track separations [15]. Starting from an in-track separation, the solution will always involve firing on the first time-step. From a closed-form ellipse, it will involve firing at some point on the ellipse to move off it. As the spacecraft moves around the ellipse, it will eventually reach a point from which the optimal plan involves firing on the next time-step, and therefore a control will be implemented to leave the ellipse. Minimum-fuel optimization will therefore lead to stable operation for spacecraft dynamics. However, for a more general solution to this issue, a small state penalty could be added. The use of such penalties is also desirable for maneuver timing, and is discussed further in Section 4.2.

In conclusion, this section has shown that using the optimizations from the previous chapter in an MPC framework will give asymptotic stability, provided the terminal point is an equilibrium. This result has not explicitly addressed the issue of uncertainty. The stability rests on the satisfaction of Eqn. 4.9, showing that the optimization cost is monotonically decreasing. This is guaranteed for the nominal case, and we expect the system to remain stable when subjected to bounded disturbances, sufficiently small such that the decrescent property of the cost is maintained.

### 4.1.3 Demonstration for ISS Rendezvous

Fig. 4-1 shows the ISS rendezvous maneuver performed by MPC with a horizon of 30 time-steps. The simulation model included an in-track drag force of $10^{-5}$ N and

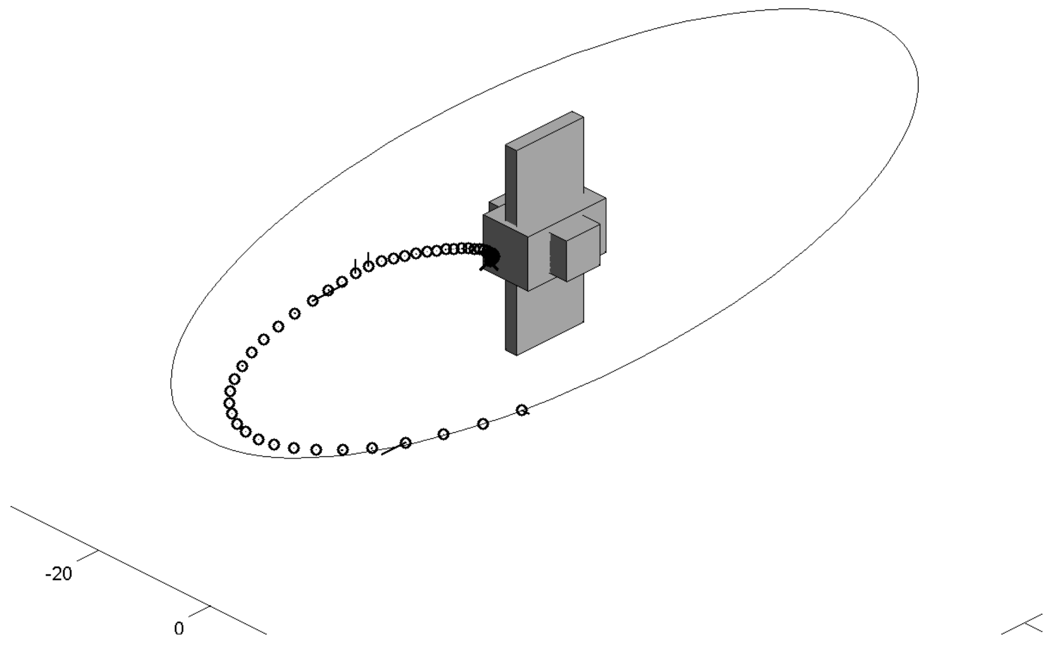


Figure 4-1: ISS Rendezvous using MPC

Gaussian white noise on each of the state measurements, with covariance 0.02 m on each position state and $5 \times 10^{-5}$ m/s on each velocity component. The prediction model did not include either of these effects. The same maneuver was considered in Section 2.5.5. Fig. 2-14 shows the designed trajectory from an open-loop planner, without the disturbances.

Fig. 4-2(a) shows a close-up of the final position when open-loop control is used in the presence of disturbances. Having solved a single optimization at the starting point, the entire sequence of control inputs is implemented. There is no compensation for the unmodeled effects, and the spacecraft does not reach the target. In contrast, Fig. 4-2(b) shows the final position under MPC. As expected, the inclusion of feedback has allowed the controller to compensate for the disturbances and the spacecraft reaches its target. This result demonstrates the key benefit of MPC: its ability to perform in the presence of uncertainty, making it suitable for real-world applications.

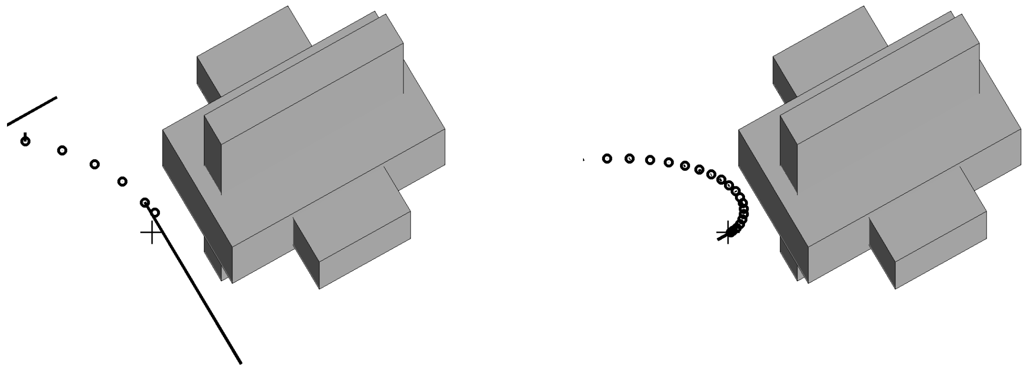

(a) Open loop (b) MPC

Figure 4-2: Comparison of final positions under open loop planning and MPC with small disturbance

Fig. 4-3 shows the computation times for the optimizations at each step of the MPC simulation. All optimizations were completed in under ten seconds, much less than the 90 second time-step of the discrete system. The sudden increase in solution times for optimizations after the 30th time-step is due to a change in the formulation used. Initially, when the spacecraft is far from the station, the plan is predicted to have the "bang-off-bang" firing profile, and the iterative method from Section 2.5.2 is used. When the spacecraft is very close to the station, this prediction is no longer valid. Therefore, the algorithm switches to the fully constrained plume impingement optimization when a certain distance threshold is crossed. This is a heuristic method and the threshold is currently chosen arbitrarily. In experiments using the interative scheme throughout, the computations at the end of the maneuver were very slow. Similarly, solving the fully-constrained problem from the beginning gave slow solution times at the start. A further development for computation is shown in Section 4.4.

