Reconfiguration of a Value Function-Based Guidance System

by

Mario J. Valenti

B.S.M.E., Temple University (2000)
B.S.E.E., Temple University (2000)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2003

Author . . . .
Department of Electrical Engineering and Computer Science
May 21, 2003

Certified by . . . .
Eric Feron
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Reconfiguration of a Value Function-Based Guidance System

by

Mario J. Valenti

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

In this thesis, we investigate and discuss strategies for trajectory planning and guidance-level reconfiguration following changes in aircraft closed-loop dynamics. The guidance model for the vehicle under study is a maneuver automaton. We discuss reconfiguration methods to handle changes in the guidance model using the Guidance Value Function that is calculated from the solution to the underlying optimal trajectory generation problem. Finally, we implement and evaluate these reconfiguration methods using numerical examples of Fixed-Wing Unmanned Aerial Vehicles (UAV).

Thesis Supervisor: Eric Feron
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

First and foremost, I would like to express gratitude to my advisor, Prof. Eric Feron, for his assistance, advice and direction in my brief time at MIT. I appreciate his enthusiasm and honesty. He has helped me to grow as a researcher and as a person, and I look forward to continuing our research efforts in the future.

Second, I would like thank my friend and colleague, Dr. Bernard Mettler, for “showing me the ropes,” working with me to improve my writing and helping me to find a voice in research. I cannot thank him enough for all of the time he has spent with me over the past two years and I look forward to continuing our research efforts in the future.

Third, I would like to thank my friends and colleagues in LIDS and beyond who have influenced my research - Francis Carr, Jan De Mot, David Dugail, Emilio Frazzoli, Vladislav Gavrilets, Sommer Gentry, Masha Ishutkina, Vishwesh Kulkarni, Julius Kusuma, Ioannis Martinos, Keith Santarelli, Tom Schouwenaars, and Kazutaka Takahashi. I would also like to thank the students and staff of Simmons Hall and members of the Eastgate Bible Study. Both Tricia and I appreciate their support and kindness. They have made living at MIT an wonderful experience.

Fourth, I would like to express my appreciation to Dr. Siva Banda, Mr. Phillip Chandler and the rest of the researchers at the Air Force Research Laboratory, Air Vehicles Directorate for the opportunity to work at AFRL with them during the Summer of 2002. In addition, I would like to thank to John Thatcher, Bradley Beckman, Maryann Skehan and the rest of the V-22 Vehicle Management Systems Flight Control Systems group at the Boeing Company in Philadelphia for supporting my academic goals.

Fifth, I would like to say thanks to Joshua Jones, Brian Kuhns, Dan McCarter, Robert Morsa and Steve Scaduto who have stood by me through thick and thin. I would like to thank my family, especially my brother Phillip, who is serving this country faithfully in harm’s way, my brother Justin, who always has creative ideas and a way with words, my mother, who has comforting thoughts in the best and worst of times, and my father, a voice of reason in challenging situations and for being a role model for me. He is one of the best engineers I have ever met.

Sixth, I would like to express my love to my wife Tricia, who has made this experience fun, exciting and bearable, even when its been stressful on both of us.

Finally, I would like to express my deepest gratitude and thanks to my Lord and savior Jesus Christ, by which *all* things are possible. Without You, I do not know where I would be... *Phil. 3:12-14*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Most unmanned aerial vehicles (UAVs) are controlled remotely by skilled human operators. Although UAVs will require human involvement for many years, the need for skilled pilots could be reduced gradually with improvements in UAV guidance systems. Currently, most UAV guidance systems do not exhibit the performance and flexibility needed to allow a non-pilot operator to guide a UAV throughout an entire mission. In addition, many of today's UAV guidance systems cannot satisfy task requirements at every stage of the vehicle's mission. For example, most UAV takeoff and landing tasks are performed manually, while unexpected changes in the vehicle's operating environment often require the guidance system to be disabled.

Computational limitations are one of many reasons most UAV guidance systems are unable to command the vehicle through complex tasks during a mission. Since many systems lack the memory or computational speed to process the data needed to make rapid decisions autonomously, many computationally efficient techniques used for on-line trajectory generation are based on simplified vehicle models. For example, some path planning techniques calculate trajectories in the X-Y plane using the Dubins Path [6, 8, 21]. Dubins showed that given a vehicle's starting and ending conditions, the minimal distance path between any two points can be found using line segments in either a Turn Circle-Straight Line-Turn Circle (TST) configuration or a Turn Circle-Turn Circle-Turn Circle (TTT) configuration. This method has been used to simplify path planning for UAVs with success in previous work [13, 8, 21]. For example, using this technique, Fowler [8] showed that a "rabbit" could be used to compensate for differences in the actual and commanded vehicle position as generated by the TST/TTT algorithm. This method was used to calculate nominal vehicle trajectories based on the Dubins path to be used in task allocation research.

On the other hand, solutions that account for vehicle dynamics may not be computationally tractable for practical use. For example, to fly UAVs autonomously, the flight control and guidance systems must account for a large set of state variables over the vehicle's entire flight envelope. Since we may want to optimize the vehicle's trajectory given mission constraints, the guidance system must consider every vehicle state in planning the optimal path to the goal. Using today's technology, this mammoth task is near impossible to calculate on-line in a dynamic environment.

A variety of methods have been introduced to address some of these computational issues. For example, methods based on an inverse dynamics approach have been proposed to make the guidance task computationally tractable. These methods enjoy considerable experimental and industrial success, although it can be difficult to effectively account for actuator and state constraints. Alternative methods have been proposed to address these limitations. For example, Faiz et al [7] proposes a transformation on the system to allow the incorporation of constraints in the generation of the path. Verma [20] proposes a solution to address system constraints, but relaxes the differential flatness requirement on the system.

In this thesis, we focus on a value function-based guidance approach. This technique, proposed by Frazzoli et al. [9, 10], requires little on-line computations and can account for a broad range of the vehicle's capabilities. It relies on a motion primitive representation of the vehicle's dynamics: The Maneuver Automaton (MA). The MA is a quantized subset of the feasible trajectory primitives for the vehicle [9]. Though this simplified structure constrains the vehicle to specific operating regions of its flight envelope, it provides many advantages. Using the MA approach, we can select the vehicle's operating states to maximize vehicle performance for a given mission. Then using methods like dynamic programming, we can calculate the system's guidance value function (a representation of the vehicle's cost-to-go to a desired destination from its current state) off-line to reduce the vehicle's on-line computational load. Most importantly, the on-line guidance problem becomes tractable. Therefore, though we constrain the vehicle to operate within the MA structure, we gain an efficient method for calculating realistic trajectories based on the vehicle's capabilities.

In summary, motion planning with a MA is computationally attractive, addresses system constraints, and can deliver the full vehicle performance. But, how robust is this guidance technique to changes in the vehicle or its operating environment? For example, if the vehicle dynamics, mission goals or operating environment changes, is the vehicle's nominal guidance value function representative of the system and can it provide a guidance solution that is tractable?

This question brings us to the issue of guidance system reconfigurability. The need for reconfiguration is well recognized when dealing with changes in the inner-loop dynamics, and it has been the subject of much research. Programs, such as RESTORE [22], have led to inner-loop reconfiguration strategies mature enough to be considered for implementation on new combat air vehicles such as UCAV [1].

Inner-loop reconfiguration is rooted in well-known results, such as the intrinsic robustness properties of the Linear Quadratic Regulator. By contrast, guidance-loop level reconfiguration comes as a much less-studied topic. In the context of this thesis, guidance-level reconfiguration pertains to updating the trajectory-planning algorithm in the face of changes in the closed-loop vehicle dynamics. Therefore, guidance-loop level reconfiguration must occur whenever inner-loop reconfiguration occurs.

To address this question, some researchers have proposed adaptive schemes that adjust vehicle parameters on-line to account for changes in the vehicle state. For example, in [15], Schierman et al. discuss an adaptive guidance reconfiguration technique for landing trajectories for the X-33 Reusable Launch Vehicle which uses Calculus of Variations to generate a series of trajectory libraries off-line that can be adapted to produce feasible trajectories on-line. In this document, we focus on guidance-loop reconfiguration when the closed-loop system under reconfiguration is either a simple

Linear Time-Invariant system or the Robust Maneuver Automaton (RMA).

1.2 Document Outline

In this thesis, we study reconfiguration issues for a value function-based guidance system. After a brief introduction on guidance techniques for unmannned vehicles, we define and describe the Maneuver Automaton and how it is used in a value function-based guidance system. Then, we focus on some of the theoretical and practical issues relating to the reconfiguration of these systems.

Using this discussion, we perform reconfiguration case-studies using simple Linear Time-Invariant (LTI) and Maneuver Automaton (MA) based systems. We use these case studies to help us understand and explore the following issues:

- Trajectory reconfiguration arising from changes in the guidance-level system dynamics
- Trade-offs between the system robustness to failure and system performance
- The fundamental theoretical issues relating to the stability and controllability of the system
- How the nominal value function can be used to develop tractable trajectories for the reconfigured system
- Practical issues relating to the implementation of this approach on an UAV platform

Following a discussion of these issues, we illustrate and discuss our reconfiguration approach using a full non-linear model of a Unmanned Combat Aerial Vehicle system. Finally, we conclude with a summary of our results and discuss future implementations and recommendations.

Chapter 2

Value Function-Based Guidance Systems

In this chapter, we provide an overview of value function-based guidance systems when the system under study is the Maneuver Automaton (MA). We investigate system properties (e.g. stability, controllability, robustness) for these guidance systems, and define and explore guidance system reconfiguration in this framework.

2.1 Maneuver Automaton (MA) Overview

In [9, 10] Frazzoli et. al. use a motion primitive representation of the vehicle dynamics to reduce the computational complexity of the real-time motion planning problem. This Maneuver Automaton (MA) representation of the system can be used to describe complex non-linear systems in terms of a finite collection of motion primitives. These primitives are divided into two categories: Trim Trajectories and Maneuvers. The trim trajectories correspond to equilibria (trim states) of the system, and maneuvers are defined as finite duration transitions between the trim states [19]. Figure 2-1 shows a graph of a simple maneuver automaton for a fixed-wing aircraft. The trim states, shown by the ovals, correspond to desirable operating states for the vehicle, e.g. level flight trim at a set speed, climb and descent trims along set flight path angles, left and right turn trims at a set speed, etc. The maneuvers are shown as

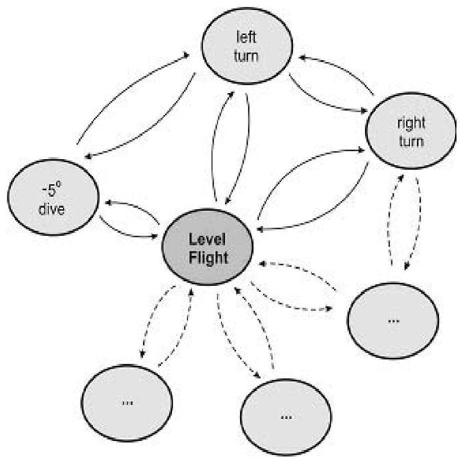


Figure 2-1: Examples of trim and maneuver states composing a Maneuver Automaton for a fixed-wing aircraft

arrows between the trim states. Each maneuver is designed to begin and end at a specific pair of trim states. In this thesis, we define Q_T as the set of all admissible trim states for a system H, and Q_M as the set of all admissible maneuvers.

The MA's state and controls contain mixed discrete and continuous variables. Each trim state is designated by an integer variable $q \in Q_T$; the evolution on a trim is described by the continuous dynamics corresponding to the trim condition and parameterized by a continuous coasting time t_q. Maneuvers are designated by an integer variable $p \in Q_M$; since different options exist to describe the motion of the vehicle along maneuvers, they typically exist in the nonlinear range of the vehicle dynamics. For path planning purposes, maneuvers are described by the pair of trim conditions they connect, the displacement of the vehicle in space and duration in time. The actual position of the vehicle in inertial space remains continuous and is a function of the evolution of the hybrid state (represented by a continuous state vector $x(t)$ and the current maneuver primitive h) [19, 12].

This MA representation of the vehicle dynamics accomplishes two purposes. First, it considerably reduces the size of the system state. Second, it changes the motion planning problem on the continuous space into a discrete sequential decision problem that can be solved off-line using dynamic programming in a tractable fashion [4, 9, 10]. Then, the optimal value function solution to this problem can be stored in the vehicle's

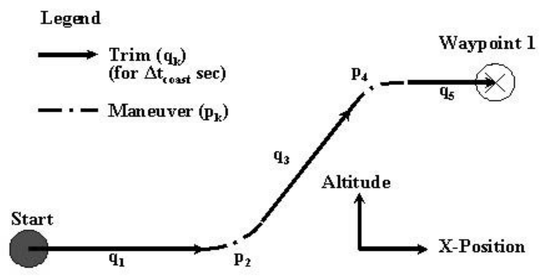


Figure 2-2: Sample trajectory using a Maneuver Automaton for a fixed-wing aircraft

memory and used in real-time applications to guide the vehicle to a destination given the vehicle's initial state and position as depicted in Figure 2-2.

Typically, the solution for this type of problem involves the minimization of a cost functional of the form

$$J(x_0, x_f) = \sum_{automaton\ transitions} g(x, u, t) + Q(x(t_f), x_f).$$

The cost function g depends on the state x and the commands u and encapsulates such notions as fuel consumption, mission execution time, furtivity constraints or any combination of these. In our context, the optimal cost J^* to this problem is characterized by *Bellman's equation*, which must be satisfied over all states

$$J^*(x) = \min_{u \in \mathcal{U}} \mathbf{E}_w \left\{ g(x, u, t) + J^*(f(x, u, w)) \right\} \tag{2.1}$$

where $f(\cdot)$ is the MA state transition function (e.g. depicted in Figure 2-1) which depends on the state x, the commands u and external perturbations w. This equation is typically used along with the boundary condition $J(x(t_f), x_f) = Q(x(t_f), x_f)$ to compute J^* via fixed-point iterations.

2.2 Properties of the Maneuver Automaton

Before we can discuss the effects of reconfiguration on a value function-based guidance system, we must define the notions of controllability and stability for this system.

2.2.1 Controllability

From classical control, the general notions of controllability for a system of the form $\dot{x} = f(x, u)$ from an initial state $x(0)$ are well-known. As posed in [23], a dynamical system described by $\dot{x} = f(x, u)$ is said to be *controllable* if, for any initial state $x(0) = x_o$, $T > 0$ and final state x_f, there exists a (piecewise continuous) input $u(\cdot)$ such that the solution of equation $\dot{x} = f(x, u)$ satisfies $x(T) = x_f$. From this statement, we can derive a similar definition for controllability in the Value-Function based system. Since we have defined our system as the MA, we can describe controllability of the MA in terms of its connectivity [9]:

Definition 2.1 (Controllability of MA-based System) *A Maneuver Automaton-based system is said to be* controllable *if, for any initial trim state $q_o \in Q_M$ and final trim state q_f, there exists a time $T > 0$ and an connected sequence of maneuver primitives $\{\{q_i\}, \{p_i\}\}$ such that $\{q_i\} \subset Q_T$ and $\{p_i\} \subset Q_M$ by which the system transitions from the initial system state $x_{init} = [q_o, x_o]$ to the final system state $x_{final} = [q_f, x_f]$ in time $t_f < T$.*

This definition shows that the maneuver primitives for a Value Function-Based guidance system must be connected in order for the system to be controllable. In other words, the system must be able to maneuver to and from any trim state in the MA in order for the guidance system to use the MA for trajectory planning [9]. If this connectivity condition does not exist for our Maneuver Automaton, the guidance system may plan trajectories using maneuver primitives that move the vehicle into a state from which it is unable to reach the desired final vehicle state. Finally, notice that we define the system state and trim state separately in this definition. The MA-based guidance system uses maneuver primitives to develop feasible trajectories for the vehicle through inertial space. Therefore, the initial and final vehicle states must also include position and orientation state variables. Otherwise, the vehicle may not be able to reach a desired location even though it maintains the capabilities to transition between trim states freely.

2.2.2 Stability

In order to guarantee the connectivity of the states in the MA, we must also form a notion of stability. From classical control, the general notions of stability for a dynamic system of the form $\dot{x} = f(x, u)$ around initial state $x(0)$ are well-known. To discuss the stability of a system, we must define an equilibrium point of the system. A point $\bar{x}$ is an *equilibrium point* of the system $\dot{x} = f(x, u)$ from time t_o if $f(\bar{x}, 0) = 0$, for all $t > t_o$ [5]. Using this definition, we can discuss the notion of *asymptotic stability*:

Definition 2.2 (Stability of a system around an Equilibrium Point) *Given a system $\dot{x} = f(x, u)$:*

- *$\dot{x}$ is called* stable *around its equilibrium point $\bar{x}$ if for every $R < \tilde{R}$, there exists an r where $0 < r < R$ such that if $x(0)$ is inside $S(\bar{x}, r)$, then $x(t)$ is inside $S(\bar{x}, R)$, for all $t > 0$*

- *$\dot{x}$ is called* asymptotically stable *around its equilibrium point $\bar{x}$ if its stable and there exists $\bar{R} > 0$ such that if $x(0)$ is inside $S(\bar{x}, \bar{R})$, then $x(t) \to \bar{x}$ as $t \to \infty$*

where $S(x, \rho)$ is a n-dimensional sphere of radius $\rho > 0$ centered around state x [11].

From this definition, it becomes obvious that the trim states of our MA must be relative equilibria for our system. In addition, maneuvers must also ensure the system remains stable in the transition from the between vehicle states which is robust to perturbations in the system.

To make these guarantees, we must first select trim states, where the state $x(t)$ will *always* remain inside $S(\bar{x}_q, \bar{R}_q)$ even under system perturbations. Second, we must ensure that every maneuver $p_{ij} \in Q_M$ which transitions the system between trim states $q_i, q_j \in Q_T$ ensures that the vehicle state at the end of the maneuver $x(t_p)$ will *always* remain inside $S(\bar{x}_{q_j}, \bar{R}_{q_j})$ even under system perturbations. This set of trims $\{\{q\}, \{p\}\} \subset \{\{Q_T\}, \{Q_M\}\}$ is our *Robust Maneuver Automaton* (RMA) [9]. Figure 2-3 shows an two-dimensional example of a maneuver between two trim states. Here we note that the image of the r_{q_i} under the maneuver p_{ij} must map to an area inside r_{q_j} to guarantee that the state trajectories around q_j will remain inside R_{q_j}.