Fig. 4-4 shows the optimization costs returned at each time-step. As predicted analytically in Eqn. 4.9, it is monotonically decreasing. Fig. 4-5 shows the position time histories throughout the maneuver. Although the planning horizon was 30 time-steps, it took approximately 50 time-steps to complete the maneuver. This shows that

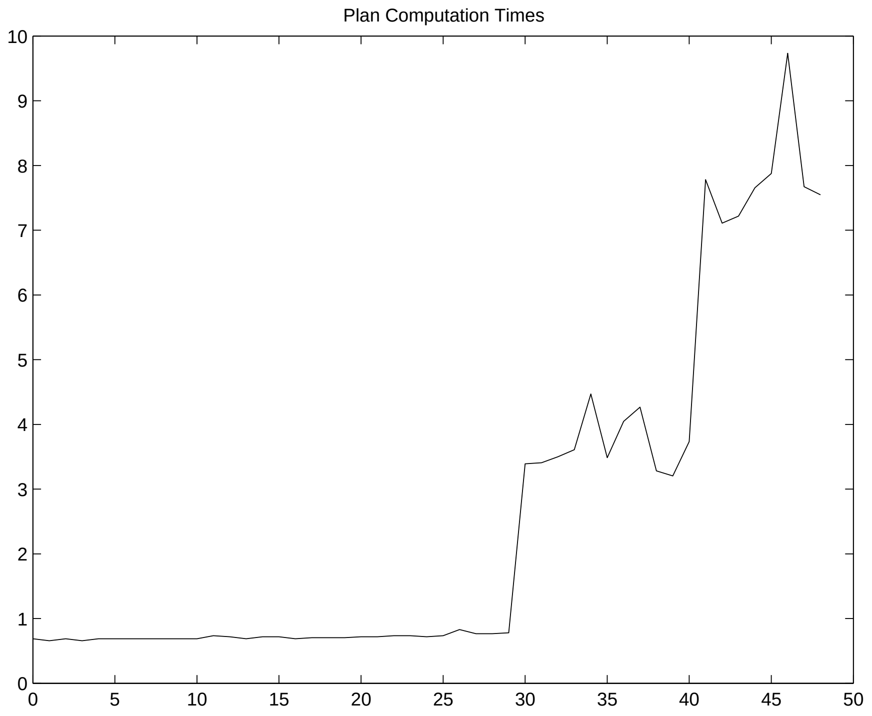


Figure 4-3: Optimization Solution Times for MPC

the optimization found a new solution, part way through executing the maneuver, requiring less fuel but taking longer to reach the target. The fuel use for the MPC maneuver was 22% lower than for the equivalent fixed horizon maneuver taking 30 time-steps. Section 4.2 discusses the issue of maneuver duration in greater detail. Also, comparison of Figures 4-2(a) and 4-2(b) shows that the new approach strategy passes much closer to the station than the original plan, making its final approach to fire away from the wall instead of along it. The smaller margin for error raises questions of robustness, discussed in Section 4.3.