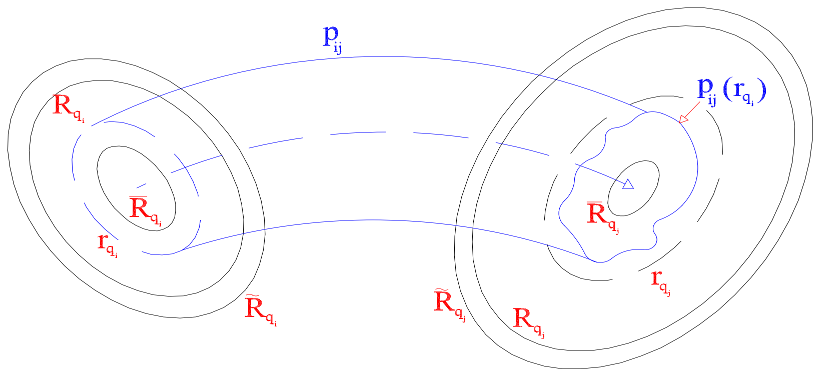


Figure 2-3: Example of Robust Maneuver p_{ij} from Trim q_i to q_j

Next, from [11] we know that given an equilibrium of the system, we can create a non-negative, real-valued, continuous function $V(\cdot)$, called a *Lyapunov function*, defined over a region Ω in the system's state space with a unique minimum at the equilibrium point $\bar{x}_q$, such that for any system trajectory with $x(0) \in \Omega$ where $V(x(0)) < \infty$, then $V(x(0)) \geq V(x(t))$ for all $t > 0$. We can use these functions to provide us with a good understanding of the trim trajectories of the system given an initial condition near an equilibrium state. The problem is that though a Lyapunov function may exist for each trim state $q \in Q_T$, a closed form for this function may be difficult to find.

In this thesis we will assume that each trim state is an equilibrium point for the system and the maneuvers are well-defined, i.e. each maneuver is robust and meets the criteria defined in [9].

2.3 Robust Value Function

This Robust Maneuver Automaton (RMA) representation of a system can be used to describe a complex non-linear system in terms of a finite collection of motion primitives. Since we are interested in the vehicle guidance problem, we can use

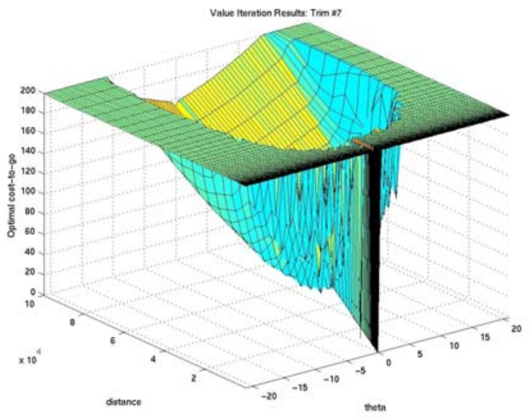


Figure 2-4: Example of Value Function for the Level Flight Trim of 15 Trim Longitudinal MA

these motion primitives to create achievable trajectories for the vehicle. The vehicle trajectories are represented by a hybrid control variable $\{(\tau_q, p)_k, k = 1, ..., N\}$, where k is a decision counter, τ_q is the coasting time for the vehicle in trim state q, and p is the maneuver from the current trim state to the next trim state [12].

Since there may be more than one combination of maneuver primitives that can be used to reach a desired state/destination, we want to optimize the vehicle's trajectory to reach the goal over a desired parameter. Problems of this form can be solved using dynamic programming and the value function $J^*(x)$ can be calculated using Eq. (2.1). This value function is an expression of the optimal cost to reach the intended destination as a function of the current system state and can be used to navigate the vehicle to the desired destination and vehicle state. An example projection of the value function for a level flight trim state from a 15-Trim Longitudinal Axis Maneuver Automaton calculated using a sample RMA is shown in Figure 2-4. This function is non-increasing with a finite cost in the vehicle's "achievable" region (i.e. the region from which the vehicle can reach the desired goal) and has a value zero-cost at the destination point. In fact, we find that along system trajectories, these value functions calculated using Eq. (2.1) meet the following definition from [3]:

Definition 2.3 (Lyapunov-like Function) *Given a strictly increasing sequence of times T in $\mathbb{R}$ (resp. $\mathbb{Z}$), we say that V is a* Lyapunov-like function for function f and trajectory $x(\cdot)$ (resp. $x[\cdot]$) over T *if*

- $\dot{V}(x(t)) \leq 0$ *(resp.* $V(x[t+1]) \leq V(x[t])$ $\forall\ t \in T$
- V *is monotonically non-increasing on* T

This definition provides a link between the value functions calculated for the RMA and Lyapunov functions. It enables us to apply many of the traditional tools used in nonlinear systems analysis to investigate changes in the RMA and its effects on the guidance value function. From these definitions, we seek to gain an understanding of how changes in the RMA will affect the value function. In addition, we seek to understand whether the value function can be used after there is a change to the RMA.

To better understand this topic, we can look at previous results from Linear Systems analysis. In [14], Safonov and Athans showed that for the Linear Quadratic State Feedback (LQSF) regulator, a system that could be formulated in terms of $\min_u J(x,u)$ subject to:

$$\dot{x}(t) = Ax(t) + Bu(t) \text{ where } x(0) = x_0, x(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}^m, A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}$$

with the performance index given by

$$J(x,u) = \int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)]dt \text{ where } Q = Q^T \geqslant 0, R = R^T > 0.$$

The optimal control $u^*(t)$ and associated state-trajectory $x^*(t)$ are given by

$$\left.\begin{array}{c} \dot{x}(t) = Ax^*(t) + Bu^*(t) \text{ where } x^*(0) = x_0 \\ u^*(t) = -Hx^*(t) \equiv -R^{-1}B^TKx^*(t) \end{array}\right\} (\Sigma^*).$$

Here, we find that $K = K^T \geqslant 0$ satisfies the Riccati equation,

$$KA + A^T K - KBR^{-1}B^T K + Q = 0$$

and has a minimal value of the performance index given by

$$J(x^*, u^*) = {x_0}^T K x_0$$

where perturbed versions of Σ^* given by $(\tilde{\Sigma})$ are stable, even under large perturbations in the plant. Safanov and Athans were able to generalize the result that single-input LQSF regulators have infinite gain margin and at least $\pm 60^\circ$ phase margin and at least 50 percent gain reduction tolerance. In addition, this property could be extended to any open-loop system with similar behavior to $(\tilde{\Sigma})$ [14]. Here, the performance index $J(x^*, u^*)$ for this linear system is a Lyapunov function that can stabilize the system despite changes or perturbations in the plant $(\tilde{\Sigma})$. Therefore, $J(x^*, u^*)$, which could be calculated using DP, could be used to help control the system when there are significant perturbations to the plant.

This result provides some insight into how changes in the RMA may affect the stability of the Value Function-based guidance system. Since we have established the Robust Value Function as Lyapunov-like using Def. (2.3), changes in the RMA can be thought of as perturbations to the system. Under this assumption, the Robust Value Function, calculated using Eq. (2.1), may still be valid after drastic changes in the structure of the RMA, and thus could guide the system to the desired state and destination.

2.4 Reconfiguration of the Robust Maneuver Automaton (RMA)

In the real-world, many systems experience changes (both expected and unforeseen), which may require modification to its existing model. Whenever the system model is adjusted or modified to fit the current state of the system, we term this action as a *reconfiguration* of the system. Note that reconfiguration in this context does not imply a failure in the system; instead, the system can be reconfigured to enhance the system performance during a mission without implying a degradation of the system's capabilities.

2.4.1 Guidance-level reconfiguration

In the Robust Maneuver Automaton (RMA) framework, the reconfiguration question refers to how changes in the structure of the nominal RMA, defined by $\{\{Q_T\}, \{Q_M\}\}$ affect the generation of feasible trajectories. The structure of the RMA can be changed in two ways: we can add a motion primitive (e.g to improve vehicle performance), or remove a motion primitive (e.g for a system failure) resulting in reduced performance. After talking about basic controllability issues, we focus our attention to the main object of this thesis, which is to rely heavily on *guidance value functions* as a central tool for guidance-level system reconfiguration.

2.4.2 Maintaining RMA Controllability under Reconfiguration

Several properties, such as controllability and maneuver primitive connectivity, need to be examined when changing the RMA structure. We must guarantee that from a given initial trim q_o and vehicle position in inertial space x_o, it is possible to find an admissible sequence of primitives to guide the vehicle to any desired destination state q_f and vehicle position in inertial space x_f in finite time T [9]. Next, we must insure that the reconfigured RMA contains a minimum set of motion primitives that make the RMA controllable. Any motion primitives not connected to this set must be discarded from the reconfigured RMA to satisfy controllability requirements.

For robustness and reconfiguration concerns, we can separate all maneuver primitives meeting our requirements into two categories [19]: Those which satisfy the controllability constraints, and those which provide enhanced performance. The first set of trims that satisfy our controllability criteria are known as the "controllability core." This "core" set of maneuver primitives define the achievable destination region of the vehicle. If any of these primitives are removed from the RMA, the vehicle will lose the capability to reach the goal from some of the achievable regions identified by the associated primitives. Therefore, if a failure occurs in our system, we want to ensure that all trim states are connected to our controllability core. The second set of maneuver primitives, known as the "performance set," add vehicle performance by taking advantage of the vehicle's capability throughout the flight envelope. These primitives may be essential to a particular mission element or segment, but will not affect vehicle controllability in the event of a failure. Combining the controllability core with the performance set conditions the robustness properties of the RMA to system failures and disturbances.

Though in theory we can select any trim and maneuver set that satisfies the connectivity and controllability requirements, in practice we have found that it is helpful to build some properties into the structure of the maneuver automaton to deal with issues related to reconfiguration. For example, one approach is to promote trim and maneuver selection based on system requirements, such as performance requirements and failure considerations (airframe and control system characteristics).

2.4.3 Reconfiguration and the Guidance Value Function

After we develop a nominal MA that meets connectivity and controllability requirements, we must determine if the nominal value function is valid for guidance for a reconfigured RMA. It is clear that the performance of a diminished RMA may be reduced, and thus the value function will underestimate the cost-to-go for some trim trajectories. Likewise, we know that by adding performance to the nominal RMA, the value function will overestimate the cost-to-go for some trim trajectories.

We propose to reconfigure the trajectory generation system by relying on the guidance value function. Consider the nominal system under nominal optimal guidance. The behavior of the system is unambiguously specified by the nominal value function corresponding J^*_{nom} satisfying Bellman's equation

$$J^*_{nom}(x) = \min_{u \in U_{nom}} \mathbf{E}_w \left\{ g(x,u,t) + J^*(f_{nom}(x,u,w)) \right\}.$$

Assume for example the new dynamics are represented by a new state transition function f_{new} and by a new set of available controls U_{new}. Instead of recomputing the corresponding guidance value function, efficient guidance may simply be achieved by using the *nominal* guidance value function, so as to provide the guidance law

$$u_{rcfg} = \arg \min_{u \in U_{new}} \mathbf{E}_w \left\{ g(x,u,t) + J^*(f_{new}(x,u,w)) \right\}. \tag{2.2}$$

Thus, the nominal value function is used to provide guidance to the reconfigured plant. While such a procedure is still heuristic, it remains rooted in its control equivalent of *control Lyapunov functions*, which have been used in a similar way to devise provably good adaptive control schemes when facing plant dynamics perturbations or changes. Much of our future research work will concentrate on establishing similar properties at the level of the guidance loop. Among these, the ability to maintain *trajectory stability* prevails over all others. Several such properties can be proved by inspection, however. For example, if we assume that the new automaton dynamics for the vehicle consist of *added* maneuvers, it is then easy to show that trajectory stability is maintained. In this case the iteration shown in Eq. (2.2) will in fact work (as shown in the next chapter).

Chapter 3

Reconfiguration Case Study: Value Function-Based Guidance Systems

In this chapter, we develop and discuss reconfiguration case studies for simplified Linear Time-Invariant (LTI) and Maneuver Automaton (MA) value function-based guidance systems. Our goal is to use these case studies to understand the relationships between the value function (VF) of the nominal and reconfigured systems. We explore the properties of the guidance value functions and use these findings to develop methodologies for reconfiguration in this framework. Finally, we summarize the results from these case studies and use our findings to implement reconfiguration strategies on value function-based guidance systems in the next chapter.

3.1 Case Study Overview

As part of our research, we developed a series of case studies to understand the effects of reconfiguration on value-function based guidance systems. These case studies included simulations to investigate the relationships between the properties of well-defined and simple systems and the resulting value function. Using these relationships, we are able to develop reconfiguration methodologies for these systems. Our goal is to understand how the nominal value function for a system can be used to guide and control a reconfigured system.

As discussed in [19], these guidance value functions are used to generate optimal trajectories for a system. We can perform a value iteration with a nominal set of motion primitives to calculate the value function for our RMA. However, if we change any of these motion primitives, this nominal value function will not be optimal for the reconfigured vehicle. Ideally, we could recalculate the value function for the new set of motion primitives. For a simple Fixed Wing MA (15 trim states and 200 maneuvers), it takes a Pentium II - 300 MHz computer about 2 hours to recalculate the value function. This approach is not practical during a mission where some baseline performance must be readily available.

An alternative to this method would be to pre-compute value functions that correspond to different failure scenarios, but these functions would need to be saved for every failure combination. To support the guidance task immediately after a failure without having to perform complex reconfiguration operations, it would be practical to be able to use the nominal value function. Therefore, we would like to establish whether the nominal value function can be used with a different MA. To do so, we use the reconfiguration strategy described by Eq. (2.1) to generate achievable trajectories using the nominal value function.

In this thesis, we define the recalculated value function as the optimal value function for the reconfigured MA. The recalculated value function is computed by running a separate value iteration for the reconfigured MA. In this thesis, we discuss trajectory planning using the following three strategies:

- **Nominal Strategy:** we use the *nominal* value function to generate trajectories for the *nominal* MA,
- **Reconfiguration Strategy:** we use the *nominal* value function to generate trajectories for the *reconfigured* MA
- **Recalculation Strategy:** we use the *recalculated* value function to generate trajectories for the *reconfigured* MA

Our goal is to understand if the reconfiguration strategy can be used to generate trajectories for the reconfigured MA using the nominal value function.

3.2 Case Study I: Double Integrator

The first system we examine is a linear time-invariant (LTI) system called the "double integrator." This simple second-order system is a common example used in optimal control literature [2, 9]. Since the solution for the optimal control policy is easy to calculate, this system provides a basis for understanding and examine the effects of reconfiguration on this system.

3.2.1 Basic Optimal Control Problem

The double integrator system is defined by:

$$\ddot{y}(t) = u(t)$$

where $u(t)$ is the input to the system and $y(t)$ is the position system at time t. Let $y(0)$ and $\dot{y}(0)$ be the initial position and velocity of the system respectively. Given $\zeta_a \leq u(t) \leq \zeta_b$, where $\zeta_a < 0$ and $\zeta_b > 0$ for all t, and initial conditions $y(0)$ and $\dot{y}(0)$, find an optimal control law π^* which moves the system to $y(T) = 0, \dot{y}(T) = 0$ in minimum time.

The solution to this minimum-time problem is well-known: Let $x_1(t) = y(t)$ and $x_2(t) = \dot{y}(t)$. We can define the minimum time to the goal from state $x = (x_1, x_2)$ as:

$$J^*(x) = g_1^*(x, \mu_1^*(x)) + g_2^*(f(x, \mu_1^*(x)), \mu_2^*(f(x, \mu_1^*(x)))) \tag{3.1}$$

where g is the cost function where $g_i^*(x_i, \mu_i^*(x_i)) = t_i^*$, f is the state transition function and μ_i^* is the optimal control at decision i. Next, we can define the current state under control policy $\mu(x_i)$ as a function of time:

$$\begin{aligned} x_1(t) &= x_1(0) + x_2(0)t + \tfrac{1}{2}\zeta_i t^2 \\ x_2(t) &= x_2(0) + \zeta_i t \end{aligned} \tag{3.2}$$

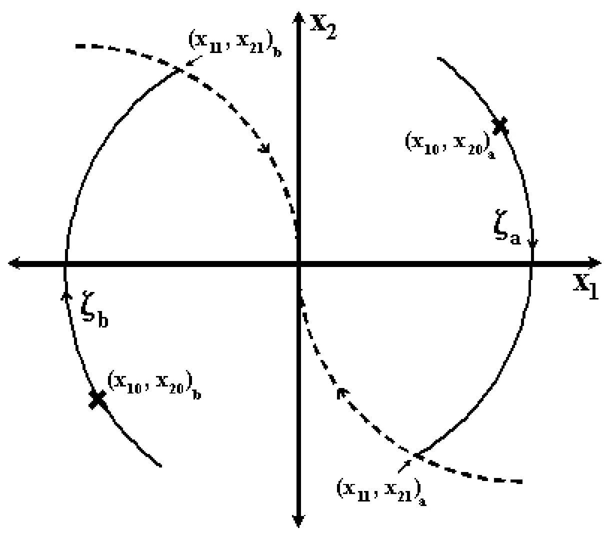


Figure 3-1: System Trajectories and Switching Curve under Optimal Policy for the Double Integrator Problem

And by combining these equations, we get the following equation:

$$x_1(t) - \frac{x_2(t)^2}{2\zeta_i} = x_1(0) - \frac{x_2(0)^2}{2\zeta_i} \tag{3.3}$$

Notice that $x_1(t) - \frac{x_2(t)^2}{2\zeta_i}$ is constant for all t. This equation defines the state trajectories for a the control policy $\mu(x) = \zeta_i$. From these equations we get the following picture in Figure 3-1, which shows the system trajectories and switching curve under this optimal policy.

As shown in Figure 3-1, if the initial state, denoted by $(x_{10}, x_{20})_a$, is above the switching curve (represented by the dashed line), the optimal control policy is to apply $\mu_1^*(x) = \zeta_a$ until the state reaches the switching curve, denoted by $(x_{11}, x_{21})_a$. Then, we apply $\mu_2^*(f(x, \mu_1^*(x))) = \zeta_b$ to reach the goal at $x_f = (0,0)$. Likewise, if the initial state, denoted by $(x_{10}, x_{20})_b$, is below the switching curve, the optimal control policy is to apply $\mu_1^*(x) = \zeta_b$ until the state reaches the switching curve, denoted by $(x_{11}, x_{21})_b$. Then, we apply $\mu_2^*(f(x, \mu_1^*(x))) = \zeta_a$. Finally, if the initial state is on the switching curve, the optimal control policy is to $\mu_2^*(x) = \zeta_i$ (where $i = a$ if $y(0) < 0$ and $i = b$ if $y(0) > 0$) and set $\mu_1^* = 0$ since we are starting on the switching curve. (Note that if $x_0 = (0,0)$, then $\pi^* = \{0, 0\}$ since our goal is to reach the origin).

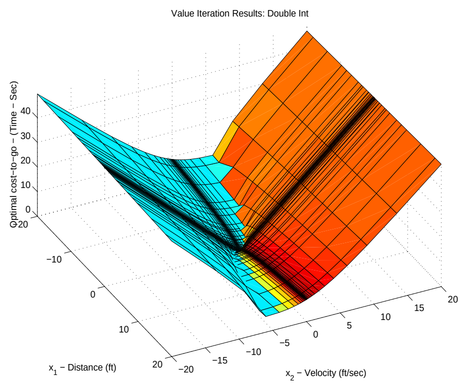


Figure 3-2: Value Function for Double Integrator System

This control policy is optimal and the minimum time trajectory from any initial state can be achieved through its implementation. Given any initial state $x_0 = (x_{10}, x_{20})$, we can calculate the optimal cost $J^*(x)$ by determining the location of $(x_{11}, x_{21})_i$ on the switching curve using Eq. (3.1), Eq. (3.2), and Eq. (3.3). Then, the optimal cost for each control becomes:

$$\begin{aligned} g_1^*(x, \mu_1^*(x)) &= t_1^* = \tfrac{x_{21} - x_{20}}{\zeta_1} \\ g_2^*(f(x, \mu_1^*(x)), \mu_2^*(f(x, \mu_1^*(x)))) &= t_2^* = \tfrac{-x_{21}}{\zeta_2} \end{aligned}$$

where $\pi^* = \{\zeta_1, \zeta_2\}$ is dependent on the initial state x_0 (note that $g_1^* = 0$ if x_0 is on the switching curve). With this information, we can move the system to the goal: first, we apply ζ_1 for t_1^* seconds to move the vehicle to the switching curve and then apply ζ_2 for t_2^* seconds to move the vehicle toward the goal.

In Figure 3-2, we show the nominal value function $J^*(x)$ for the continuous time double integrator system where $\zeta_a = -1$ and $\zeta_b = 1$. In this figure, we show the value function calculated over a grid. One implementation issue we encounter in applying this technique is the need to represent this function in the memory of the system. By adding more points to the value function, the amount of memory space required to store this value function in memory also increases. Therefore, we must select a grid system to adequately represent the shape of the function across the state space (we will discuss this issue in more detail later in this chapter). In this figure the

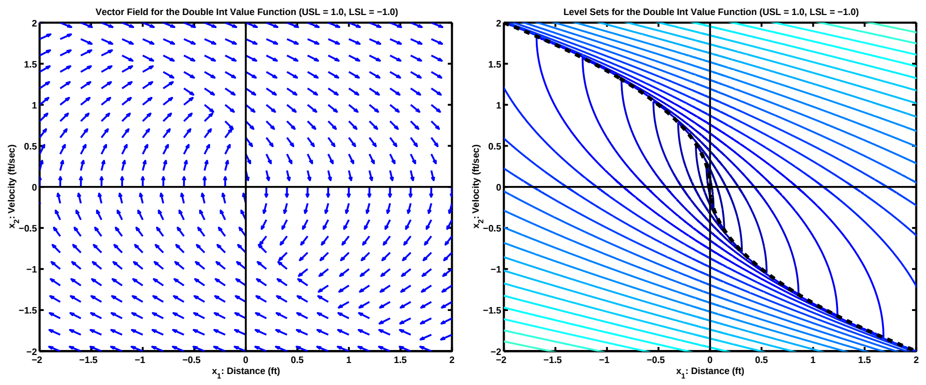


Figure 3-3: Vector Field and Contour Plot for the Value Function for Double Integrator System

parabolic-like "valley" in the switching curve for this system. To better illustrate this switching curve and the contour of this value function, in Figure 3-3 we show the vector field and contour plot for this nominal value function. Notice that the vector field points toward the switching curve. In the contour plot we see how the level sets are shaped around the switching curve, represented by the dashed line.

3.2.2 Value Function Control vs. Linear Feedback Control

Therefore, given any initial condition, we can calculate this optimal control to guide the system to the goal. As the system becomes more complicated, it may take more time to calculate the optimal trajectory. By storing the cost-to-go in the value function for any initial state, we can determine the optimal control for this system by following the gradient of the guidance value function $J^*(x)$ since cost decreases along system trajectories [9]. Therefore, this value function-based guidance system is a feedback system where the controller input is the current state of the system, and the output is the optimal control (based on the cost in the value function) to command the vehicle toward the goal.

To illustrate how the value function can be used as a controller in a feedback system, we have developed the following implementation example of a value function-based guidance system for the double integrator using Simulink, as shown in Fig 3-4. The two major parts of this model are the Trajectory Generation block (the value

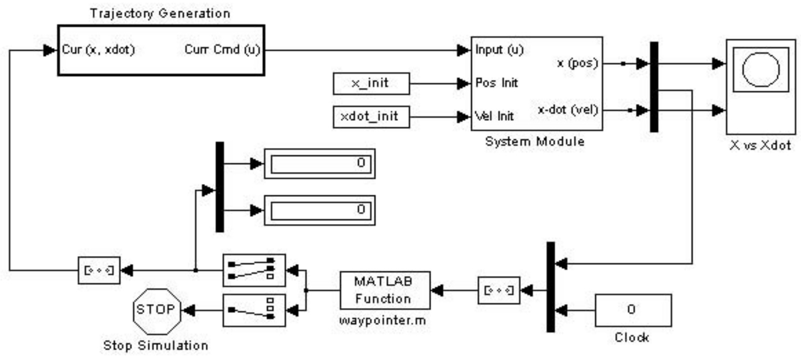


Figure 3-4: Simulink Implementation of Double Integrator Guidance System

function-based controller for this system) and the Plant Module. The remaining components are used to start and stop the simulation.

For this illustration, we selected $x_0 = [x_{init} = 5ft, \dot{x}_{init}3ft/sec]$ as the system's initial conditions. Figure 3-5 shows the path generated by this system. Notice that the system's trajectory resembles the switching curve arcs (shown in Figure 3-1) as the vehicle starts to maneuver toward the goal. Also, we observe that the system changes its trajectory to compensate for its current position as it moves toward the goal. Notice that this curve is not smooth like the switching curve from Figure 3-1. This maneuvering action is caused by the grid spacing in the value function. Since the guidance system interpolates the current cost between grid points, the system may choose to maneuver rather than to maintain the current command because of interpolation errors across the grid. Finally, in Figure 3-5, notice that the vehicle never actually reaches the origin (shown in the figure on the right); instead, the vehicle's state enters into a limit cycle around the origin. Again, this is a result of interpolation errors combined with the simulation fidelity.

Next we can compare the performance of the value function-based guidance system model with a controller for the double integrator system using the LQR method shown

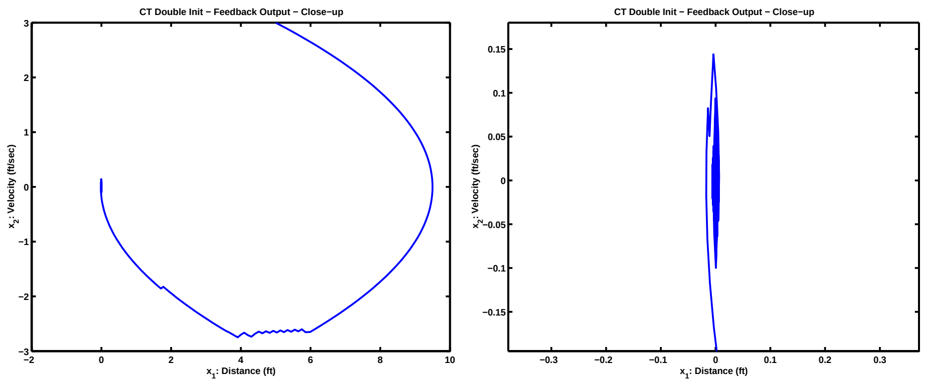


Figure 3-5: Trajectory for $x_0 = [5ft, 3ft/sec]$ using Simulink VF-Based Guidance Double Integrator System

in Chapter 2. The state space model for this system is:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

where $x(0) = [x_{init}, \dot{x}_{init}]^T$ and $|u(t)| \leq 1$. We can set $Q = I_2$ and $R = I_1 = 1$, then we can find K using the Riccati equation:

$$\begin{aligned} KA + A^T K - KBR^{-1}B^T K + Q &= 0 \\ K \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} K - K \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} K + I &= 0 \end{aligned}$$

By solving the equation above, we obtain $K = \begin{bmatrix} \sqrt{3} & 1 \\ 1 & \sqrt{3} \end{bmatrix}$. The optimal control is determined using:

$$\begin{aligned} u^*(t) &= -R^{-1}B^T K x^*(t) \\ &= -\begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{3} & 1 \\ 1 & \sqrt{3} \end{bmatrix} x^*(t) \\ &= -x_1^*(t) - \sqrt{3}x_3^*(t) \end{aligned}$$

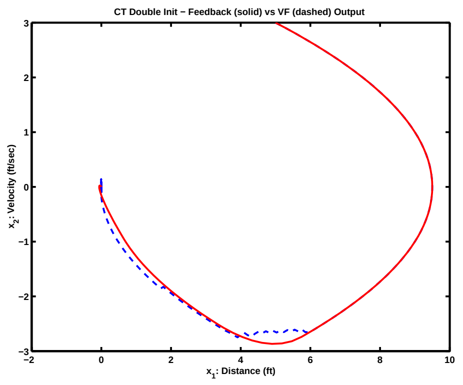


Figure 3-6: Trajectory for $x_0 = [5ft, 3ft/sec]$ using Simulink Feedback-Based (solid) vs. VF-Based (dashed) Guidance Double Integrator System

Next, we substitute this controller in place of the guidance value function and compare the performance of the system. For comparison purposes, we have simulated the output for the same starting conditions ($x_0 = [x_{init} = 5ft, \dot{x}_{init} = 3ft/sec]$).

In Figure 3-6 we see the path generated by this system. In this figure, we notice that as the vehicle starts to maneuver toward the goal, the system's trajectory resembles the switching curve arcs in Figure 3-1. Notice that this system trajectory is very smooth and does not have the same "jagged" characteristics as the state approaches the goal, as we see in Figure 3-5. Again, under the LQR policy the vehicle gets very close, but does not reach the $x_f = [0, 0]$ - just like the value function-based system. Finally, we find that both systems require almost the same amount of time to move within the tolerance limit of $\sqrt{{x_1}^2 + {x_2}^2} \leq 0.1$, despite the extra maneuvering exhibited by the value function-based system for this grid. The Value Function-based system took 9.2 seconds to reach the tolerance region, while the feedback system to 9.9 seconds to reach this region. The main reason the feedback system takes longer is that it uses inputs that range between from $-1 \leq u(t) \leq 1$, while the value function based system uses only the values for zeta described in the basic problem. As a result, when the controller output u in the feedback system tries to switch between $u = -1$ and $u = 1$, it takes on all of the intermediate values, causing the vehicle to transition much slower than in the value-function based model. Therefore, although there are performance differences between the two guidance strategies, both systems use feed-

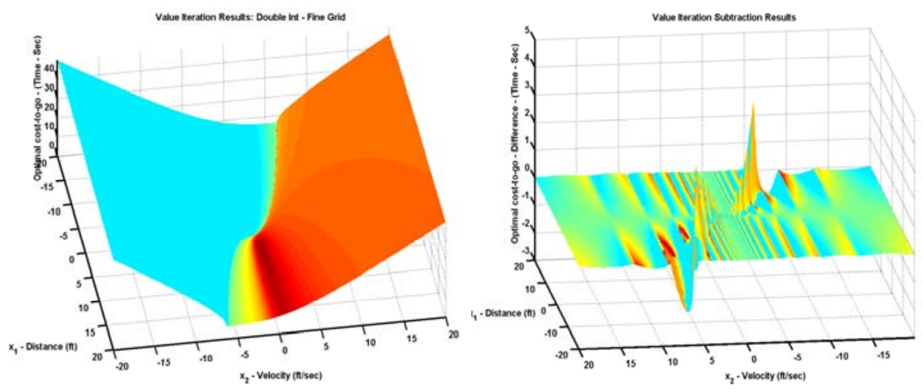


Figure 3-7: Value Function with Finer Grid Size for Double Integrator System

back to direct the vehicle toward the goal by minimizing the error (i.e. minimizing the cost in the value function based system) from the vehicle's current position to the goal state.

3.2.3 System Robustness vs. System Performance

This discussion brings up another key issue: by introducing a strategy that is robust to perturbations in the system, how is system performance affected? To obtain insight, we can use the value function-based guidance system to explore this relationship. We will look at the following two cases: 1) Nominal Case (as used in the previous subsection) and 2) Nominal Case plus additional grid points around the switching curve. Since we have an example of the nominal case from the previous section, we will start by looking at the second function.

In Figure 3-7 we show the value function developed by adding additional grid points near the switching curve on the left and the differences in cost from the value function in Figure 3-2 on the right. By adding more grid points, the interpolated costs between grid points and around the switching curve are more representative of the optimal cost for this system. From the difference function in Figure 3-7, we see that the largest differences in the cost between the two value functions are near the switching curve. In addition, since there are more grid points in this value function, the system requires more storage space to save the value function in memory. (Note:

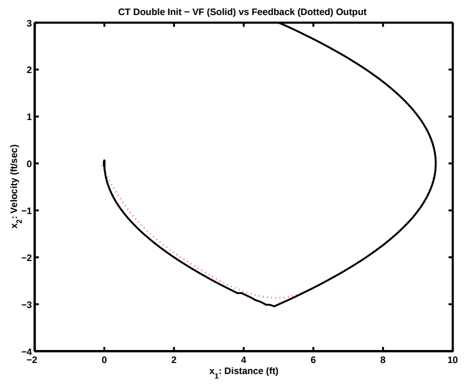


Figure 3-8: Trajectory for $x_0 = [5ft, 3ft/sec]$ using Simulink VF-Based with Fine Grid (solid) vs. Feedback-Based (dashed) Guidance Double Integrator System

since large quantities of memory can be purchased at reasonable prices, the issue of storing large cost functions in residual memory is becoming a smaller concern). Notice that the change in cost over the grid is more gradual around the switching curve than as observed in Figure 3-1.

In Figure 3-8 we show the results from a simulation comparing the performance of the system using the value functions with the coarse- and fine-grid for the initial conditions $x_0 = [x_{init} = 5ft, \dot{x}_{init} = 3ft/sec]$. Notice that the trajectory for the system using the fine-grid value function resembles the path planned using the nominal cost function in Figure 3-5, except that the state trajectory is much "smoother" as the state approaches the goal along the switch curve and resembles the optimal trajectory in Figure 3-4. Finally, we find that the difference in the time to get within our tolerance limit of $\sqrt{{x_1}^2 + {x_2}^2} \leq 0.1$ is 9.1 seconds - which is better than the other two systems for this example.

Therefore, we have shown by example that the performance of the value-function based guidance system can be improved by adding a controlled set of grid points. But, how robust is this system to small perturbations? For example, a perturbation could be caused by a small time delay in the implementation of the next command or changes in the operating environment of the vehicle (i.e. a wind gust). To illustrate this point, we adjusted the simulation to delay in the implementation of the vehicle command as it approaches the switching curve miss the switching curve to simulate

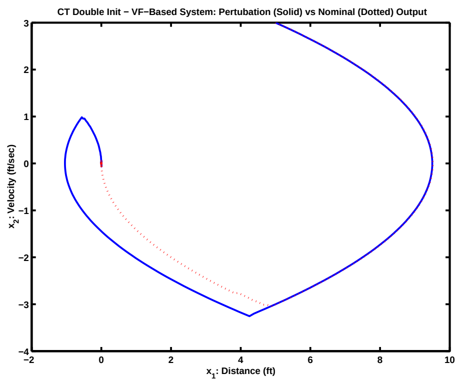


Figure 3-9: Trajectory for $x_0 = [5ft, 3ft/sec]$ using Simulink VF-Based with Fine Grid for Double Integrator Guidance System: Nominal (dotted) vs. Perturbed (solid)

a small perturbation as the system approaches the switching curve. In Figure 3-9, we show the difference between the nominal trajectory for the fine-grid value function (dotted) versus the "perturbed" system trajectory near the switching curve (solid). We observe that the system still converges to the goal even in the presence of this time delay. Note that the system trajectory must reach the other switching curve in order to proceed to the goal.

This example helps to illustrate that though this value function-based system will perform well in a controlled environment, it is robust to perturbations. Note that it is possible that the system may never reach the goal in cases where the system is unable to reduce the current cost-to-go after the application of the optimal control law in the presence of large perturbations. This result is discussed by Branicky in [4] where contraction mapping functions are necessary for the stability of an iterated function system (IFS). In this case, if we are unable to find a control that will reduce the cost along a system trajectory in the value function, then the system will not be stable given this set of control laws.

It is clear that the value function-based guidance system is a form of a feedback system. We can see that there is a relationship between the system's response using both a value function and a linear controller. This relationship helps us to understand how value function-based guidance systems will react to perturbations. In this example we see that if the switching curve was defined over a region, rather than a line, we may increase the robustness of the guidance solution. By adding a "terminal" region, we reduce the performance of the system since the system may switch before or after the switching curve. On the other hand, the terminal region allows for more flexibility in developing a planned trajectory for the vehicle *a priori* using the guidance value function to compensate for errors in the actual trajectory of the vehicle. Since the guidance system performs interpolation to estimate the cost between grid points in the value function, the output trajectory of the guidance system may be constantly adjusted as the system approaches the goal near the switching curve. But, if a feedback was used control the system in this "well-defined" region of the state-space, then we could use a value-function control to get us near the goal region of the vehicle and then switch to feedback control to perform fine adjustments on the system state as it approaches the goal. This topic will be discussed in the next chapter.

3.2.4 Continuous Model vs. Maneuver Automaton

To this point, we have investigated continuous-time system models without hybrid states. Because the double-integrator system is well-defined, we can use this model to compare the maneuver automaton-based guidance system and the continuous time system.

The first question we must address is how the continuous and MA-based guidance models differ. One of the main differences is the introduction of maneuver primitives. This problem is well-addressed by Frazzoli in [9]. For the double integrator system, trim states define the velocities of the system, and the maneuvers connect these trim states. Next, we must decide which trim states to use. From the previous section we found that to increase system performance and reduce interpolation errors, we need to define more velocity states near the goal. Therefore, as proved in [9], we can choose a logarithmic distribution of the trim states (with maneuvers connecting them) so that the vehicle can make fine adjustments to the system as it approaches the goal.