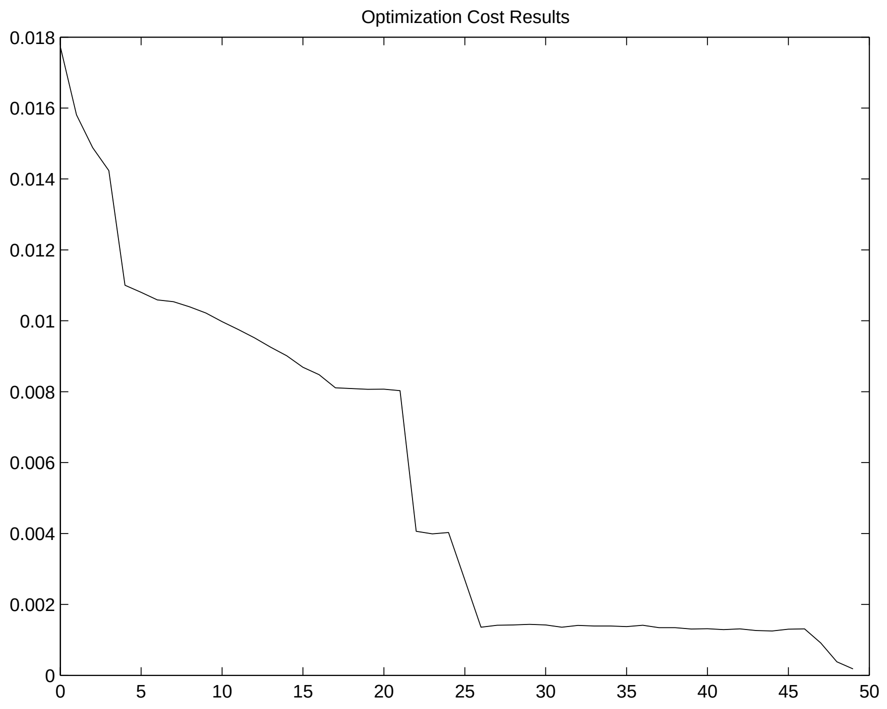


Figure 4-4: Optimization Cost Results under MPC

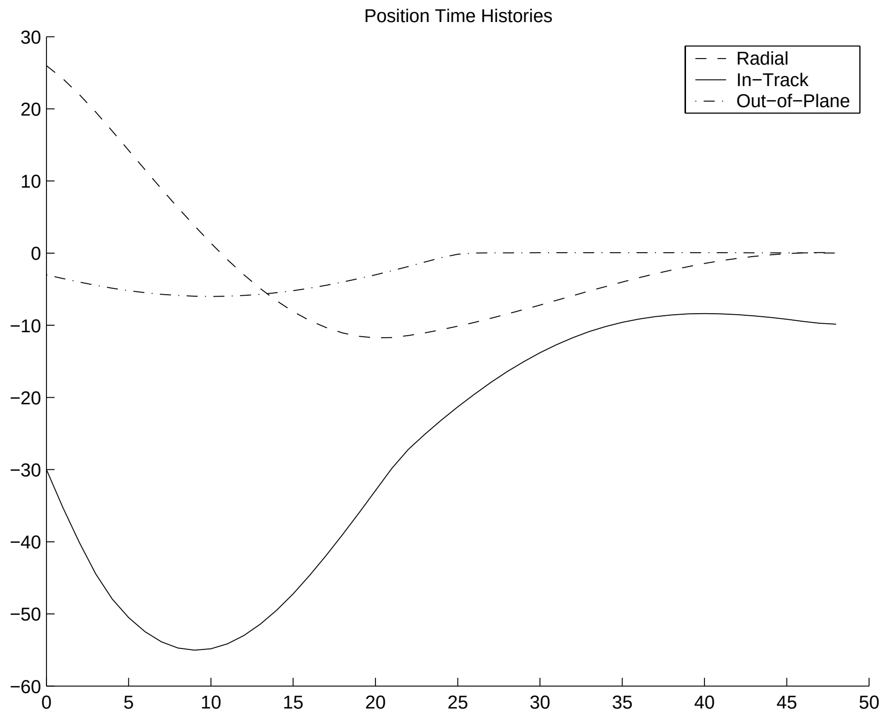


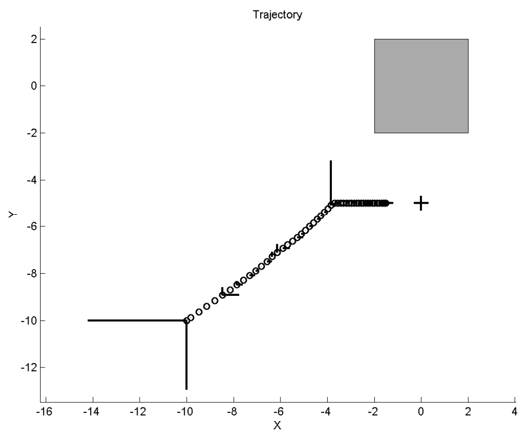
Figure 4-5: Position Time Histories under MPC

## 4.2 Maneuver Timing in MPC

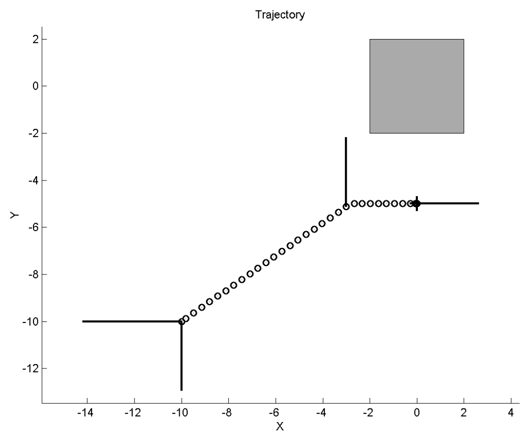
In Section 4.1.2, it was argued that penalizing state errors in the cost function was not essential for asymptotic stability. However, this section shows an example of the effect that such weightings can have on the transient behavior of the system under MPC. This is of primary interest in maneuvering problems. The example in this section involves a 2-D rendezvous maneuver as seen in Section 2.4.3. A vehicle moves in 2-D free-space and is required to stop at the target marked by $\times$ without impinging upon the obstacle. MPC with a horizon of 30 time-steps was used.

Fig. 4-6 compares the trajectories and position time-histories for two different levels of state error weighting. For the case in Fig. 4-6(a), the state error was not penalized, and the vehicle does not reach the target in 50 seconds. Fig. 4-6(c) shows that the vehicle position tends exponentially to the target but never actually gets there. Since the system dynamics are those of a point mass in free space, momentum is conserved, and the fuel use to decelerate while approaching the target is independent of the time taken. This is the equivalent of a singular arc [37] in classical optimal control, since the optimization does not uniquely define the control. In this case, the optimization algorithm returns the slowest possible solution, and the vehicle decelerates at every step, always planning to reach the target at the very end of the horizon.

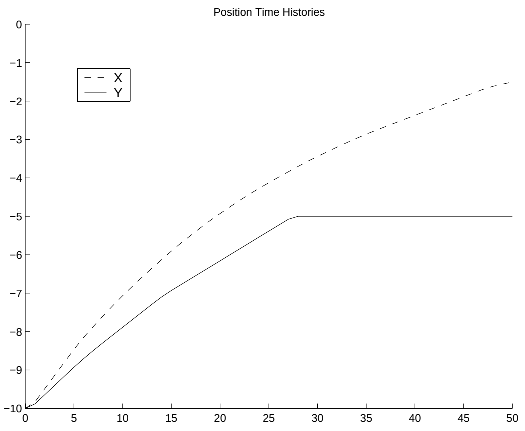
In Fig. 4-6(b), the one-norm of the state error at each step was penalized with a weighting of magnitude $10^{-4}$. The vehicle now reaches the target. The singularity is removed, as the optimization now favors an early completion time in the absence of a decision on fuel use alone. Fig. 4-6(d) shows that the maneuver is completed in 30 steps, equal to the horizon length. However, the low weighting of the state error means that the final fuel cost is close to the minimum fuel use for a maneuver duration equal to the horizon length. In conclusion, the addition of a small state error penalty in the cost function helps convergence, particularly for momentum-conserving dynamics, with little impact on performance.



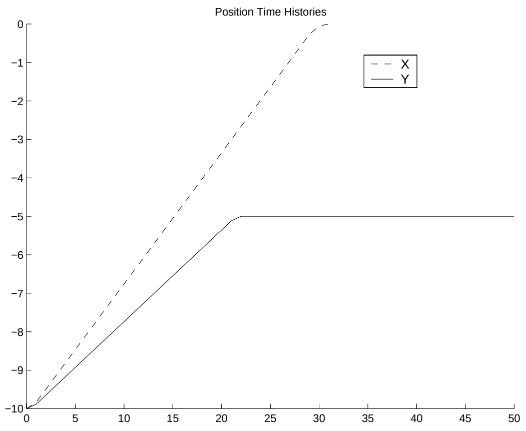
(a) Trajectory without state penalty



(b) Trajectory with state penalty



(c) Position history without state penalty



(d) Position history with state penalty

Figure 4-6: State histories for 2-D Vehicles with Different State Error Weightings

## 4.3 Including Soft Constraints for Robustness

Section 4.1.3 showed by experiment that small disturbances can be accommodated by MPC. However, the proof of stability in Section 4.1.2 assumed that the system behaved exactly as predicted by the model and did not explicitly address robustness. Furthermore, the disturbances used in the previous section were artificially small. Recent research involving differential GPS for spacecraft [6] has achieved accuracy on the order of centimeters for position and millimeters per second for velocity. When these levels of uncertainty are included in the simulation from the Section 4.1.3, the spacecraft can reach states from which the optimization cannot be solved, no control can be found and, at worst, the spacecraft hits the station. Theoretical results exist concerning MPC robustness [62], and future research will investigate their application to avoidance problems. This section demonstrates a heuristic technique for improving robustness.