Next, we must understand how the value functions for the continuous time and the

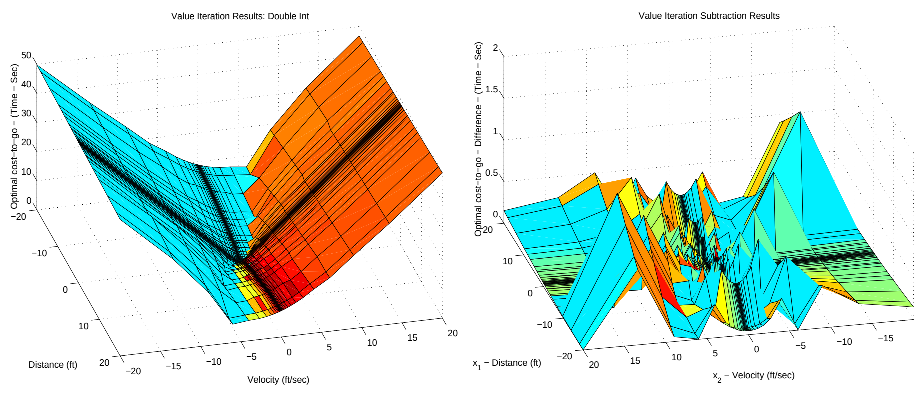


Figure 3-10: MA-Based Value Function for Double Integrator System

MA-based systems differ. To illustrate this example, we developed a MA with trims based on the location of the velocity grid lines shown in Figure 3-2 with maneuvers connecting all of the trim states. In Figure 3-10 we show the MA-based value function and the differences in cost from the value function for the continuous system in Figure 3-2. At first glance, it appears like there are no differences between the MA-based and the continuous time value function; however, by examining the difference function graph, we notice that the MA-based value function has a "higher" cost over most of the cost function grid. The reason for this change in cost has to do with the "optimality gap" between the systems. Unlike the continuous value function, the MA-based system can only operate at trim states. Therefore, if a trim state is not defined for given vehicle state, then the MA-based system does not have a cost for this point. For this reason, the value function shown in Figure 3-10 is deceiving. The value function for the MA-Based system can only be evaluated along the velocity grid lines, and therefore the MA-based value function undefined the velocity grid lines. Figure 3-11 helps to illustrate this difference between the continuous model and the MA-based model using a sample trajectory. In this figure, the vehicle starts at the initial hybrid state $[x_{10}, q_0]$. Next, the vehicle follows a similar trajectory as we see in Figure 3-1 using maneuver $p_1(\zeta_b)$ to move the vehicle toward the switching curve. But, since maneuvers begin and end at trim states, the vehicle is unable to reach the switching curve under this maneuver. If a trim state is not defined at the point where the vehicle would intersect the switching curve under the continuous model, the vehicle

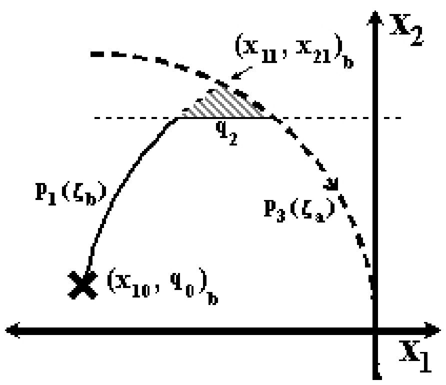


Figure 3-11: System Trajectories under Optimal Policy for MA-Based Double Integrator Problem

will stop at the closest trim state to the switching curve. Then, the vehicle will coast on the trim state q_2 until it reaches the switching curve, at which point the system will maneuver to the goal state using $p_3(\zeta_a)$. This "optimality gap", represented by the shaded region in Figure 3-11, results from the fact that the system may not be able to maneuver directly to the switching curve - resulting in an additional cost t_{q_2} from coasting.

From this example, we see that the optimal cost-to-go from any initial state $x_{q0} = [x_0, q_0]$ for the MA-based system can be found using:

$$J^*(x_{q0}) = g_1^*(x_{q0}, p_1(x_{q0})) + g_2^*(x_{p_1}, \tau_{q_2}(x_{p_1})) + g_3^*(x_{q_2}, p_3(x_{q_2})) \tag{3.4}$$

where g is the cost function where $g_i^*(x_i, \mu_i^*(x_i)) = t_i^*$, and μ_i^* is the optimal control at decision i. Notice that the MA-based system may lose some performance by adding a cost term for coasting because of the quantization of the state space; however, we gain the ability to quickly calculate the optimal trajectory for the vehicle (as we will see later in this chapter).

3.2.5 Reconfiguration of the MA-Based System

Up to now, the properties of these systems have been based on a nominal model of the system. But, what happens if this system changes? For this simple double

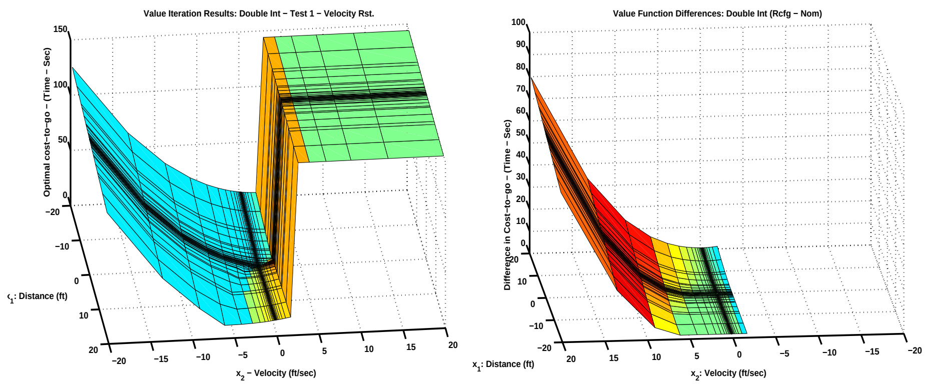


Figure 3-12: Recalculated Continuous Value Function for Test 1

integrator system, we can calculate a new value function for the coarse grid in less than five seconds; however, it may take hours or even days to calculate a multi-dimensional value function. For example, the value function for a 3-Trim lateral-directional axis MA (which we will discuss later in this chapter) can take up to three days to calculate with a fine grid. Therefore, we would like to know if we can use the nominal value function to drive the reconfigured system. First, we explore the effects of reconfiguration on the continuous double integrator system. We examine the effects of restricting the velocity and changing the output command limits for the system. Next, we investigate the reconfiguration of the MA-based double integrator system. We explore the effects of reconfiguration on the system after we remove and add maneuver primitives. Finally, we look at the performance of the reconfigured MA-based system when the command limits on the maneuvers change.

Continuous Value Function - Test 1: Velocity Restrictions

In this first test, we want to examine what happens to the system trajectory when we restrict the operating velocities of the system. For this test we will assume that the system is unable to reach velocities less than -2 ft/sec. In Figure 3-12, we see the recalculated cost function for the reconfigured system and the differences in cost from the nominal value function in Figure 3-2. Notice that the cost is set to ∞ for velocities below -2 ft/sec.

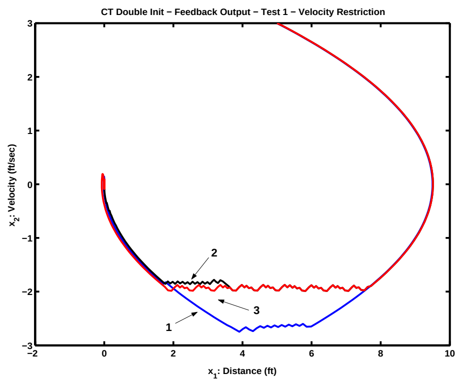


Figure 3-13: Reconfigured Continuous Double Integrator System Test 1: Trajectory for $x_0 = [5ft, 3ft/sec]$

Traj	Name	Time (sec)
1	Nominal (Blue)	9.19
2	Reconfigured (Black)	9.93
3	Recalculated (Red)	9.83

Table 3.1: Time Comparison for Rcfg Continuous Double Integrator Test 1

Next, to investigate the difference between the nominal, reconfigured and recalculated strategies for this condition we simulated trajectories for the initial condition $x_0 = [5ft, 3ft/sec]$. The actual system trajectories are shown in Figure 3-13 and their projected times are compared in Table 4.1.

From this test we see that all three strategies calculate trajectories which guide the system toward the goal. Second, all three strategies follow almost the same trajectory when they reach the switching curve. Notice that there is a slight difference between the reconfigured and recalculated strategies for this test case as the vehicle approaches the switching curve. Finally, the reconfigured strategy using the nominal cost function directs the vehicle to the goal state, despite the fact that the cost function is not optimal for the reconfigured system.

Continuous Value Function - Test 2: Changes in Command Limits

In this second test, we want to examine what happens to the system trajectory when we change the output command limits from the controller. For this test we will

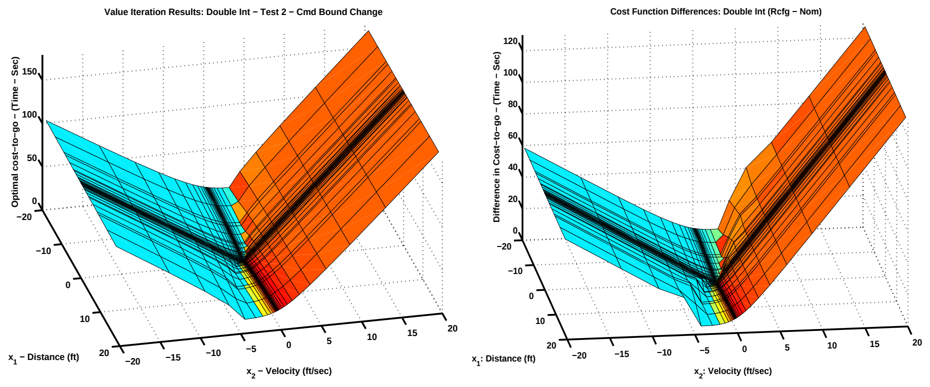


Figure 3-14: Recalculated Continuous Value Function for Test 2

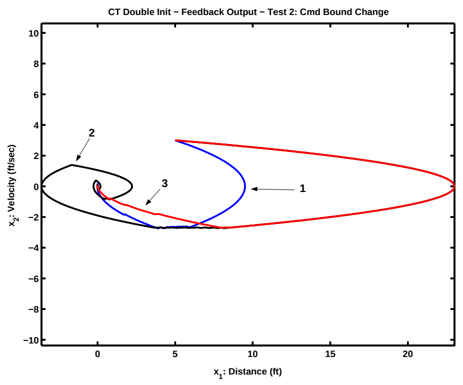


Figure 3-15: Reconfigured Continuous Double Integrator System Test 2: Trajectory for $x_0 = [5ft, 3ft/sec]$

assume that the system has lost some control authority and the command limits have been reduced to $\zeta_a = -0.25$ and $\zeta_b = 0.5$. In Figure 3-14, we see the recalculated cost function for the reconfigured system and the differences in cost from the nominal value function in Figure 3-2. First, notice that there is a change in the location of the switching curve. Since the command bounds have been decreased, the switching curves are closer to the x_1-axis than in Figure 3-2.

Next, to investigate the difference between the nominal, reconfigured and recalculated strategies for this test condition, we simulated trajectories for the initial condition $x_0 = [5ft, 3ft/sec]$. The actual system trajectories are shown in Figure 3-15 and their projected times are compared in Table 3.2.

From this test we see that all three strategies calculate trajectories which guide

Traj	Name	Time (sec)
1	Nominal (Blue)	9.19
2	Reconfigured (Black)	47.31
3	Recalculated (Red)	28.83

Table 3.2: Time Comparison for Rcfg Continuous Double Integrator Test 2

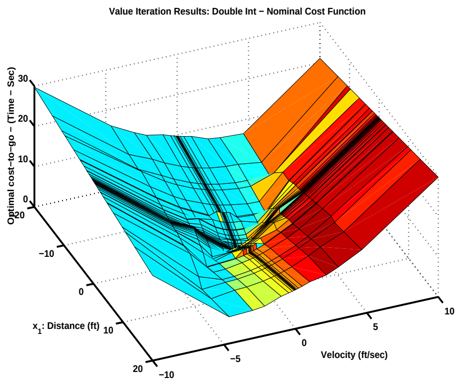


Figure 3-16: Nominal MA-Based Value Function for Double Integrator System Reconfiguration Tests

the system toward the goal. Although the reconfigured strategy takes almost twice as long as the recalculated strategy to guide the system to the goal, the strategy produces a stable and feasible solution. Notice that the reconfigured strategy guides the vehicle according to the location of the switching curve for the nominal system. When the system reaches this point, the reconfigured strategy employs the opposite control. Since this reconfigured system has less control authority than the nominal system, the vehicle overshoots the goal state. Eventually, the vehicle reaches the goal - showing that the reconfigured strategy can be used for this test case.

MA-Based Value Function Tests

For all of the Maneuver Automaton-based Double Integrator system examples, we have defined the nominal system as detailed in Table A.1 listed in Appendix A. Using this nominal MA, the value function for this system is shown in Figure 3-16. To compare the effects of reconfiguration on this system with the previous examples in this section, we have added a trim with a velocity at 3 ft/sec, in order to use the

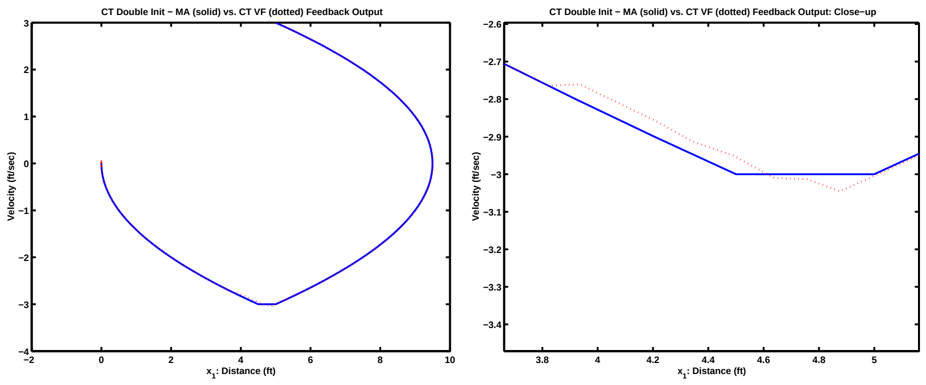


Figure 3-17: Trajectory for $x_0 = [5ft, 3ft/sec]$ using Simulink CT VF-Based Fine Grid (dotted) vs. Nominal MA-based System (solid)

same starting condition used in the previous examples. In Figure 3-17, we show the difference between the nominal trajectory using the fine grid value-function based system (dotted) and the nominal MA-based system trajectory (solid). From this figure, we see that the MA-based system will coast for a short period of time, while the continuous time value function-based system will constantly maneuver - resulting in a small difference in the cost.

MA-Based Value Function Test 1: Trim Removal

In this first test, we examine what happens to the system trajectory when we remove trim states from the MA. We use the reconfigured MA as listed in Table A.2 in Appendix A.

In this test we are simulating a change in the minimum velocity the system can achieve. This test will show how the nominal value function can be used with the reconfigured MA when Trim 3 is removed. For this test, we selected an initial condition where the planned trajectory coasts on Trim 3 ($v = -1$ ft/sec) for 0.2 seconds under the nominal policy. In Figure 3-18, we see the recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-16. First, notice that the cost is set to ∞ in areas where a trim does not exist. Second, upon close inspection of the difference function (on the right), notice that there is an area which has no change in the cost. Therefore, despite changes in

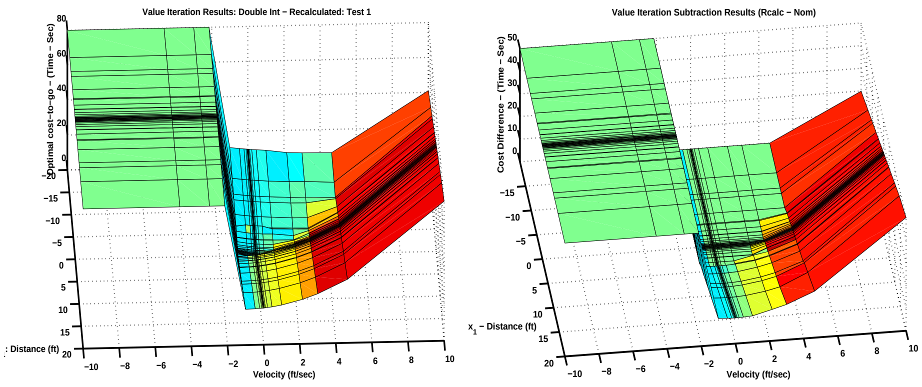


Figure 3-18: Recalculated MA-Based Value Function for Test 1

the MA, there are regions of the value function which are unaffected because they do not make "use" of the unavailable trim states in their optimal path. The existence of this region can be explained by the Principle of Optimality [2]:

Theorem 3.1 (Principle of Optimality) *Let* $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ *be an optimal policy for* $J^*(x) = \min_{u\in\mathcal{U}} \mathbf{E}_w \{g(x,u,t) + J^*(f(x,u,w))\}$ *and assume that when* π^**, a given state* x_i *occurs at time* i *with positive probability. Consider the subproblem where by we are at* x_i *at time* i *and from time* i *to time* N *we wish to minimize* $\mathbf{E}_w \left\{ g_N(x_N) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\}$*, then the truncated policy* $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ *is optimal for the subproblem.*

Optimal trajectories moving from "undefined" regions of the recalculated cost function must approach the switching curve through regions where the cost has not changed. As a result, trajectories starting from the "unchanged" regions of the value function are the sub-trajectories for system trajectories beginning in the "undefined" regions by the Principle of Optimality.

Next, to investigate the difference between the nominal, reconfigured and recalculated strategies for this failure condition we simulated trajectories for the initial condition $x = [x_0 = 5ft, q = 21(q_{vel} = 3ft/sec)]$. The actual system trajectories are shown in Figure 3-19 and their projected times are compared in Table 3.3.

From this test we see that all three strategies calculate trajectories which guide the system toward the goal. Second, all three strategies follow the same trajectory

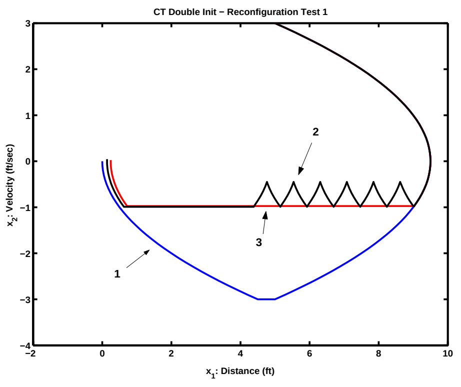


Figure 3-19: Reconfigured MA for Double Integrator Test 1: Trajectory for $[x_0 = 5ft, q = 21(q_{vel} = 3ft/sec)]$

Traj	Name	Time (sec)
1	Nominal (Blue)	9.1667
2	Reconfigured (Black)	15.2228
3	Recalculated (Red)	13.5000

Table 3.3: Time Comparison for Reconfiguration Test 1

when they reach the switching curve, despite the fact that they do not end in the same place. In Figure 3-19 notice that the reconfigured vehicle trajectory initially switches back and forth between two trims states. Using the value function to calculate the vehicle's next decision, the planner selects the next maneuver primitive by looking for the best cost using the maneuver primitives available from the current state. To find the optimal solution the planner must look for the trajectory which minimizes the cost across the value function for each state. The problem is that when we change the MA, the planner will find the minimum cost path to the next vehicle state using the restricted set of maneuver primitives over the nominal value function - though it may not be the best decision under the reconfigured MA. As a result, the planner selects the maneuver primitive which minimizes the cost at each decision, rather than over the trajectory - resulting in this switching behavior. Next, notice that the final vehicle states of the three trajectories does not necessarily correspond with the desired final state $x_f = [0, 0]$. Since we have not implemented any feedback around the trim state velocities, errors resulting from maneuver and simulation tolerance can cause the

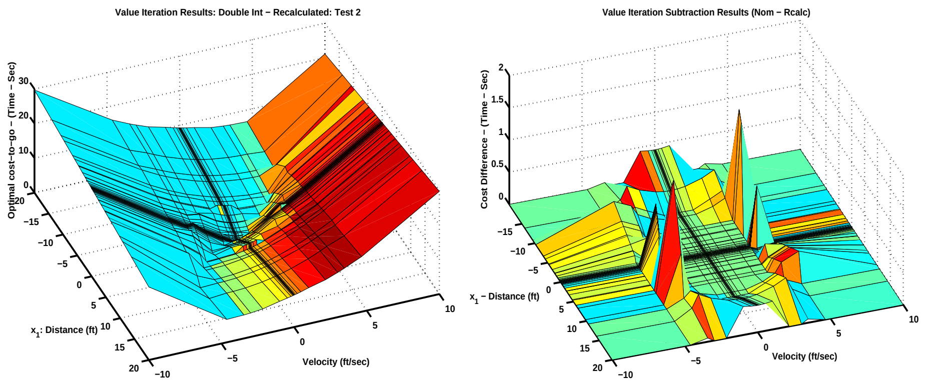


Figure 3-20: Recalculated MA-Based Value Function for Test 2

vehicle to miss the desired destination. For this reason, a feedback or error correcting loop around the vehicle state could be used to ensure the vehicle reaches the desired final state. Finally, the reconfigured strategy using the nominal cost function directs the vehicle to the goal state, despite the fact that the cost function is not optimal for the reconfigured system (similar to the results from the continuous model test shown earlier).

MA-Based Value Function Test 2: Trim Addition

In this second test, we examine what happens to the system trajectory when we add trims to the MA. For most systems, it is difficult to add a trim condition to the value function. Since this system is simple, we can interpolate the cost between grid points on the value function to add a new trim for the nominal system. In Matlab, this interpolation can be done using the *interp2* by defining the new grid for the nominal value function. This technique may not work with other systems, but we can use it with this example to understand this process. We have added two trims at ± 3.45 ft/sec. The reconfigured MA for this system is listed in Table A.3 in the appendix.

In Figure 3-20, we see the recalculated value function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-16. Again, there are areas in this value function which are unaffected by adding these trim states. Moreover, upon close inspection of the difference function (on the right), notice that

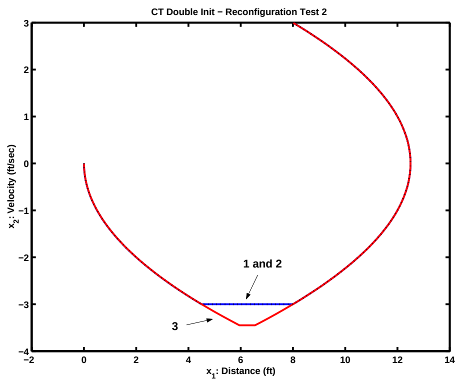


Figure 3-21: Reconfigured MA for Double Integrator Test 2: Trajectory for $[x_0 = 8ft, q_{vel} = 3ft/sec]$

Traj	Name	Time (sec)
1	Nominal (Blue)	10.1667
2	Reconfigured (Black)	10.1667
3	Recalculated (Red)	10.0732

Table 3.4: Time Comparison for Reconfiguration Test 2

there are sharp changes in cost around the switching curve. Some of these changes, resulting from interpolation errors, show that the nominal cost function has a lower cost than the reconfigured value function at the new trim states. These errors occur because we are interpolating to estimate the values of the reconfigured value function (using the nominal value function) at points where a trim state did not exist when the original cost function was calculated. For this reason, it may be very difficult to use this technique on value functions which are more complex since the interpolated reconfigured cost from the nominal value function may not be representative of the actual cost.

Next, to investigate the difference between the nominal, reconfigured and recalculated strategies for this test condition we have calculated trajectories using each system for the initial condition $x_{init} = [x_0 = 8ft, q = 21(q_{vel} = 3ft/sec)]$. The actual system trajectories are shown in Figure 3-19 and their projected times are compared in Table 3.4.

Again, we observe that all three strategies calculate trajectories which guide the system toward the goal. Notice that by adding performance to the system, the recalculated trajectory is "faster" than the nominal trajectory. Also, note that the reconfigured strategy directs the vehicle to the goal state, but it does not use the additional trim state to reach the goal. Because we used the interpolation function over the nominal value function to calculate the costs for the additional trim states, the cost between the original grid points does not change in slope to approximate the actual cost new trim points when using the new MA. As a result, the interpolated cost for the new trim states was not an improvement over the nominal cost for the original trajectory. Therefore, though the reconfigured strategy calculated a feasible trajectory to the goal, it did not use the trim that provided more performance.

From these two tests, we see that when we take away controls, there are some regions of the value function that are unaffected by the change in controls. Likewise, when we add controls, the nominal value function becomes an upper bound for the optimal reconfigured strategy. We can formulate these ideas into the following lemma:

Lemma 3.1 *Let $\mathcal{U}_{nom}$ be the set of all connected controls where $\mu_k \in \mathcal{U}_{nom}$ for the nominal system and let $J^*_{nom}(x)$ satisfy Eq. (2.1) $\forall$ $\mu_k \in \mathcal{U}_{nom}$ and $x \in \mathcal{X}$. Then, the following condition exists:*

- *For any $\mathcal{U}_{nom} \subset \mathcal{U}_{rcfg}$ where $\mathcal{U}_{rcfg}$ is connected, then $J^*_{nom}(x) \geq J^*_{rcfg}(x)$*
- *For any $\mathcal{U}_{rcfg} \subset \mathcal{U}_{nom}$ where $\mathcal{U}_{rcfg}$ is connected, then $\forall_{x\in\mathcal{X}}$ where $J^*_{nom}(x)$ is found using $\{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\}$ and the set of controls $\{\mu_k\} \subset \mathcal{U}_{rcfg}$, then $J^*_{nom}(x) = J^*_{rcfg}(x)$*

Proof (by Contradiction):

- Assume not. Then, for any $\mathcal{U}_{nom} \subset \mathcal{U}_{rcfg}$ where $\mathcal{U}_{rcfg}$ is connected, $\exists_{y\in\mathcal{X}}$ where $J^*_{nom}(y) < J^*_{rcfg}(y)$. Then, $\exists$ π^* where the set of all controls $\{\mu_k\}_{\pi^*} \in \mathcal{U}_{nom}$ such that $J^*_{nom}(y) < J^*_{rcfg}(y)$. But, if $\{\mu_k\}_{\pi^*} \in \mathcal{U}_{nom}$, then $\{\mu_k\}_{\pi^*} \in \mathcal{U}_{rcfg}$, which means that $J^*_{rcfg}(y)$ does not satisfy Eq. (2.1), and is not optimal. This is a contradiction since both $J^*_{nom}(y)$ and $J^*_{rcfg}(y)$ are optimal under $\mathcal{U}_{nom}$ and $\mathcal{U}_{rcfg}$ respectively. Therefore, $J^*_{nom}(x) \geq J^*_{rcfg}(x)$. $\square$

- ($\Rightarrow$) To show $J^*_{nom}(x) \geq J^*_{rcfg}(x)$: assume not. Then, for $\mathcal{U}_{rcfg} \subset \mathcal{U}_{nom}$ where $\mathcal{U}_{rcfg}$ is connected, $\exists_{y \in \mathcal{X}}$ where $J^*_{nom}(y) < J^*_{rcfg}(y)$. Then, $\exists\ \pi^*_{y_{nom}}$ where $\{\mu_k\}_{\pi^*_{y_{nom}}} \in \mathcal{U}_{rcfg} \subset \mathcal{U}_{nom}$. But, since both $J^*_{nom}(y)$ and $J^*_{rcfg}(y)$ are optimal, then $J^*_{rcfg}(y)$ must use not use $\pi^*_{y_{nom}}$, which violates the optimality principle. This is a contradiction since $J^*_{rcfg}(y)$ under $\mathcal{U}_{rcfg}$ is the optimal cost under Eq. (2.1) and therefore must use $\pi^*_{y_{nom}}$ where $\{\mu_k\}_{\pi^*_{y_{nom}}} \in \mathcal{U}_{rcfg} \subset \mathcal{U}_{nom}$.

  ($\Leftarrow$) To show $J^*_{rcfg}(x) \geq J^*_{nom}(x)$, we can apply the statement under the first bullet in this lemma and replace $\mathcal{U}_{rcfg} \rightarrow \mathcal{U}_{nom}$ and $\mathcal{U}_{nom} \rightarrow \mathcal{U}_{rcfg}$. $\square$

This lemma provides us with insight into a favorable reconfiguration strategy for complex systems. For systems with degraded performance, if we can find the regions of the value function which are unaffected by changes in the MA, then we can try to find a sequence of maneuver primitives that will move the vehicle to these regions of the value function. Since this region is unaffected by the change in the MA, it can be used to find a guidance solution to the goal. Likewise, if we add performance to the system, we recognize that in order for the vehicle to select these new maneuver primitives, they must provide large gains in cost. Otherwise, they will not be selected over the maneuver primitives in the nominal system (we will see an example of this case in the next section).

MA-Based Value Function Test 3: Changes in Maneuver Command Limits

Up to this point we have assumed that the maneuver command limits are symmetric and have remained unchanged ($\zeta_a = -1$, $\zeta_b = 1$). In this third test we investigate how changes in command limits effect the VF-based guidance system. In this section we set $\zeta_a = -\frac{1}{2}$ and $\zeta_a = 1$. Notice that this system has less performance since we have raised the lower saturation limit of the system. The reconfigured MA for this test is listed in Table A.4 in the appendix.

In Figure 3-22, we see the recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-16. In contrast to the other two MA-based system examples, it appears that the entire cost function is

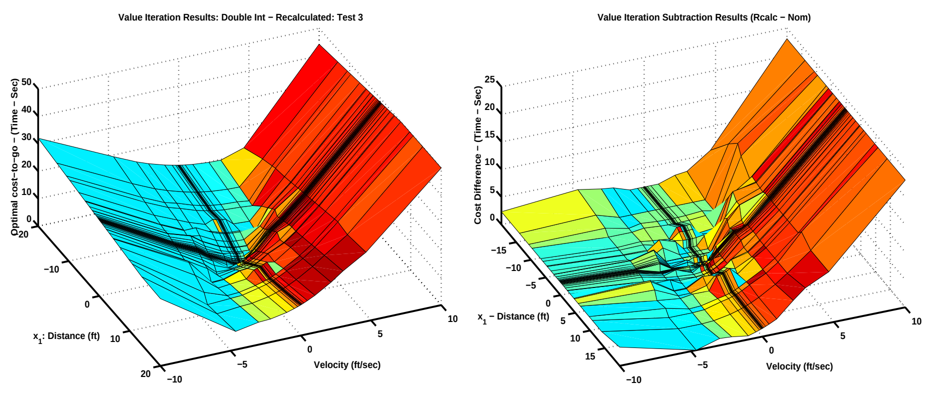


Figure 3-22: Recalculated MA-Based Value Function for Test 3

Traj	Name	Time (sec)
1	Nominal (Blue)	3.5000
2	Reconfigured (Black)	83.0845
3	Recalculated (Red)	5.5000

Table 3.5: Time Comparison for Reconfiguration Test 3

affected by this change. In almost every case (except points along the lower switching curve), the vehicle must use a maneuver affected by this change to reach the goal. Second, notice that the switching curve is not "well-defined" in the upper right quadrant of the value function in Figure 3-22, as compared to the nominal value function in Figure 3-16. Since we have defined the grid based on the symmetric, unitary command limits, the grid points may not be aligned with the revised switching curve. Thus, the grid of the recalculated value function is not optimized for the reconfigured MA.

To understand the differences between the three reconfiguration cases for this test condition, we have calculated trajectories using each system for the initial condition $x = [x_0 = 1ft, q = 19(q_{vel} = 1ft/sec)]$. The actual system trajectories are shown in Figure 3-23 and their projected times are compared in Table 3.5.

Again, we see that all three strategies calculate trajectories which guide the system toward the goal. Although the reconfigured strategy that takes sixteen times longer than the recalculated strategy, it calculates a feasible trajectory and stabilizes the system. The reason for this large difference is related to the strategy used to find

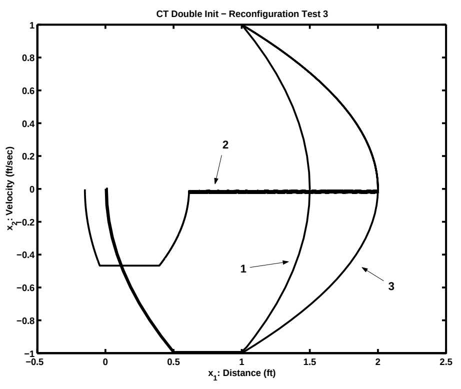


Figure 3-23: Reconfigured MA for Double Integrator Test 3: Trajectory for $[x_0 = 1ft, q = 19(q_{vel} = 1ft/sec)]$

the optimal path. The planner uses the value function to calculate the vehicle's next decision by looking for the best cost based on the maneuver primitives available from the current state. To find the optimal solution the planner must look for the trajectory which minimizes the cost across the value function for each state. The problem is that when we change all the maneuvers, the planner will find the minimum cost path to the next vehicle state using the new set of maneuvers over the nominal value function - though it may not be the best decision under the reconfigured MA. As a result, this decision process causes the planner to switch between trim states 10 (-.01 ft/sec) and 11 (-.0215 ft/sec) for over 60 seconds. Next, notice that the actual vehicle trajectory under the reconfigured strategy misses the goal. The drift in the vehicle trajectory is a result of simulation precision during the maneuver transitions between trim states 10 and 11. This drift is not a serious issue since the planner could be designed to update the trajectory based on the system's state after a period of time, thus reducing any position errors in the vehicle trajectory.

This test points out some of the difficulties encountered with the reconfigured strategy under the current guidance law. Though this strategy provides us with a feasible trajectory, we would like the planner to use the nominal cost function to calculate a trajectory to the goal in a *reasonable* period of time with respect to the recalculated trajectory. To better understand how we can develop a strategy to improve the reconfiguration strategy, we can start by looking at the following lemma.

Lemma 3.2 *Let $\mathcal{U}_{nom}$ be the set of all connected controls where $\mu_k \in \mathcal{U}_{nom}$ for the nominal system and let $J^*_{nom}(x)$ satisfy Eq. (2.1) $\forall\ \mu_k \in \mathcal{U}_{nom}$ and $x \in \mathcal{X}$. Define $C \subset \mathcal{X}$ as the set where $\forall\ x \in C \subset \mathcal{X}$, $J^*_{nom}(x) = J^*_{rcfg}(x) < \infty$. For any $y \in \mathcal{X}$, if there exists a sequence of controls $\pi_y = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\}$ where the set of controls $\{\mu_k\} \subset \mathcal{U}_{rcfg}$ and $f(y, \pi(y)) \in C$, then $J^*_{rcfg}(y) < \infty$.*

This lemma says that given an initial state and a sequence of controls that moves the system into a region in the recalculated value function that is unaffected by the change to the MA (i.e. has the same cost as the nominal value function), then a feasible trajectory for the reconfiguration strategy exists. The proof is very simple and uses the connectivity condition we impose on the reconfigured MA.

Proof: Let $x = f(y, \pi(y)) \in C$ where $J^*_{rcfg}(x) < \infty$. We can write $J_{rcfg}(y) = h(y, \mu(y)) + J^*(x)$ where we can define $h(y, \pi(y)) = \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k))$ which is the current cost under control policy π_y. By the connectivity condition imposed on $\mathcal{U}_{rcfg}$, $h(y, \pi(y)) < \infty$, so $J_{rcfg}(y) < \infty$. Finally, using Eq. (2.1), we find that $J^*_{rcfg}(y) < \infty$. $\square$

The next question we can ask is related to the situation we have in this third test case. Here, the entire cost function changed because of the change in the maneuver bounds. As a result, there may not be a well-defined region in the cost function which has not changed. Therefore, if we can develop a strategy that defines an "achievable" region in the value function based on the current MA, then we may be able to use the value function to find a suitable trajectory under the reconfiguration strategy.

Lemma 3.3 *Let $\mathcal{U}_{nom}$ be the set of all connected controls where $\mu_k \in \mathcal{U}_{nom}$ for the nominal system and let $J^*_{nom}(x)$ satisfy Eq. (2.1) $\forall\ \mu_k \in \mathcal{U}_{nom}$ and $x \in \mathcal{X}$. Define $C \subset \mathcal{X}$ as the set where $\forall\ x \in C \subset \mathcal{X}$, $\exists\ \mu(x) \in \mathcal{U}_{rcfg} \subset \mathcal{U}_{nom}$ such that $J^*_{rcfg}(x) = g(x, \mu(x)) < \infty$. For any $y \in \mathcal{X}$, if there exists a sequence of controls $\pi_y = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\}$ where the set of controls $\{\mu_k\} \subset \mathcal{U}_{rcfg}$ and $f(y, \pi(y)) \in C$, then $J^*_{rcfg}(y) < \infty$.*

This lemma says that given an initial state and an "achievable" region in the value function, defined as a set of states by which the goal state can be reached under a control in the reconfigured MA. If a sequence of controls exists that moves the system into the "achievable" region, then a feasible trajectory for the reconfiguration strategy exists. The proof is identical to the proof in Lemma (3.2) and uses the connectivity condition we impose on the reconfigured MA.

Proof: Let $x = f(y, \pi(y)) \in C$ where $J^*_{rcfg}(x) < \infty$. We can write $J_{rcfg}(y) = h(y, \mu(y)) + J^*(x)$ where we can define $h(y, \pi(y)) = \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k))$ which is the current cost under control policy π_y. By the connectivity condition imposed on $\mathcal{U}_{rcfg}$, $h(y, \pi(y)) < \infty$, so $J_{rcfg}(y) < \infty$. Finally, using Eq. (2.1), we find that $J^*_{rcfg}(y) < \infty$. $\square$

Proposed Improvement to the Reconfiguration Strategy

Using this information we can try to improve the reconfiguration strategy. We proposed using the following process: First, use the nominal guidance value function to find trajectories that will guide the system to a region of the value function with a lower cost than the initial vehicle state. Then, select the guidance solution that has the smallest estimated cost-to-go from the next state to the "achievable" region described in the previous subsection. By looking ahead, we can minimize the cost of the trajectory rather than over each decision. In Figure 3-24, we show the actual system trajectories for the three reconfiguration cases with initial condition $x = [x_0 = 1ft, q = 19(q_{vel} = 1ft/sec)]$ in Test 3 using a "look ahead" strategy. This strategy is designed to, first, find all of the trajectories that move the system to a region of lower cost in the value function. Then, for each of these trim states, the system estimates how long it will take the system to reach the achievable destination region from each trim state. Finally, the system selects the trim with the best overall cost. The projected times for the nominal, recalculated and reconfigured with the "look ahead" strategy are compared in Table 3.6.