In Fig. 4-2(b), the spacecraft moves very close to the station during its final approach. Instinctively, this is bad from a robustness perspective. It occurs in part due to the longer time available for the MPC maneuver: as the spacecraft approaches the station, it is able to find plans using all the available time and maneuvering space. We therefore include a *soft constraint* to encode the requirement that the spacecraft should not enter a certain region around the station unless absolutely essential. Soft constraints are often used to ensure feasibility of the optimization [60]. A similar formulation has been discussed as a way of embedding heuristics, such that a certain value is placed on obeying a constraint if possible [47]. It was also used for the inclusion of penalty zones in Section 3.4.3. In this case, additional 'softened' obstacle constraints are added.

$$\begin{array}{rrcll} \forall p, \forall l, \forall i \in [1 \ldots T-1]: & x_{\mathrm{ipn}} & \geq & \hat{U}_{\mathrm{ln}} - M\hat{a}_{\mathrm{ipln}} & \forall n \\ \text{and} & x_{\mathrm{ipn}} & \leq & \hat{L}_{\mathrm{ln}} + M\hat{a}_{\mathrm{ipl(n+N)}} & \forall n \\ \text{and} & \sum_{k=1}^{2N} \hat{a}_{\mathrm{iplk}} & \leq & 2N - 1 + s & \end{array} \tag{4.12}$$

These are modified versions of the original obstacle constraints in Eqn. 2.12. An enlarged set of obstacles are specified in $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$. A new set of binary variables $\hat{\mathbf{a}}$ are used. The crucial modification is the addition of the binary variable $s$ to the last line. If $s = 0$, these constraints are the same as those in Eqn. 2.12 and the spacecraft remains outside the enlarged obstacles. If $s = 1$, the logical constraint in the last line allows all of the other constraints to be relaxed, therefore allowing the vehicle to enter the obstacle. Finally, the cost function from Eqn 2.7 is augmented by a penalty for entering these regions.

$$J = \sum_{i=0}^{T-1} \sum_{p=1}^{V} \sum_{n=1}^{N} |u_{\mathrm{ipn}}| + Cs \tag{4.13}$$

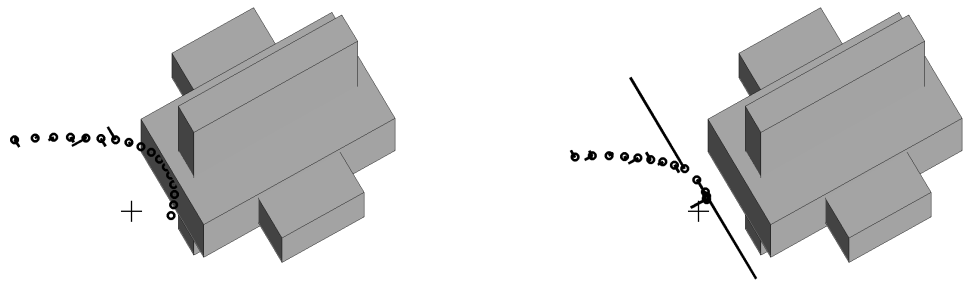

(a) Without soft constraints (b) With soft constraints

Figure 4-7: Final Stages of Maneuvers Subject to Large Disturbance, With and Without Soft Constraints

where C is a positive number, larger than the maximum fuel cost that will ever be incurred. Therefore, any feasible solution remaining outside the soft obstacles will be favorable to a solution entering them, and the trajectory design remains outside the soft obstacles unless absolutely necessary for a feasible solution.

Fig. 4-7 shows the effect of including soft constraints on a problem subject to large disturbances. The velocity noise was increased to 1 mm/s and position noise to 5 cm. Fig. 4-7(a) shows the trajectory followed under MPC without soft constraints. The optimization becomes infeasible and the spacecraft hits the station. Fig. 4-7(b) shows the trajectory followed using MPC with soft constraints 2 m beyond the station in all directions. The spacecraft remains much further from the station and the optimization is feasible throughout.

## 4.4 LP Presolve for MPC

This section describes the inclusion of a "presolve," in which a linear program (LP) is solved before each MILP. In the example of Section 4.1.3, all the problems were solved in under ten seconds, shown in Fig. 4-3. The time-step for the discretized system was 90 seconds, so the solution time seems satisfactory for real-time operation. However, the upper bound on the computation time of the $\mathcal{NP}$-hard MILP is much higher, so there is no absolute guarantee that the problem will be solved in the available time. The LP presolve guarantees the availability of a solution in a short, bounded computation time. The LP is solved by the simplex algorithm, which has been proven to solve in polynomial time [67]. Should the MILP fail to find a satisfactory solution before the control is required, the LP solution can be used in its place, without loss of stability.

### 4.4.1 Presolve Formulation

The LP presolve is identical to the full MILP except for one key aspect. The MILP solves for the binary variables as decision variables. The presolve takes the binary variables as fixed parameters, chosen outside the optimization, and solves for the continuous variables only. The binary settings satisfy two important constraints. The first is that the continuation of the plan from the previous solution must be a feasible solution to the LP presolve. This satisfies the requirement of the stability proof in Section 4.1.2: the new cost should be no greater than the cost to complete the problem using the previous plan. The use of the LP solution for the control then leads to stable operation. The second constraint is that the binary settings satisfy the logical constraints to prevent collision and plume impingement. This ensures that the solution of the presolve will not violate the avoidance constraints.

The binary variables are not uniquely determined by the constraints established above. Consider the scenario in Fig. 4-8, in which binary variables are to be determined to prevent impingement by a plume, marked by the dashed rectangle, at a particular time-step. At the corresponding step in the previous solution, the vehicle is at position $(x_\mathrm{i}, y_\mathrm{i})$ and is not firing in the direction shown. Therefore, three possible constraints are considered, each enforced by setting a particular binary variable to zero in the full plume constraints (Eqn. 2.17):

1. constrain the vehicle to remain to the right of line A–A
2. constrain the vehicle to remain below line B–B
3. prevent the vehicle from firing in the direction shown

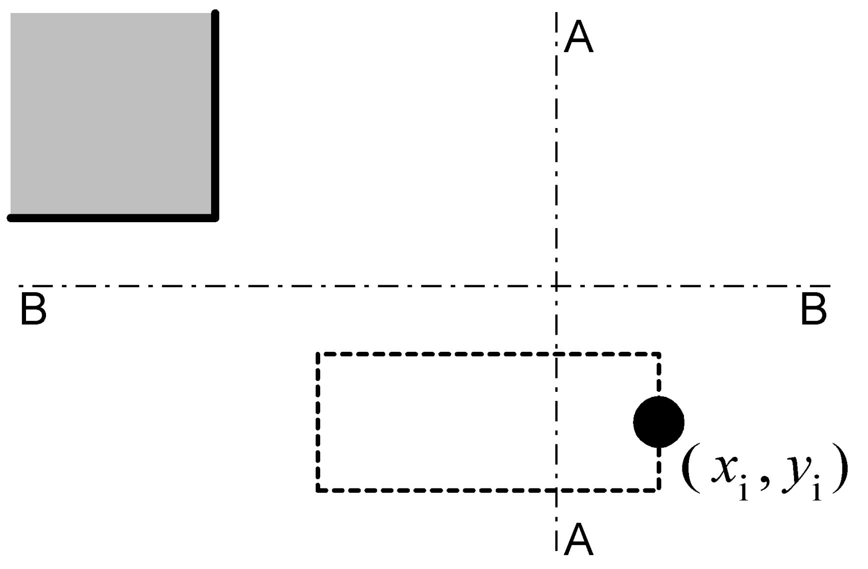


Figure 4-8: Example Scenario for Presolve

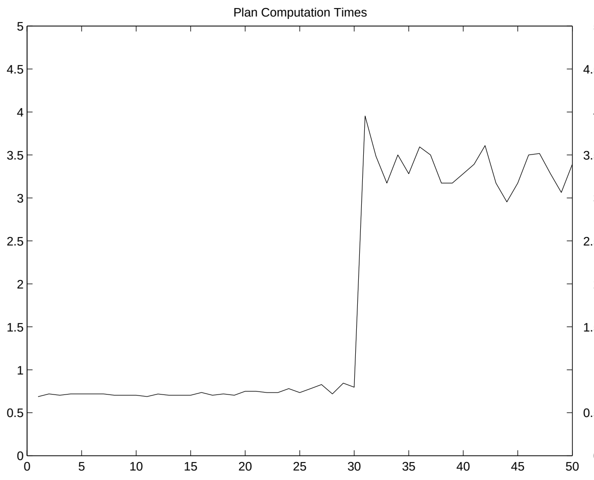
Each of these constraints is satisfied by the existing solution and each would prevent impingement, so the inclusion of any one of them would ensure stable, impingement-free operation. A position constraint is always preferable to a firing constraint, since prevention of firing reduces the ability of the presolve to compensate for unmodeled effects. An algorithm has been implemented to select the settings according to this preference. If a position constraint can be used to prevent impingement, it is enforced, otherwise firing is prevented. If multiple position constraints can be used, the one that is 'least active' is enforced. For example, constraint 2 would be used for the scenario in Fig. 4-8, as the vehicle is further from line B–B than from line A–A.

### 4.4.2 Example: ISS Rendezvous with Presolve

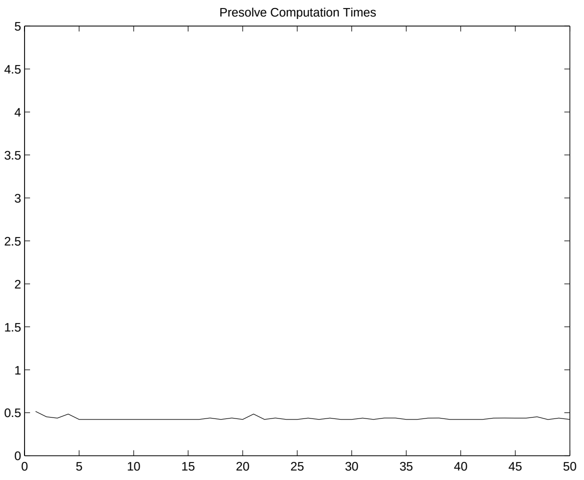
This technique has been applied to the example of the ISS rendezvous shown in Fig. 4-1. The results are shown in Fig. 4-9. Figs. 4-9(a) and 4-9(b) compare the solution times of the full MILP optimization and the LP presolve respectively. The presolve is solved in less than a second at every step. Figs. 4-9(c) and 4-9(d) compare the cost results from the two optimizations. At every step, the presolve is able to reproduce the performance of the full MILP. This exact match of cost results is not seen for all starting points, but observations show that the LP presolve cost is usually no more than 5% higher than the full MILP cost.

Note that the LP presolve was able to find the change in planned final approach direction that occurred during this maneuver. The initial plan involves approaching the target in a direction parallel to the station wall, as seen in Fig. 4-2(a). The final approach followed under MPC is away from the station, as in Fig. 4-2(b). The algorithm for selecting the binary settings was designed to give maximum firing flexibility in the presolve problem. While the initial plan did not involve firing away from the station at the final step, it would still have been possible to do so without impinging. By allowing this possibility in its constraints, the presolve was able to find the revised approach plan when it became optimal for the time available. This demonstrates that the selection of binary settings, particularly when they are not uniquely determined by the previous solution, has a strong impact on the performance of the controller.

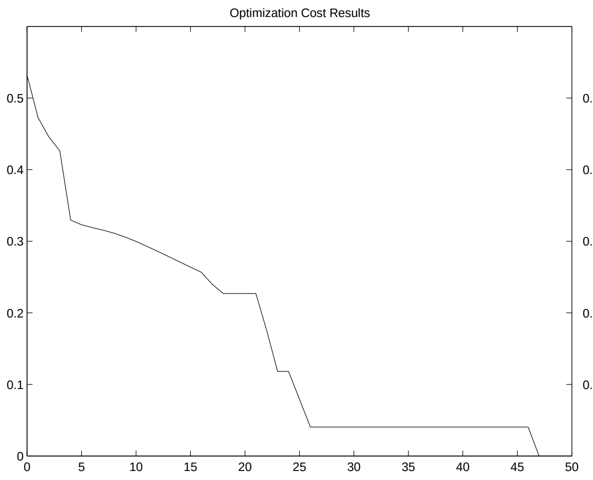
These results suggest the possibility of the application of MPC in real time with reduced computing resources. It would be possible to dispense with the MILP optimization at each step and solve only the LP presolve: the binary settings are determined from the state and control histories, not the actual binary variables from the previous plan. As long as the previous plan satisfies the avoidance constraints, feasible binary settings can be found for the next presolve. In the example in Fig. 4-9, proceeding with the LP solution alone would give the same trajectory and fuel use as using the full MILP solution. It would still be necessary to solve the MILP for the first step, but this could be done off-line before starting the maneuver.



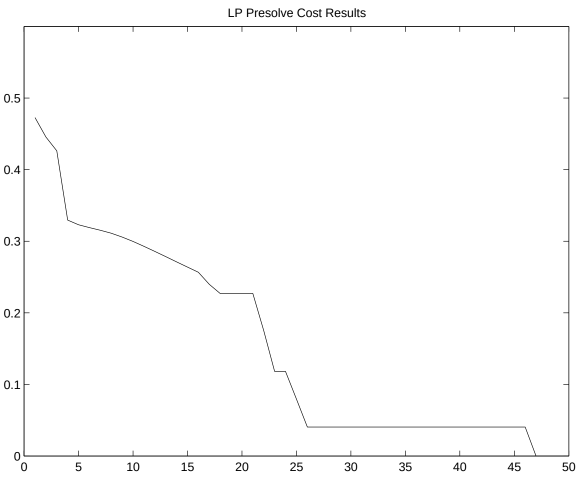
(a) Full MILP Optimization Times



(b) LP Presolve Times



(c) Full MILP Optimization Costs



(d) LP Presolve Costs

Figure 4-9: Solution Times and Costs for On-Line Optimizations

## 4.5 Summary

This chapter has demonstrated the potential of MPC for application to real-time spacecraft control subject to avoidance constraints. By repeatedly solving a MILP trajectory optimization from Chapter 2, provably-stable control is achieved, able to compensate for unmodeled effects. State penalty weightings are used to help convergence to the target, and soft constraints can be added to ensure feasibility, and corresponding robustness, in the presence of realistic disturbances and noise. All of the approximation techniques developed in Section 2.5 can be employed to accelerate the MILP solution process such that it can be employed in real-time. A LP presolve can also be performed, providing a back-up plan to guarantee stability given the finite computation time available.