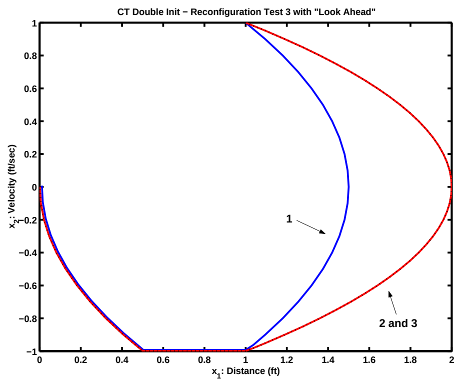


Figure 3-24: Reconfigured MA with "Look Ahead" Strategy for Double Integrator Test 3: Trajectory for $[x_0 = 1ft, q = 19(q_{vel} = 1ft/sec)]$

Traj	Name	Time (sec)
1	Nominal (Blue)	3.5000
2	Reconfigured (Black)	5.5000
3	Recalculated (Red)	5.5000

Table 3.6: Time Comparison for Reconfiguration Test 3 with "Look Ahead" Strategy

From this figure, we see that the "look ahead" strategy enhances the performance of the reconfiguration strategy. In fact in this example, the recalculation strategy and reconfiguration strategy produce the same trajectory.

3.2.6 Double Integrator Case Study Test Summary

From these tests we have shown that the nominal value function can be used for trajectory generation for a reconfigured system (both continuous and a maneuver automaton). Using these test cases we showed how changes in the system affect the shape of the value function and trajectories under each strategies. We observed that the reconfiguration strategy can provide an "achievable" path for the reconfigured MA. However, these paths may not be "reasonable" for use. We proposed an improvement to the reconfiguration strategy and found that if the reconfiguration strategy is designed to search for a region in the nominal value function that connects to the goal, then we can find an "achievable" path for the reconfigured system. By using a "look-ahead" strategy, we can use the nominal value function to calculate trajectories that move the vehicle toward the goal and provide the best cost over the length of the trajectory.

3.3 Case Study II: Longitudinal MA Control

The second system we will investigate is a simplified closed-loop model of a longitudinal axis Fixed-Wing UAV maneuver automaton. As posed in [19], we have set the following requirements: 1) minimize the travel time between waypoints, 2) provide a maximum acceleration limit to bound body forces during a maneuver, and 3) optimize fuel consumption for each maneuver. Given our performance constraints and the linear plant model of the system, we used Linear Programming to develop optimal maneuvers for the transitions between states. Each maneuver is characterized by a start and end trim state, an execution time and a displacement in the x- and z-coordinate.

3.3.1 Maneuver Design for Longitudinal MA Tests

As presented in [19], to generate the these maneuvers, we represent the vehicle dynamics by a discretized state space model $(\mathbf{A}, \mathbf{B})$. The state variables in our model are the velocity v, the acceleration $\dot{v}$, the heading γ and the change in heading per unit time $\dot{\gamma}$. The input variables are the reference velocity v_{ref} and the reference heading γ_{ref}. Given minimum and maximum constraints on all state and input variables, the problem of finding a maneuver that takes N discrete time steps and minimizes a total cost J, can be formulated as follows:

$$\begin{aligned}
\min J = \sum_{k=0}^{N} f_k(\mathbf{x}_k, \mathbf{u}_k) & & & \\
\text{subject to} \quad \mathbf{x}_{k+1} &=& \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k & \\
\mathbf{x}_0 &=& \mathbf{x}_{from} & \\
\mathbf{x}_N &=& \mathbf{x}_{to} & \qquad (3.5)\\
\mathbf{x}_k &\geq& \mathbf{x}_{min} & \\
\mathbf{x}_k &\leq& \mathbf{x}_{max} & \\
\mathbf{u}_k &\geq& \mathbf{u}_{min} & \\
\mathbf{u}_k &\leq& \mathbf{u}_{max} &
\end{aligned}$$

Here, $\mathbf{x}_k$ and $\mathbf{u}_k$ are respectively the state and input vectors at the kth time step. Solving the above optimization problem yields an optimal input sequence that brings the vehicle from the initial trim state $\mathbf{x}_{from}$ to the desired trim state $\mathbf{x}_{to}$ in N time steps ΔT. If the cost function J is linear, the problem becomes a linear program (LP) and can be readily solved using standard optimization software.

The shortest time maneuver corresponds to the minimum number of time steps N_{min} for which the problem is still feasible. The search for N_{min} can be carried out by solving a series of LP's, in which the number of time steps is systematically decreased to check for feasibility. An efficient method is to use a bisection approach: at each iteration, the interval in which the minimal time $T_{min} = N_{min}\Delta T$ lies, is halved.

Since the set of feasible candidate solutions is independent of the specific cost function, a quick feasibility check can be carried out by solving the LP with a simple cost function. Once the minimum number of inputs is found, a more realistic linear cost function can be used to compute the final optimal maneuver. In our case, as a measure of fuel consumption, we minimized a weighted sum of the absolute values of the inputs, which can be linearized by introducing auxiliary variables and constraints.

3.3.2 MA Connectedness and Controllability

Given a MA, we use the value function to find the optimal sequence of maneuver primitives to direct the vehicle to the goal state (the hybrid state combining the current position of the vehicle in the inertial space with the trim state of the vehicle). Notice that this sequence of maneuver primitives form the optimal control law $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ where either $\mu_i^* \in Q_T$ or $\mu_i^* \in Q_M$. Therefore, when we discuss reconfiguration of value function-based systems, we notice the strong connection between the connectivity of maneuver primitives and the controllability of the vehicle. If we are unable to find a sequence of connected maneuver primitives from a given hybrid state, then the vehicle cannot be directed to the goal state for the given MA. These results are summarized in Lemma (3.1) and Lemma (3.2) in the previous section.

3.3.3 Controllability Core vs Performance Set

This discussion brings up another key issue: given a MA, can we find a set of maneuver primitives that define whether a region of the value function will have a finite cost? First, we know that the structure of the MA will define the achievable destination regions of the cost function. The cost for a given state $x_{int} = [x_o, q_{int}]$ is dependent on the current trim state, the connectivity of the MA, and the initial position of the vehicle. For example, if we constrain a vehicle to planning in the longitudinal axis, then the vehicle must have the control authority available to climb/dive to the desired goal state. For example, if the vehicle's flight path angle is restricted to $|\gamma| < 10$ degrees, then it will not be able to reach a waypoint located at $x_f = [\Delta_x = 10000ft, \Delta_z = 10000ft]$ without maneuvering in the lateral/directional axis. In this case, the vehicle's inability to achieve steeper flight path angles limits its capability to develop an feasible trajectory to this waypoint.

Next, we must determine which maneuver primitives from our MA are needed to guarantee that an admissible sequence of primitives from a given initial trim q_o exists to guide the system to a desired destination state in a given time T [9].

In Chapter 2 we separated all trim states into two categories: those which satisfy the controllability constraints (known as the "controllability core"), and those which provide enhanced performance (known as the "performance set"). First, maneuver primitives in the "controllability core" define the achievable destination region of the vehicle. For example, if a trim state in the controllability core is removed from the MA, the vehicle will lose capability to reach the goal from some of the achievable regions identified by this trim. This results in a region of the cost function having infinite cost under the reconfigured MA policy. In order to use the nominal cost function in the reconfiguration strategy, we must first catalog all of the controllability core maneuver primitives of the nominal MA and then determine which controllability core maneuver primitives (if any) have been added or removed in the reconfigured MA. Otherwise, the vehicle may enter into a state in the nominal cost function which is not defined for the reconfigured policy, and thus the vehicle will not reach the desired goal state. Second, maneuver primitives in the "performance set" add vehicle performance by taking advantage of the vehicle's capability throughout the operating envelope. Therefore, if a performance set maneuver primitive is removed from the MA, the vehicle will lose some performance characteristics, though the reachable region of the vehicle will be maintained.

Since guidance value functions are constantly decreasing, a guidance solution can be found from any initial condition y_0 in the reachable region of the vehicle even if a performance maneuver primitive is removed from the MA. Therefore, since the loss of a performance maneuver primitive does not affect the controllability of the vehicle (by definition), the cost at y_0 satisfies $J^*_{rcfg}(y_0) < \infty$ by Lemma (3.1). But, what does this tell us about $J^*_{nom}(y_0)$ in terms of the reconfiguration strategy? We know that if we are able to find a set of maneuver primitives that can move the system to the "controllable" regions of our robust value functions, then the cost at y_0 in the nominal value function is less than ∞. If we are able to find this set of maneuver primitives under the reconfiguration strategy, the cost at y_0 is "valid" for the reconfiguration strategy. We know that this condition will always exist whenever we add maneuver primitives to the MA. In addition, since the value function is strictly decreasing, we know that if a set of controls that move the vehicle into a region defined by the controllability core exists, the cost in the nominal value function will be "valid" for the reconfiguration strategy. From these conditions we can state the following condition:

Lemma 3.4 *Let $\mathcal{U}_{nom}$ be the set of all connected controls where $\mu_k \in \mathcal{U}_{nom}$ for the nominal system and let $J^*_{nom}(x)$ satisfy Eq. (2.1) $\forall\ \mu_k \in \mathcal{U}_{nom}$ and $x \in \mathcal{X}$. Define $C \subset \mathcal{X}$ as the set where $\forall\ x \in C \subset \mathcal{X}$, where $J^*_{nom}(x) < \infty$ is strictly decreasing in finite time under f for some control policy $\pi_y = \{\mu^*_0, \mu^*_1, \ldots, \mu^*_{N-1}\}$ where the set of controls $\{\mu_k\} \subset \mathcal{U}_{rcfg}$ (i.e. $J^*_{nom}(f(x_i, \mu(x_i))) \leq J^*_{nom}(x_i)$ and $J^*_{nom}(f(x_{N-1}, \mu(x_{N-1}))) = 0$). Then, for any $y \in \mathcal{X}$, if there exists a sequence of controls π_y where $\{\mu_k\}_{\pi_y} \subset \mathcal{U}_{rcfg}$ such that $J^*_{nom}(f(y_i, \mu(y_i))) \leq J^*_{nom}(y_i)$ and $x = f(y_i, \pi(y_i)) \in C$, then $J^*_{nom}(y)$ is valid under $\mathcal{U}_{rcfg}$.*

This lemma says that given an initial state under the reconfiguration strategy, if we can find a policy that moves the vehicle to a achievable destination region in the nominal value function (i.e. connected by the controllability core to the goal state) under the reconfigured MA, then the cost of the nominal value function at that initial state is valid for the reconfiguration strategy.

Proof: Let $x = f(y, \pi(y)) \in C$ where $J^*_{nom}(x) < \infty$. We can write $J_{rcfg}(y) = h(y, \mu(y)) + J^*(f(y, \pi(y)))$ where we can define $h(y, \pi(y)) = \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k))$ which is the current cost under control policy π_y. By the connectivity condition imposed on $\mathcal{U}_{rcfg}$, $h(y, \pi(y)) < \infty$. Finally, because of the decreasing condition of $J^*_{nom}(x)$ for $x \in C$, $\exists$ a control policy π_x where $J^*_{nom}(f(x, \pi(x))) = 0$, the combination of the two policies $\pi_{(y,x)}$ reaches the goal state. $\square$

Using this information we will be able to understand the cost differences between the nominal and recalculated value functions.

3.3.4 Reconfiguration of the MA-Based System

Using this information and the knowledge from the first case study, we would again like to investigate if we can use the nominal value function for the reconfigured system.

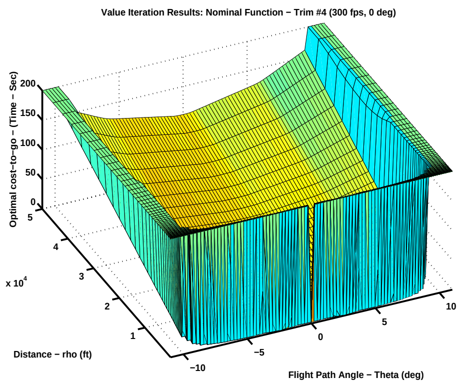


Figure 3-25: Nominal MA-Based Value Function for Trim 4 (300 fps, 0 deg) for 9-Trim Reconfiguration Tests

First, we investigate how the value function-based guidance system for a simplified Longitudinal Axis system changes after we remove trims and maneuvers. Second, we explore how this system changes after we remove maneuvers from the MA. Finally, we investigate the performance of this system when we add maneuvers to the MA.

For most of these examples, we define the nominal systems as detailed in Table A.5 (for the 9-Trim system) listed in Appendix A. Using this MA, the value function for the nominal system is provided in Figure 3-25. Since we are planning in the longitudinal axis frame, we must plan trajectories based on the following criteria: 1) initial and final trim state $q_i = [v_i, \gamma_i]$ where v_i is the vehicle velocity in the body frame and γ_i is the flight path angle of the vehicle, and 2) initial and final vehicle position in the longitudinal inertial frame (x_{pos}, z_{pos}). In Figure 3-26, we show an example of a trajectory using the Nominal Longitudinal MA VF-based system with LP-generated maneuvers. For this example, we selected a waypoint where $x_f = (\Delta$ in Distance $=$ 14000 ft, Δ in Altitude $= 800$ ft$)$. From this figure, we see that the trajectory "steps" upward toward the goal. This phenomena is a result of the LP-maneuver design: it is better for the vehicle to gradually increase its altitude by taking advantage of low-cost maneuvers between trim states.

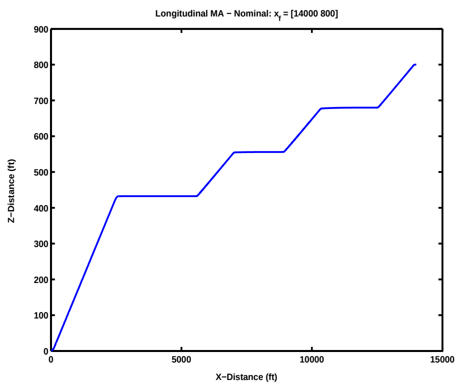


Figure 3-26: Trajectory for $x_0 = [14000, 800]$ using Simulink VF-Based System using Nominal VF for 9-Trim Reconfiguration Tests

Test 1: Trim Removal

In this first test, we examine what happens to the system trajectory when we remove trim states from the MA. This test is broken into two parts: we explore the effects on the VF-based system when 1) a trim state in the controllability core is removed, and 2) a trim state in the performance set is removed.

Part 1: Controllability Core Trim Removal

In this first example, we discuss what happens to the Guidance VF-based system after a controllability core trim is removed from the MA. In this first example, we will assume the vehicle has a structural failure during a mission and cannot climb more than +7.5 degrees. For this failure condition, the vehicle will no longer be able to reach Trim State 9 (300 fps, +10 deg). The reconfigured MA will look as described in Table A.6.

In Figure 3-27, we see the recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-25. Since there are no trim states in the reconfigured MA for flight path angles above +5 degrees, it is very clear that the reconfigured vehicle will be unable to reach the goal where the vehicle is required to climb more than +5 degrees. As discussed earlier, we can employ the reconfiguration strategy on the nominal value function to account for the loss of Trim 9 and plan paths for the reconfigured MA. Note that there is *no*

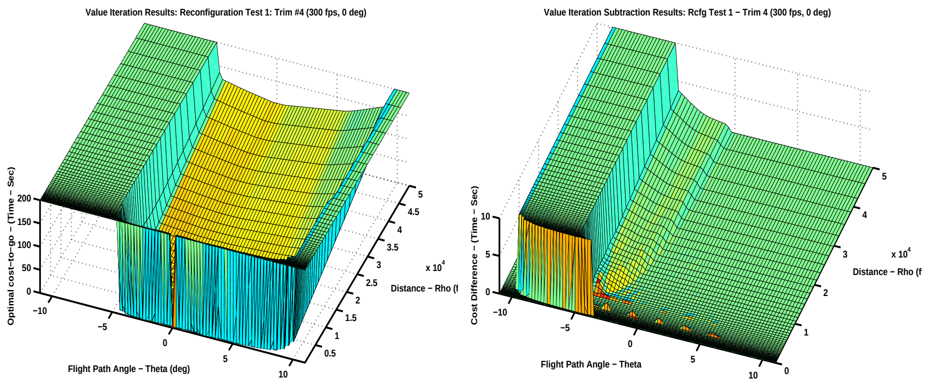


Figure 3-27: Recalculated MA-Based Value Function for Test 1 - Part 1

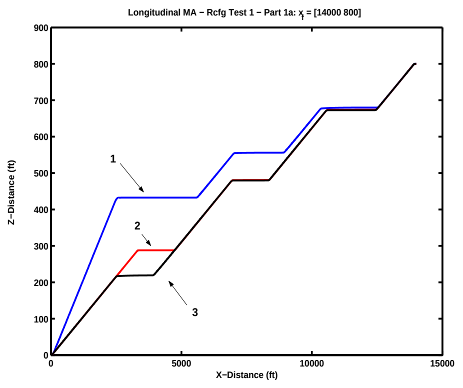


Figure 3-28: Reconfigured MA for Longitudinal MA Test 1 - Part 1a: Trajectory for (Δ in Distance = 14000 ft, Δ in Altitude = 800 ft)

change in the computation time required to calculate an achievable trajectory using the reconfiguration strategy.

To test the reconfiguration strategy for this failure condition, we selected waypoints where Trim 9 was used in the path computed with the nominal value function. Then, we planned trajectories to the same point using the reconfiguration and recalculation strategies. For the point (Δ in Distance = 14000 ft, Δ in Altitude = 800 ft), the difference between the nominal (1), reconfigured (2) and recalculated (3) trajectory is shown in Figure 3-28 and their projected times are compared in Table 3.7.

Again, we see that all three strategies calculate trajectories which guide the system toward the goal. Notice that the reconfigured and recalculated strategies follow

Traj	Name	Time (sec)
1	Nominal (Blue)	32.26
2	Reconfigured (Black)	33.37
3	Recalculated (Red)	33.37

Table 3.7: Time Comparison for Longitudinal MA Reconfiguration Test 1: Part 1a

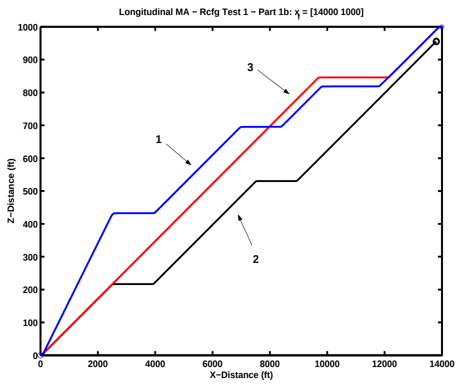


Figure 3-29: Reconfigured MA for Longitudinal MA Test 1 - Part 1b: Trajectory for (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft)

slightly different trajectories but have almost the exact same cost. From this example, we see that our reconfiguration strategy provides a good approximation of the recalculated optimal trajectory for this waypoint without recalculating the value function for the reconfigured MA. But, again, this is not always the case. Under this strategy, it is possible that the reconfigured strategy cannot compute a path to goal, even though a trajectory exists using the recalculated cost function. In Figure 3-29, we have selected the waypoint (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft) and their projected times are compared in Table 3.8.