# Chapter 5

# Conclusion

## 5.1 Discussion

This thesis has demonstrated the potential of solving a variety of trajectory optimization problems using Mixed-Integer Linear Programming (MILP). The problems all have the common feature of non-convexity, arising either from avoidance constraints in the operating space, or from the inclusion of target assignment in the problem. This non-convexity is represented by integer variables in a linear optimization (a MILP), which is consequently solved using powerful, commercial software.

The technique has been applied to problems involving spacecraft and aircraft. Spacecraft problems involving proximity operations, such as formation flying and rendezvous, have been shown to be well-suited to solution by this approach. Aircraft problems involving fleets of autonomous UAVs are readily solved using MILP, which can capture the inherent coupling between path-planning and assignment that makes these problems so difficult.

The MILP approach is limited by its computational complexity. Like the original path-planning problems, their representations in MILP form are $\mathcal{NP}$-hard. While the linearity of the problem enables globally optimal solutions to be found in many instances, the general problem remains an intensive, centralized computation. Techniques have been presented to improve the efficiency of the computation, and solution times have been presented to evaluate the limits of the current method. It performs well for problems of limited size: problems involving up to five vehicles or ten obstacles have been shown to solve in practical times.

An extension to the use of MILP for trajectory design is its incorporation in Model Predictive Control (MPC). In this scheme, a trajectory design optimization is solved on-line at each execution step. This technique has been demonstrated to have the benefits of feedback control, compensating for uncertainty, for maneuvering problems involving avoidance constraints.

## 5.2 Contributions

The following list summarizes the novel contributions in this thesis.

- Extension of the avoidance formulation to include plume impingement constraints for spacecraft (Section 2.4)
- Development of techniques for accelerating solution times, including an iterative technique and time-step grouping (Section 2.5)
- Development of an approximate model of aircraft dynamics, extending the MILP approach to aircraft problems (Section 3.3)
- Inclusion of a general form of vehicle assignment, including timing constraints and heterogenous vehicle capabilities, tailoring MILP for UAV co-ordination problems (Section 3.4)
- Demonstration of the use of MILP in an MPC scheme, offering compensation for uncertainty in real-time, with provable stability (Chapter 4)

## 5.3 Recommendations for Future Work

Future work divides into two distinct areas: the “full-horizon” problem and MPC. The “full-horizon” category includes extensions to the work in both Chapters 2 and 3, since there is considerable cross-over between the spacecraft and aircraft problems in this form.

### 5.3.1 Future Work for Full-Horizon MILP

Developments in this area should begin with the pursuit of faster solutions to the problems. The iterative scheme for plume impingement avoidance has been shown to offer substantial reductions in computation time. Iterative approaches in operations research [59] have shown similar benefits for scheduling problems. This principle might extend to other constraints considered in this thesis, such as collision avoidance and assignment.

In addition, the formulation could be extended to cover more complicated path-planning scenarios, such as design in the presence of uncertain information. Analysis tools could be added to assist the user in interpreting the results, such as sensitivity analysis and identification of active constraints.

### 5.3.2 Future Work for MILP/MPC

The coverage of MPC in Chapter 4 has demonstrated the potential of the method, but much remains to be examined in detail. The stability analysis should be extended to include uncertainty in the model and environment, leading to some provable level of robustness. General forms of robustness analysis have been developed for MPC [62], but their application has been restricted to certain classes of systems, excluding the non-convex constraints in this thesis. This should be extended to the non-convex case. A further extension would consider the transient timing in the analysis. For maneuver planning, the transient from initial condition to final state is the only important consideration. These investigations should aim for a formulation with provable robustness and timing properties.

MPC is also the most demanding application of MILP in terms of computation, since it involves real-time operation. Therefore, MPC would benefit from the work discussed in Section 5.3.1 concerning accelerated solution of MILP problems. Further research might consider special techniques for MPC implementation to guarantee stability and performance properties subject to the availability of limited solution time.

# Bibliography

[1] F. H. Bauer, J. Bristow, D. Folta, K. Hartman, D. Quinn, J. P. How, "Satellite Formation Flying Using an Innovative Autonomous Control System (AUTOCON) Environment," in the proceedings of the *AIAA/AAS Astrodynamics Specialists Conference*, AIAA Paper 97-3821, AIAA, Reston, VA, 1997.

[2] F. H. Bauer, K. Hartman, E. G. Lightsey, "Spaceborne GPS: Current Status and Future Visions," in the proceedings of the *ION-GPS Conference*, Institute of Navigation, Alexandria, VA, 1998, pp. 1493–1508.

[3] F. H. Bauer, K. Hartman, J. P. How, J. Bristow, D. Weidow, and F. Busse, "Enabling Spacecraft Formation Flying through Spaceborne GPS and Enhanced Automation Technologies," in the proceedings of the *ION-GPS Conference*, Institute of Navigation, Alexandria, VA, 1999, pp. 369–384,.

[4] A. Das and R. Cobb, "TechSat21 – Space Missions Using Collaborating Constellations of Satellites," in the proceedings of the *12th Annual AIAA/USU Conference on Small Satellites,* AIAA, Reston, VA, SSC98-VI-1, August 1998.

[5] A. Robertson, G. Inalhan, and J. P. How, "Spacecraft Formation Flying Control Design for the Orion Mission," in the proceedings of the *AIAA Guidance, Navigation, and Control Conference*, AIAA, Reston, VA, August 1999, pp. 1562–1575.

[6] G. Inalhan, F. D. Busse, and J. P. How, "Precise Formation Flying Control Of Multiple Spacecraft Using Carrier-phase Differential GPS," in the proceedings

of the *AAS/AIAA Spaceflight Mechanics Meeting*, AIAA, Reston, VA, January 2000, pp. 151–165.

[7] M. Tillerson, G. Inalhan, and J. How, "Coordination and Control of Distributed Spacecraft Systems Using Convex Optimization Techniques," accepted for publication in the *International Journal of Robust and Nonlinear Control*, Wiley, New York NY, Aug. 2001.

[8] M. Tillerson and J. How, "Formation Flying Control in Eccentric Orbits," in the proceedings of the *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2001-4092, AIAA, Reston, VA, August 2001.

[9] J. H. Reif, "Complexity of the Mover's Problem and Generalizations," in the proceedings of the *20th IEEE Symposium on the Foundations of Computer Science,* IEEE, Washington DC, 1979, pp. 421-427.