From this example we can see a problem that may arise when using the reconfiguration strategy. Clearly, there is a feasible trajectory to the goal using the reconfigured

Traj	Name	Time (sec)
1	Nominal (Blue)	33.47
2	Reconfigured (Black)	No Solution
3	Recalculated (Red)	34.71

Table 3.8: Time Comparison for Longitudinal MA Reconfiguration Test 1: Part 1b

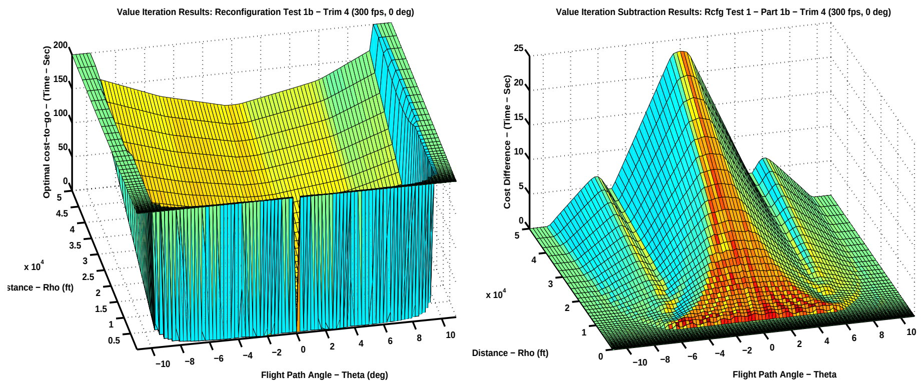


Figure 3-30: Recalculated MA-Based Value Function for Test 1: Part 2

MA - as demonstrated by the recalculated MA trajectory in Figure 3-29. But, notice that the reconfigured strategy does not reach the goal. Since the nominal cost function is calculated using trim 9, the reconfigured strategy will follow the gradient of the value function which assumes trim 9 is available for use. Although the reconfiguration strategy is minimizing the cost at each decision, it is not checking (or "looking ahead") to see if it will be able to reach the goal *after* after it makes a decision - resulting in a situation where a feasible trajectory under this policy appears unachievable. If we employ a "look ahead" strategy to ensure that the selected command for a given trajectory will result in a final *achievable* solution for a given goal state, we find that the system will reach the goal when a solution is feasible.

Part 2: Performance Set Trim Removal

In this second example, we discuss what happens to the Guidance VF-based system after a performance set trim is removed from the MA. We will assume the vehicle has a structural failure during a mission and cannot climb at speeds greater than 450 fps. For this failure condition, the vehicle will no longer be able to reach Trim State 6 (500 fps, 0 deg). The reconfigured MA for this test is described in Table A.7.

In Figure 3-30, we see the recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-25. Since there are no trim states in the reconfigured MA for velocities above 450 fps, the vehicle will require more time to reach the goal from areas of the achievable space.

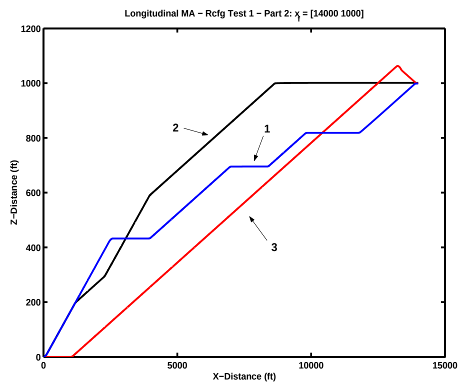


Figure 3-31: Reconfigured MA for Longitudinal MA Test 1 - Part 2: Trajectory for (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft)

Traj	Name	Time (sec)
1	Nominal (Blue)	33.47
2	Reconfigured (Black)	37.35
3	Recalculated (Red)	36.45

Table 3.9: Time Comparison for Longitudinal MA Reconfiguration Test 1 - Part 2

As discussed earlier, we can employ the reconfiguration strategy using the nominal value function to account for the loss of Trim 6 and plan paths for the reconfigured MA without significant changes in the computation time required to calculate an achievable trajectory.

To test the reconfiguration strategy for this failure condition, we selected waypoints where Trim 6 was used in the path computed with the nominal value function. Then, we planned trajectories to the same point using the reconfiguration and recalculation strategies. For the point (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft), the difference between the nominal (1), reconfigured (2) and recalculated (3) trajectory is shown in Figure 3-31 and their projected times are compared in Table 3.9.

Again, we see that all three strategies calculate trajectories guide the system toward the goal. Though the reconfigured and the recalculated strategies calculate different trajectories to the goal, there is a small difference in the cost for these trajectories. Notice that the recalculated trajectory overshoots the goal altitude because

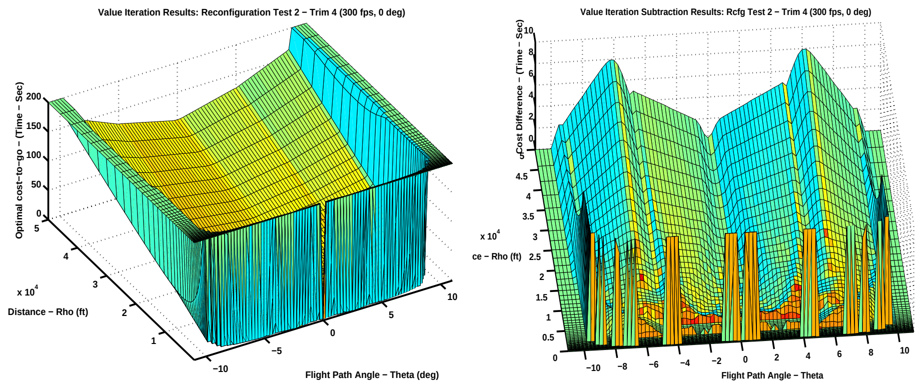


Figure 3-32: Recalculated MA-Based Value Function for Test 2

of the cost to maneuver between each trim state. From this example, we see that the reconfiguration strategy gives our vehicle a trajectory that achieves the goal state without recalculating the value function for the reconfigured MA. But, just as in Part 1, since we are minimizing the cost over each transition rather than the trajectory this is not always the case. By using a "look ahead" strategy, the system can determine if a trajectory is feasible for a given goal state.

Test 2: Maneuver Removal

In the second test, we examine how the system's trajectory changes when we remove maneuvers from the MA. As mentioned above, if the vehicle experiences a failure, it may not have the capability to perform some of the primitives in the MA. In this example, we will assume that vehicle has a failure during a mission that limits the system to a subset of the maneuvers in the nominal MA. For this failure condition, the reconfigured MA is described in Table A.8.

The recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function (in Figure 3-25) are shown in Figure 3-32. Notice that by reducing the number of available maneuver primitives, there is an increase in cost over most of the value function. In addition, there are regions in the recalculated value function where the vehicle can no longer maneuver to the goal for this under the reconfigured MA. This reduction in the vehicle's "achievable" region is caused by

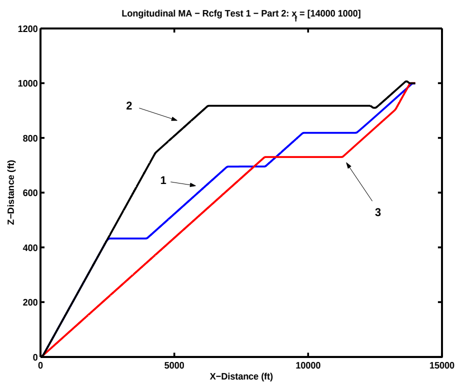


Figure 3-33: Reconfigured MA for Longitudinal MA Test 2: Trajectory for (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft)

Traj	Name	Time (sec)
1	Nominal (Blue)	33.47
2	Reconfigured (Black)	37.54
3	Recalculated (Red)	35.83

Table 3.10: Time Comparison for Longitudinal MA Reconfiguration Test 2

removing some of the maneuvers between the controllability core trims.

Next, we can employ the reconfiguration strategy on the nominal value function to account for the loss of these maneuvers and plan paths for the reconfigured MA. To test the reconfiguration strategy for this failure condition, we selected waypoints where the removed maneuvers were used to compute paths under the nominal strategy. We then used the reconfiguration and recalculation strategies to plan trajectories to these waypoints. For the point (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft), the difference between the nominal (1), reconfigured (2) and recalculated (3) trajectory is shown in Figure 3-33 and their projected times are compared in Table 3.10.

As in Test 1, we see that all three strategies calculate trajectories which guide the system toward the goal. Again, the reconfigured and recalculated strategy calculate slightly different paths, but the difference in the cost between both trajectories is small. Notice that the reconfigured trajectory actually overshoots the goal altitude near the goal because the planner selects this maneuver sequence to minimize the cost at each transition as the vehicle approaches the goal. Again, we see that our

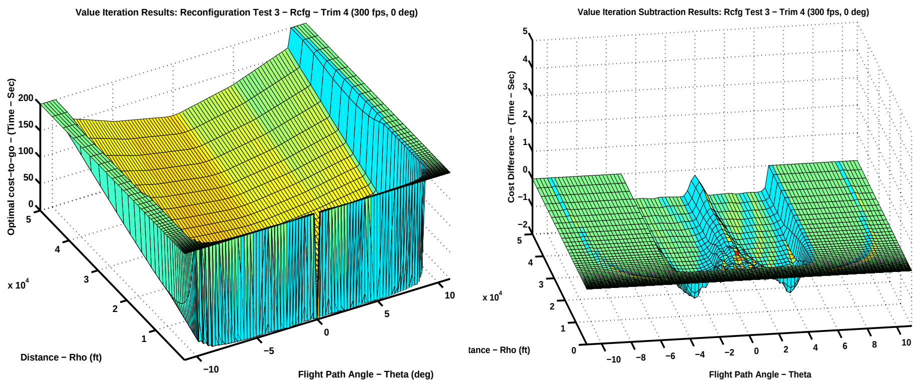


Figure 3-34: Recalculated MA-Based Value Function for Test 2

reconfiguration strategy gives our vehicle a trajectory that achieves the goal state even after a fraction of maneuvers have been removed from the MA without recalculating the value function for the reconfigured MA.

Test 3: Maneuver Addition

In the third example, we will discuss adding maneuvering performance or capability to the system. To illustrate this point, the nominal and reconfigured vehicle will have the maneuver automata described in Table A.9 and Table A.10 respectively. Notice that we are using the reconfigured MA from Test 2 as our nominal system in Test 3. The nominal value function is shown on the left in Figure 3-32.

The recalculated cost function for the reconfigured MA and the differences in cost for Test 3 from the nominal value function for Test 3 are shown in Figure 3-34. Notice that there is a reduction in cost in the regions of the value function where these maneuvers (which improve system performance) can be used. To compare the reconfiguration strategies, we planned trajectories to the same point using the nominal, recalculated and reconfiguration strategies. For the point (Δ in Distance $= 14000$ ft, Δ in Altitude $= 1000$ ft), the difference between the nominal (1), reconfigured (2) and recalculated (3) trajectory is shown in Figure 3-35 and their projected times are compared in Table 3.11.

As expected, the recalculated trajectory is faster than the nominal trajectory

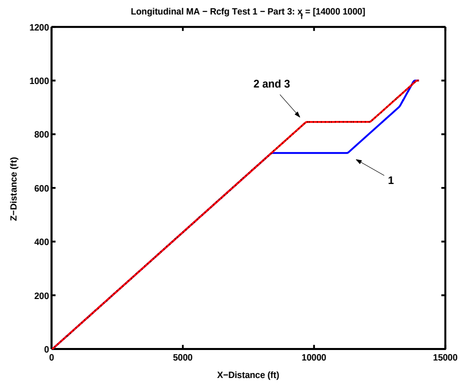


Figure 3-35: Reconfigured MA for Longitudinal MA Test 3: Trajectory for (Δ in Distance = 14000 ft, Δ in Altitude = 1000 ft)

Traj	Name	Time (sec)
1	Nominal (Blue)	35.83
2	Reconfigured (Black)	34.71
3	Recalculated (Red)	34.71

Table 3.11: Time Comparison for Longitudinal MA Reconfiguration Test 3

since it uses the performance maneuvers in planning its trajectory. In addition, the reconfigured trajectory also uses the performance maneuvers in planning its trajectory. Unlike Test 2 in Case Study I, the reconfigured strategy uses the performance enhancements to the MA and plans the same trajectory used in the recalculated strategy. Again, the reconfiguration strategy gives our vehicle a good approximation of the recalculated optimal trajectory for this point without recalculating the value function for the reconfigured MA.

3.3.5 Summary

In this case study, we showed that the nominal value function can be used for trajectory generation with a reconfigured MA. We demonstrated how performance and controllability changes to the MA affect the performance of the system under each trajectory generation strategy. We also showed that the reconfiguration strategy can provide path that may not reach the goal state when a feasible trajectory exists. By using a “look-ahead” strategy (as demonstrated in Case Study I), we can use the value function to find trajectories that will move the vehicle toward the goal and then select the trajectory that will provide the best cost over the length of the trajectory. Finally, by restricting the vehicle to motion planning in the longitudinal plane, we may not be able to find a feasible trajectory to a desired waypoint, though the vehicle may possess the capabilities to reach the destination. In these tests, since we did not allow the vehicle to maneuver in the lateral/directional axis, some waypoints were not achievable because the vehicle was unable to climb/decend to the desired altitude in the distance provided in the longitudinal plane. In the next chapter we will discuss how path planning for vehicles with six degrees of freedom by combining a longitudinal and lateral/directional axis planner can be performed.

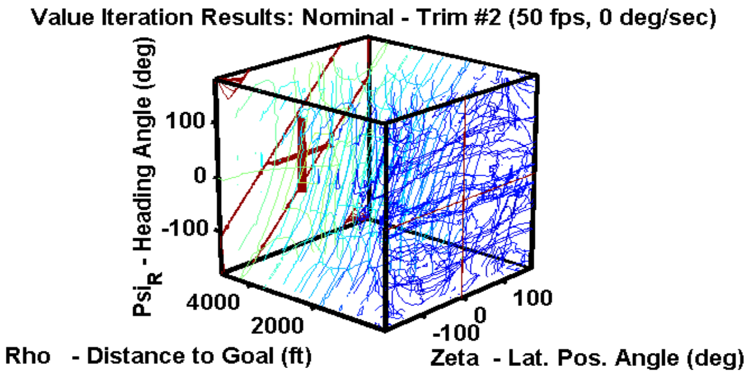
3.4 Case Study III: Lateral MA Control

The final system we will investigate is a simplified closed-loop model of a lateral/directional axis small Fixed-Wing UAV maneuver automaton. Since we are planning in the lateral/directional axis, we must plan trajectories based on the following criteria: 1) initial and final trim state $q_i = [v_i, \dot{\psi}_i]$ where v_i is the vehicle velocity in the body frame and $\dot{\psi}_i$ is the turn rate of the vehicle in the body frame, 2) initial and final vehicle position in the lateral/directional inertial frame (x_{pos}, y_{pos}), and 3) initial and final vehicle heading where North = 0 deg (360 deg).

3.4.1 Reconfiguration of the Lateral-Directional MA-Based System

Using our knowledge from the other case studies, we would like to investigate if we can use the nominal value function for the reconfigured system. In this test, we will investigate how the value function-based guidance system for a simplified Lateral/Directional Axis system changes after we remove trims and maneuvers.

For this test, we define the nominal simplified 3-Trim system in Table A.11 listed in Appendix A. In this test we require that the vehicle starts and ends its trajectory



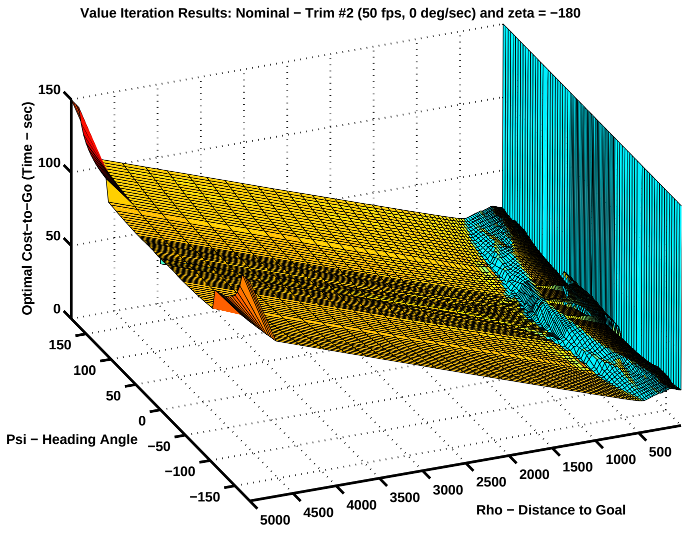


Figure 3-36: Nominal MA-Based Value Function for Trim 2: (50 fps, 0 deg/sec), (Bottom: Angular Position (ζ) = 180 deg.) for 3-Trim Lateral/Directional Reconfiguration Tests

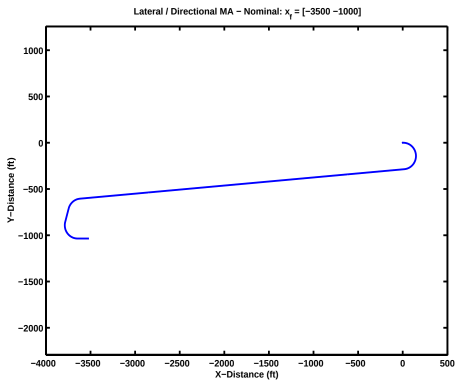


Figure 3-37: Trajectory for $x_0 = [-3500, -1000]$ using Simulink VF-Based System using Nominal VF for 3-Trim Reconfiguration Tests

pointing East ($\psi_o = 90$ deg). The value function for the nominal Lateral/Directional MA system is provided in Figure 3-36. In Figure 3-37, we show an example of a trajectory using the Nominal Lateral/Directional MA VF-based system. For this example, we selected a waypoint where x_f = (Δ in X-Distance = -3500 ft, Δ in Y-Distance = -1000 ft). Since the waypoint is behind the vehicle, the vehicle first turns around, moves toward the goal and then turns back toward the final destination. Notice that the vehicle coasts slightly to "line up" on the correct heading on its final approach. This coasting period is a result of the maneuver design combined with the grid alignment in this portion of the value function. Since the lateral/directional value functions are four-dimensional (trim q_i, distance from goal ρ, angular position with respect to final heading in inertial frame ζ, vehicle heading in inertial frame ψ), the cost is interpolated across eight points. To reduce the size of the value function, the value function grid has more points near the goal region to improve the performance of the guidance system near the goal. Therefore, as the vehicle approaches the goal, it may coast slightly to reduce position errors on its final approach when aligning itself with the final heading.