[10] H. P. Williams and S. C. Brailsford, "Computational Logic and Integer Programming," in *Advances in Linear and Integer Programming,* Editor J. E. Beasley, Clarendon Press, Oxford, 1996, pp. 249–281.

[11] A. Bemporad and M. Morari, "Control of Systems Integrating Logic, Dynamics, and Constraints," in *Automatica,* Pergamon / Elsevier Science, New York NY, Vol. 35, 1999, pp. 407–427.

[12] C. A. Floudas *Nonlinear and Mixed-Integer Programming – Fundamentals and Applications*, Oxford University Press, New York NY, 1995 pp. 95–107.

[13] R. Subramanian, R. Scheff, Jr., J. Quillinan, D. Wiper, and R. Marsten, "Coldstart: Fleet assignment at Delta Air Lines," *Interfaces,* Institute for Operations Research and Management Sciences, Linthicum MD, 24(1), Jan.-Feb. 1994, pp. 104–20.

[14] *ILOG AMPL CPLEX System Version 7.0 User's Guide*, ILOG, Incline Village, NV, 2000, pp. 17–53,

[15] M. H. Kaplan, *Modern Spacecraft Dynamics and Control,* Wiley, New York NY, 1976 pp 108–115.

[16] R. Sedwick, D. Miller, and E. Kong, "Mitigation of Differential Perturbations in Synthetic Apertures Comprised of Formation Flying Satellites," *Advances in Astronautical Sciences: Space Flight Mechanics*, Univelt, San Diego, Vol. 102, 1999, pp. 323–342.

[17] J. Barraquand, L. Kavraki, J.-C. Latombe, R. Motwani, T.-Y. Li, and P. Raghavan, "A Random Sampling Scheme for Path Planning," *International Journal of Robotics Research,* MIT Press, Cambridge MA, Vol. 16, No. 6, 1997, pp. 759–774.

[18] J. Krozel and M. Peters, "Strategic Conflict Detection and Resolution for Free Flight," in the proceedings of the *36*th *Conference on Decision & Control,* IEEE, Washington D.C., December 1997, pp. 1822-1828.

[19] Z.-H. Mao and E. Feron, "Stability of Intersecting Aircraft Flows under Decentralized Conflict Avoidance Rules," in the proceedings of the *AIAA Guidance, Navigation and Control Conference,* AIAA, Reston, VA, August 2000, pp. 1042–1052.

[20] A. B. Roger and C. R. McInnes, "Safety Constrained Free-Flyer Path Planning at the International Space Station," *Journal of Guidance Control and Dynamics,* AIAA, Reston, VA, Vol. 23, No. 6, Dec. 2000, pp. 971-979.

[21] G. Singh and F.Y. Hadaegh "Collision Avoidance Guidance for Formation Flying Applications," in the proceedings of the *AIAA Guidance Navigation and Control Conference*, AIAA Paper 2001-4088, AIAA, Reston VA, August 2001.

[22] M. B. Milam, K. Mushambi and R. M. Murray, "A New Computational Approach to Real-Time Trajectory Generation for Constrained Mechanical Systems," in the proceedings of the *39*th *IEEE Conference on Decision and Control,* IEEE, Washington DC, 2000, pp. 845-851.

[23] P. T. Spehar and T. Q. Le “Automating an Orbiter Approach to Space Station Freedom to Minimize Plume Impingement,” in *NASA Automated Rendezvous and Capture Review,* NASA N93-22305, Williamsburg, VA, NASA, Washington DC, 1993.

[24] J. How, R. Twiggs, D. Weidow, K. Hartman, and F. Bauer, “Orion: A low-cost demonstration of formation flying in space using GPS,” in *AIAA Astrodynamics Specialists Conference*, AIAA, Reston, VA, August 1998, pp. 276-286.

[25] P. Ferguson, F. Busse, B. Engberg, J. How, M. Tillerson, N. Pohlman, A. Richards, and R. Twiggs, “Formation Flying Experiments on the Orion-Emerald Mission,” presented at the *AIAA Space 2001 Conference*, AIAA Paper 2001-4688, AIAA, Reston, VA, August 2001.

[26] C. A. Beichman, “The Terrestrial Planet Finder: The search for life-bearing planets around other stars,” in the proceedings of the *SPIE Conference on Astronomical Interferometry*, Society of Photo-Optical Instrumentation Engineers, Bellingham, WA, Vol. 3350, 1998, pp. 719-723.

[27] H-.H. Yeh and A. Sparks, “Geometry and Control of Satellite Formations,” in the proceedings of the *American Control Conference*, IEEE, Washington DC, Vol. 1, June 2000, pp. 384-388.

[28] G. Inalhan, M. Tillerson, and How, J. P., “Relative Dynamics & Control of Spacecraft Formations in Eccentric Orbits,” *Journal of Guidance Control and Dynamics*, AIAA, Reston, VA, Vol. 25, No. 1, Jan. 2002, pp. 48-59.

[29] K. T. Alfriend, H. Schaub, and D.-W. Gim, “Gravitational Perturbations, Nonlinearity and Circular Orbit Assumption Effects on Formation Flying Control Strategies,” *Advances in the Astronautical Sciences, Guidance and Control*, Univelt, Inc., San Diego, CA, Vol. 104, 2000, pp. 139–158.

[30] P. K. C. Wang and F.Y. Hadaegh "Optimal Formation-Reconfiguration for Multiple Spacecraft," in the proceedings of the *AIAA Guidance Navigation and Control Conference,* AIAA, Reston, VA, August, 1998, pp. 686-696.

[31] R. E. Burkard and E. Çela, "Linear Assignment Problems and Extensions," In Z. Du and P. Pardalos, editors, *Handbook of Combinatorial Optimization,* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999, pp. 75-149.

[32] M. Tillerson and J. How, "Advanced Guidance Algorithms for Spacecraft Formation Flying," to appear at *2002 American Control Conference,* IEEE, Washington, DC, September, 2002.

[33] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization,* Athena Scientific, Belmont MA, 1997, pp. 17–19.

[34] T. Schouwenaars, B. DeMoor, E. Feron and J. How, "Mixed Integer Programming for Multi-Vehicle Path Planning," in the proceedings of the *European Control Conference,* European Union Control Association, Porto, Portugal, September, 2001, pp. 2603-2608.

[35] T. Schouwenaars, "Mixed integer programming for optimal collision-free path planning of autonomous vehicles," SM Thesis, Katholieke Universiteit Leuven, Belgium, May 2001.

[36] A. Richards, J. How, T. Schouwenaars and E. Feron, "Plume Avoidance Maneuver Planning Using Mixed Integer Linear Programming," in the proceedings of the *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2001-4091, AIAA, Reston, VA, August 2001.

[37] D. E. Kirk, *Optimal Control Theory: an Introduction,* Prentice Hall, Upper Saddle River NJ, 1970, pp. 260–262.

[38] R. Fourer, D. M. Gay, and B. W. Kernighar, *AMPL, A modeling language for mathematical programming*, Boyd & Fraser, Danvers MA, (originally published by The Scientific Press) 1993, pp 291–306.

[39] A. Richards, T. Schouwenaars, J. How, E. Feron, "Spacecraft Trajectory Planning With Collision and Plume Avoidance Using Mixed-Integer Linear Programming," accepted for publication in the *Journal of Guidance, Control and Dynamics*, AIAA, 2002

[40] A. Bicchi, L. Pallottino "On Optimal Cooperative Conflict Resolution for Air Traffic Management Systems," *IEEE Trans. on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 221-231, Dec. 2000

[41] T.S. Perry, "In Search of Future of Air Traffic Control," *IEEE Spectrum*, vol. 34, pp. 18-35, Aug. 1997.

[42] P.R. Chandler, M. Pachter, S. Rasmussen, "UAV Cooperative Control," *IEEE ACC*, June 2001.

[43] C. Schumaker, P. Chandler, S. Rasmussen, "Task Allocation for Wide Area Search Munitions via Network Flow Optimization" proceedings of the *AIAA Guidance, Navigation, and Control Conference and Exhibit,* Montreal, Canada, Aug. 6-9, 2001.

[44] Z.-H. Mao, E. Feron, K. Bilimoria "Stability of Intersecting Aircraft Flows under Decentralized Conflict Avoidance Rules," *AIAA GN&C Conf.,* Aug. 2000.

[45] T. Schouwenaars, B. DeMoor, E. Feron and J. How, "Mixed integer programming for safe multi-vehicle cooperative path planning," *ECC,* 2001.

[46] A. G. Richards, J. P. How, "Aircraft Trajectory Planning with Collision Avoidance using Mixed Integer Linear Programming," submitted to the *American Control Conference,* 2002.

[47] A. Bemporad and M. Morari, "Control of Systems integrating Logic, Dynamics and Constraints," *Automatica* 35(1999), Pergamon Press, UK, pp. 407-427

[48] H. P. Williams and S. C. Brailsford, "Computational Logic and Integer Programming," in *Advances in Linear and Integer Programming,* Editor J. E. Beasley, Clarendon Press, Oxford, 1996, pp. 249–281.

[49] H. P. Rothwangl, "Numerical Synthesis of the Time Optimal Nonlinear State Controller via Mixed Integer Programming," *ACC,* 2001.

[50] J.-H. Chuang, "Potential-Based Modeling of Three-Dimensional Workspace for Obstacle Avoidance," *IEEE Transactions on Robotics and Automation,* Vol. 14, No. 5, October 1998.

[51] R. Ghosh and C. Tomlin, "Maneuver Design for Multiple Aircraft Conflict Resolution," *Proceedings of the American Control Conference*, Chicago, IL, June 2000, IEEE, Washington, DC, pp. 672-676.

[52] J. S. Bellingham, A. G. Richards and J. P. How, "Receding Horizon Control of Autonomous Aerial Vehicles", to appear at the *American Control Conference,* 2002.

[53] J. S. Bellingham, M. J. Tillerson, A. G. Richards, J. P. How, "Multi-Task Assignment and Path Planning for Cooperating UAVs," to appear in *"Cooperative Control: Models, Applications and Algorithms,"* Editors S. Butenko, R. Murphey, and P. Pardalos, Kluwer Academic Publishers, 2002.

[54] S. A. Bortoff, "Path Planning for UAVs," *Proceedings of the American Control Conference*, Chicago, IL, June 2000, IEEE, Washington, DC, pp. 364–368.

[55] P. R. Chandler and M. Pachter, "Research Issues in Autonomous Control of Tactical UAVs," *Proceedings of the American Control Conference*, Philidelphia, PA, June 1998, IEEE, Washington, DC, pp. 394–398.

[56] T. W. McLain and R. W. Beard, "Trajectory Planning for Coordinated Rendezvous of Unmanned Air Vehicles," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Paper No. AIAA-2000-4369, AIAA, Reston, VA, 2000.

[57] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamics Planning," *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, IEEE, Washington, DC, 1999, pp. 473–479.

[58] E. Frazzoli, M. A. Dahleh, E. Feron, “Real-Time Motion Planning for Agile Autonomous Vehicles,” *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Paper No. AIAA-2000-4056, AIAA, Reston, VA, 2000.

[59] V. Jain and I.E. Grossmann, “Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems”, *INFORMS Journal of Computing*, 13, 258-276 (2001).

[60] J.M. Maciejowski, *Predictive Control with Constraints*, Prentice Hall, England, 2002.

[61] D. Q. Mayne, J. B. Rawlings, C. V. Rao, P. O. M. Scokaert, “Constrained Model Predictive Control: Stability and Optimality,” *Automatica*, 36(2000), Pergamon Press, UK, pp. 789–814.

[62] E. C. Kerrigan and J. M. Maciejowski, “Robust Feasibility in Model Predictive Control: Necessary and Sufficient Conditions,” in *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, FL, December 2001, IEEE, Washington, DC, pp. 728–733.

[63] I. Kawano, M. Mokuno, T. Kasai, T. Suzuki, “Result of Autonomous Rendezvous Docking Experiment of Engineering Test Satellite-VII,” *Journal of Spacecraft and Rockets*, Vol. 38, No. 1, January 2001, AIAA, Reston, VA, pp. 105–111.

[64] D. Waltz, *On-Orbit Servicing of Space Systems*, Krieger Publishing, FL, 1993, pp. 193–227.

[65] M. E. Polites, “An Assessment of the Technology of Automated Rendezvous and Capture in Space,” NASA Technical Report, No. TP-1998-208528, NASA MSFC, July 1998

[66] DARPA Orbital Express Website, `http://www.darpa.mil/tto/PROGRAMS/astro.html`, April 2002

[67] Smoothed Analysis Website, `http://www-math.mit.edu/~spielman/simplex/`, April 2002

[68] TechSat 21 Factsheet, available at `http://www.vs.afrl.af.mil/Factsheets/techsat21.html`, April 2002, Air Force Research Lab.

[69] V. Manikonda, P. Arambel, M. Gopinathan, R. K. Mehra and F. Y. Hadaegh, "A Model Predictive Control-based Approach for Spacecraft Formation Keeping and Attitude Control," in *Proceedings of the American Control Conference*, June 1999, IEEE, Washington DC, pp 4258-4262.

[70] W. B. Dunbar, M. B. Milam, R. Franz and R. M. Murray, "Model Predictive Control of a Thrust-Vectored Flight Control Experiment," accepted for *15th IFAC World Congress on Automatic Control*, 2002