Test 1: Trim Removal

In this test, we examine what happens to the system's trajectory when we remove a trim state from this MA. For this MA, this means that the reconfigured vehicle can

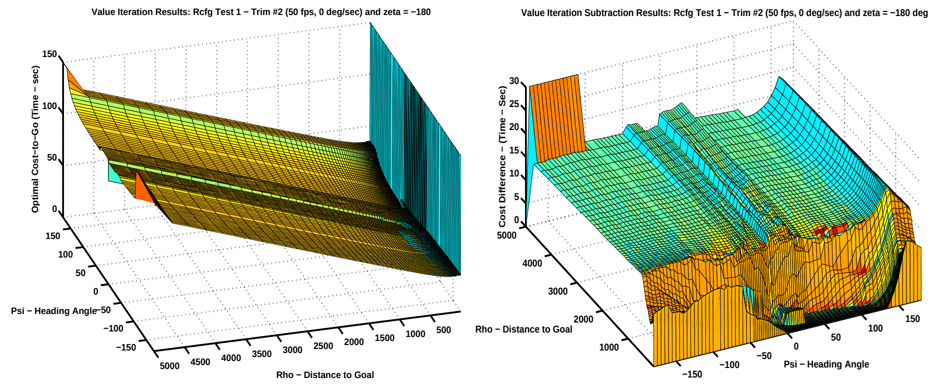


Figure 3-38: Recalculated Lateral/Directional MA-Based Value Function for Test 1

only turn in one direction. Since the vehicle will have one trim to turn the vehicle, this test explores the effects of removing a performance set trim from a Lateral/Directional VF-based Guidance system [9]. For practical purposes, we can assume that the vehicle has a structural failure during a mission and is unable to turn right. For this failure condition, the vehicle is unable to reach Trim State 3 (50 fps, +20 deg/sec). The reconfigured MA is described in Table A.12.

In Figure 3-38, we see the recalculated cost function for the reconfigured MA and the differences in cost from the nominal value function in Figure 3-36. Since the vehicle is unable to turn right, the vehicle incurs a large cost penalty for having to turn to correct for vehicle heading along the final approach. We can employ the reconfiguration strategy on the nominal value function to account for the loss of Trim 3 and plan paths for the reconfigured MA without significant changes in the computation time required to calculate an achievable trajectory using the reconfiguration strategy.

To test the reconfiguration strategy for this failure condition, we selected waypoints where Trim 3 was used in the path computed with the nominal value function. Then, we planned trajectories to the same point using the reconfiguration and recalculated strategies. For the point (Δ in X-Distance = 2000 ft, Δ in Y-Distance = -500 ft), the difference between the nominal (1), reconfigured (2) and recalculated (3) trajectory is shown in Figure 3-39 and their projected times are compared in Table 3.12.

This test shows that all three strategies calculate trajectories which guide the

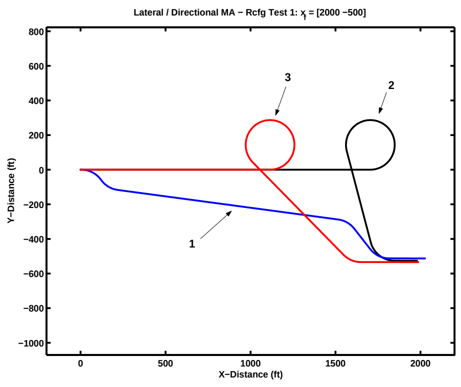


Figure 3-39: Reconfigured MA for Lateral/Directional MA Test 1a: Trajectory for (Δ in X-Distance = 2000 ft, Δ in Y-Distance = -500 ft)

Traj	Name	Time (sec)
1	Nominal (Blue)	43.40
2	Reconfigured (Black)	65.75
3	Recalculated (Red)	62.25

Table 3.12: Time Comparison for Longitudinal MA Reconfiguration Test 1a

system toward the goal. Notice that the reconfigured and recalculated strategies use slightly different trajectories but have small differences in cost. The reconfigured strategy waits to turn toward the goal since its cost effective under the original strategy to maneuver to a left-hand turn rather than turn right. The vehicle continues on the straight trajectory until its absolutely necessary to turn to reach the goal using the nominal value function. But, this example shows that the reconfiguration strategy gives our vehicle a good approximation of the recalculated optimal trajectory for this point without recalculating the value function for the reconfigured MA. Again, this is not always the case. Since the nominal strategy has the ability to use both right- and left-hand turns, the reconfiguration strategy will wait to maneuver into a turn until it is absolutely necessary. In Figure 3-40, we have selected the waypoint (Δ in X-Distance = -3000 ft, Δ in Y-Distance = 2000 ft) and the projected trajectory times for each strategy are compared in Table 3.13.

This example clearly shows the types of problems that may arise when using the reconfiguration strategy. Clearly, all three trajectories develop paths to the goal

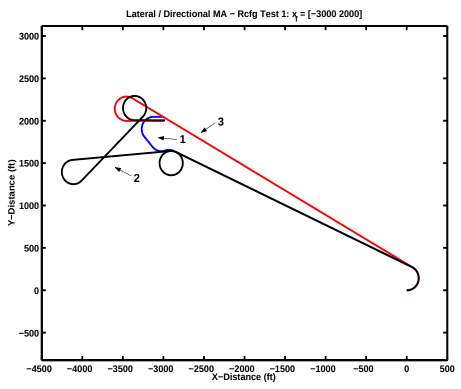


Figure 3-40: Reconfigured MA for Lateral/Directional MA Test 1a: Trajectory for (Δ in X-Distance = -3000 ft, Δ in Y-Distance = 2000 ft)

Traj	Name	Time (sec)
1	Nominal (Blue)	90.25
2	Reconfigured (Black)	171.40
3	Recalculated (Red)	107.40

Table 3.13: Time Comparison for Longitudinal MA Reconfiguration Test 1b

state - as demonstrated in Figure 3-40; however, because the reconfigured strategy is minimizing the cost at each decision, the planner waits until it *must* turn the vehicle in order to reach the goal. As a result, the vehicle spends an extra 60 sec re-orienting itself in the proper direction to reach the goal state.

3.4.2 Summary

In this case study, we showed that the reconfiguration strategy can be used for path planning in the Lateral/Directional Axis. In this example, we demonstrated how changes to the performance to the MA affect system performance under each trajectory generation strategy. Finally, it is important to note that the trajectory planning for this scenario was performed in an obstacle-free environment. Adding obstacle would further complicate this scenario by introducing the need for the vehicle to safely maneuver in a confined environment. As a result, the vehicle may require higher levels of performance in order to avoid threats. In fact, it is clear that a two-trim MA may not be sufficient to control a vehicle in such an environment. Therefore, when we discuss vehicle performance and controllability it is important to consider the vehicle's environment in addition to the structure of the MA.

3.5 Case Study Summary

These case studies demonstrates that the nominal cost function can be used with a reconfigured MA to plan trajectories for the reconfigured vehicle. Using the reconfiguration strategy for an initial state x_{init}, we showed that if the Value Function-based Guidance system is able to find a trajectory that moves the vehicle to a region in the value function with a known, finite cost under the reconfigured MA, then the nominal cost function is "valid" at this initial state. We also showed that in some cases, the reconfiguration strategy will not produce a feasible trajectory to the desired goal state. Since the trajectory planner is minimizing the cost-to-go at each decision, the planner may direct the vehicle on a path that will not result in a feasible solution because the value function was calculated using a maneuver primitive that resulted in the optimal achievable solution from the initial state. Therefore, since the gradient of the cost along the vehicle trajectory will be decreasing until the vehicle is forced to use the maneuver primitive, the system will continue to follow the trajectory until the system reaches the state where it must use the unachievable maneuver primitive - thus directing the vehicle to an infeasible solution. To alleviate this condition we showed that we can calculate a feasible solution using a strategy that plans over multiple decisions, i.e. a "look ahead" strategy. First, we determine which transitions will lower the cost-to-go from the current state to the next state. Second, from this subset of commands we check to see if we can calculate a feasible solution from the next state to the goal state. Then, we select the maneuver primitive sequence which minimizes the cost-to-go over both transitions. We showed that this technique helps to ensure that the planner uses the nominal cost function with the reconfigured MA to find a feasible solution if it exists.

We investigated the differences between robustness and performance. We showed that we can add performance to an MA-based guidance system by adding maneuver primitives or more grid points to the value function grid. By calculating the cost at more points in the value function, the interpolated cost between grid points is more representative of the actual cost in the nominal value function calculated without maneuver primitives. We showed that adding performance to the system can reduce the robustness of the system. We demonstrated by an example that if there is a disturbance in the system around the switching curve in the double integrator examples, the system will have to maneuver to the "other" switching curve to reach the goal. If the disturbance is large enough, it is possible that the vehicle will be unable to maneuver to a lower cost in the value function at each transition.

Finally, we investigated the differences between performance and controllability. We showed that there trims (and the maneuvers connecting them) can be separated into two classes which describe the controllability and performance of the system from any point in the state space of the vehicle. We showed that losing either a controllability or performance maneuver primitive can have adverse effects on the system under the reconfiguration policy.

Chapter 4

The Implementation of Reconfiguration Systems

In this chapter, we discuss some of the issues and methods associated with the implementation of a reconfigurable value function-based guidance systems using a maneuver automaton. We discuss the use of feedback in the guidance loop with this system and present a technique for task planning with multiple vehicles using the guidance value function.

4.1 Value Function-Based Guidance System Implementation Issues

We have shown through examples that we can generate optimal trajectories using guidance value functions. These functions are calculated using a value iteration performed with a nominal set of trim and maneuvers for our RMA. Since these value functions are stored in system memory, they are calculated across a set of grid points (for ease of storage) and the cost-to-go for the system at any hybrid state $x = [q, x_{pos}]$ can be obtained through interpolation between grid points. Thus, defining the grid points influences the shape the value function and must be done carefully.

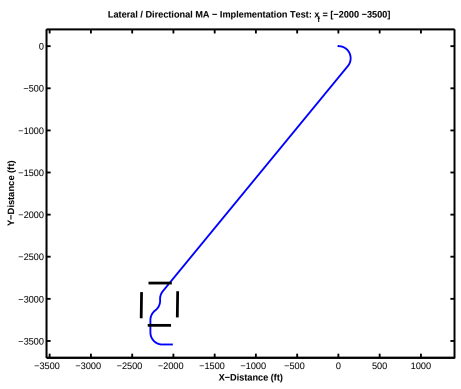


Figure 4-1: Effects of Grid Coarseness on Lateral/Directional Axis Trajectory Planning

4.1.1 Value Function Discretization

In Chapter 3, we showed that adding grid points to the cost function helps to improve system performance over the vehicle's achievable region. But, there are other grid implementation issues that must be addressed. First, the grid size and coarseness throughout the vehicle's achievable region must be considered. Using a Pentium II - 300 MHz Microprocessor, we found a strong correlation between the number of grid points, the size of the MA and the computation time of the value iteration. To decrease computation time while maintaining performance, we can design the grid to have a higher concentration of points around the goal state to improve path planning accuracy near the goal. This causes the gradient of the value function to be more representative of the actual cost-to-go along system trajectories, thus improving system performance around the goal state without adding a large number of points across the entire value function. Although this technique reduces the size of the value function in the system's memory, the system may plan achievable trajectories to the goal that may require the vehicle to re-adjust its heading as it approaches the goal.

For example, in Figure 4-1 we show a trajectory calculated using the 3-Trim Lateral/Directional Planner with the Nominal Value Function from Figure 3-36. In this figure, we see that the grid has more points near the goal along the radial (ρ) axis. This smaller spacing of grid points near the goal is important for the Lateral/Directional axis path planner. Since we are using trims with a set velocity and turn rate (which imply a set turn radius), there is a minimum distance the vehicle must be from the goal to directly approach the goal without turning around [21]. Therefore, as the vehicle gets closer to the goal, the value function must have a higher concentration of grid points to produce a feasible trajectory for the vehicle; otherwise, the trajectory generation system may plan paths that direct the vehicle around in circles before finding an appropriate trajectory. These extra grid points helps the path planner to align the trajectory of the vehicle along a proper approach heading in this region. In Figure 4-1, the heading of the vehicle is corrected as the vehicle approaches the goal - resulting in the small adjustment in the trajectory depicted in the black box in Figure 4-1. Note that selecting a fine grid size across the entire value function may improve slightly the accuracy of the value function, but the computation time required to calculate this function will grow exponentially for this small gain in accuracy.

This example helps to illustrate how the grid size of the value function effects the path planning capabilities of the guidance system. Some of our future work will focus on understanding the sensitivity of the path planner to the value function grid size. For example, this test helps to illustrate that the grid should be sized with a consideration for the implementation of the guidance law. If the system requires the guidance system to make calculations to the next waypoint at a high frequency, a finer grid size in the radial direction may be required to increase the accuracy of the system. However, a designer must also consider the need for a high frequency of the calculations given the aircraft mission - which may result in a need for a fine maneuver automaton.

4.1.2 Implementation of the Guidance Law

After calculating the value function, the motion planner will use the guidance law based on the value function's cost criteria to calculate trajectories for the vehicle. This guidance law is based on the optimality principle (in Thm. 3.1) and derived for the vehicle's guidance requirements. This guidance law (based on value iteration parameters) is selected to calculate the best path between destination points by evaluating the path over a specified time step. The guidance law time step must coincide with the coasting time step and grid resolution used in calculating the value function. For example, setting the coasting time step to a small value does not necessarily improve guidance system performance for coarse grid size. Therefore, the grid size, frequency of position calculations, guidance law, and coarseness of the maneuver automaton are related. Notice that increasing the frequency of computation or increasing the number of grid points does not guarantee the best path if the grid size and coasting time step does not have the same resolution.

This relationship raises issues related to the grid point evaluation of the motion planner. First, the guidance law must determine when the vehicle either coast or maneuver between destination points. The value function grid size must be selected to ensure that the required performance of the system under the guidance law selected. Finally, it is important to ensure that the guidance law is created in accordance with the controllability requirements for each state of the maneuver automaton. The guidance law must not violate the stability constraints imposed on the MA. For example, the guidance law must not command the vehicle to regions where the value function is undefined. On the other hand, the maneuver automaton must be "well-posed," so that the guidance law directs the system to the desired location. Otherwise, the guidance law will try to command the vehicle to states (defined by the maneuver automaton) that may be unachievable or unstable.

4.2 Implementation Methods for Reconfiguration

Next, in order to implement a reconfiguration technique, the vehicle must have a system that identifies the changes that need to be made in the guidance system based the current and new vehicle configuration. For example, if the vehicle has a failure during a mission, the guidance system must identify the vehicle's current capabilities before determining whether it needs to reconfigure itself in order to finish the mission and/or bring the vehicle back to safety. Ideally, we could design the system to recalculate the optimal value function for the new set of motion primitives. But, as mentioned before for MAs with a large number of maneuver primitives it can take a long time to recalculate the value function. Therefore, since the vehicle must have some baseline performance readily available, we need to use a reconfiguration technique that can guarantee that the vehicle can be navigated to safety. In this section we will discuss two different implementation methods for reconfiguration: pre-computing value functions and partitioning methods for the nominal value function.

4.2.1 Multiple Pre-Computed Value Functions

If we change any of the motion primitives in the MA, the nominal value function will not be optimal for the reconfigured vehicle. To account for this change in the MA, we can use pre-calculated value functions for reconfigured MAs corresponding to different failure or performance scenarios. As a result, the vehicle will be able to use the optimal value function for each MA configuration. The difficulty with this technique is that these functions would need to be saved for multiple failure/performance combinations. Therefore, a system using this technique would need enough memory to store the value functions for each system configuration.

In the event of a failure, the vehicle must first identify the current vehicle configuration and then load the proper value function. While the vehicle is determining the current vehicle configuration, the guidance system should command the vehicle into a safe, known state. For example, if the MA was designed such that every trim state is connected to every trim state in the controllability core, in the event of a failure we can maneuver the vehicle to a controllable trim state until the new system configuration is known. Then, the vehicle would recalculate the vehicle trajectory using the reconfigured MA - which will be optimal under this reconfiguration policy.

4.2.2 Value Function Partitioning

In Chapter 3, we showed that the nominal value function can be used with the reconfigured MA to calculate trajectories to the goal state. One of the major difficulties of this technique is determining the regions of the value function that are “not valid” under the new MA. One way to determine which trim states are valid under the new

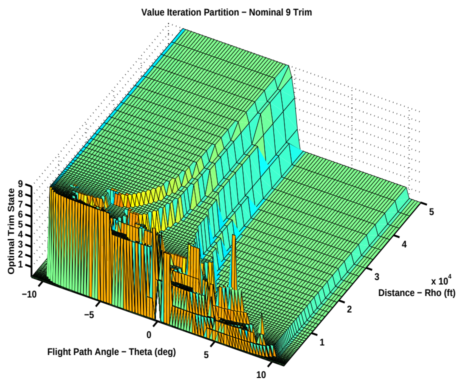


Figure 4-2: Partition Function for the 9-Trim Nominal Value Function

policy is to use a process called "partitioning." The basic idea behind this method is for each grid point in the value function, we determine which trim states are used under the optimal policy to direct the vehicle to the goal state and store this information with the function. Figure 4-2 shows an example of a partition function. This figure shows which trim is the most optimal to be used from each grid point in the nominal value function to reach the goal for the nominal MA. The partition function also includes the set of all trims (and the corresponding maneuvers) used to move the vehicle to the goal from each region.

After the MA is reconfigured, the guidance system can use this additional information to determine which maneuver primitives can be used to plan trajectories with the nominal value function with the reconfigured MA. Then, we can combine this technique with a "look ahead" policy to optimize the vehicle trajectory under the reconfigured MA.

4.3 Implementation on a Non-Linear UCAV Model

After investigating different reconfiguration techniques, we tested our methods on a non-linear model for a unmanned combat air vehicle (UCAV) simulation model. This model was provided to our research group for the Intelligent Control project, a jointly-funded research grant by the Air Force Research Laboratories and Office of Naval Research. Our task was to develop a guidance system that could be used with the non-linear model (developed in Simulink) and could calculate achievable trajectories after the vehicle model changes.

As part of this research, we experimented with all of the techniques detailed in this document. One of the obstacle our research team faced was the development of a health monitoring system with the ability to determine which maneuver primitives were achievable after a change significant the vehicle model. For the purpose of work, we were able to assume that the guidance system was provided with this information (i.e. turn rate limits, flight path angle limits, velocity limits). As a result, we chose to implement the method where we pre-computed multiple value functions for different vehicle configurations with baseline guidance performance to demonstrate the effects of reconfiguration at the guidance level after a change in the vehicle dynamics. We designed the guidance system to use the MA-based guidance value function method to calculate trajectories for the vehicle in the lateral/directional axis with a simple feedback control law with limited control authority to command the longitudinal axis of the vehicle. The design of the flight control system allowed us to de-couple most of the longitudinal and lateral/directional task planning.

As part of the final demonstration, the vehicle's guidance system had to command the vehicle to fly a "race track" loiter pattern and show performance when performing climb and descend tasks to multiple waypoints. In Figure 4-3, we show a picture of the vehicle flying a multiple waypoint race track pattern with waypoints requiring the vehicle to climb and descend along the trajectory. The guidance system was programmed to direct the vehicle to fly a spiral pattern to gain altitude over short distances, as shown in this figure. To help visualize the vehicle trajectories in an actual environment, we saved the simulation data calculated in Matlab using Simulink and used the AVDS Flight Simulation Software's Playback mode to show the output trajectories. Our tests showed that the vehicle performed well in the system's initial tests. After a series of tests, we found that error could "build up" between waypoints since the system was designed to recalculate trajectories for the vehicle to implement without a trajectory tracking loop. To address this problem, we increased the frequency at which the guidance system calculated the vehicle's trajectory to the

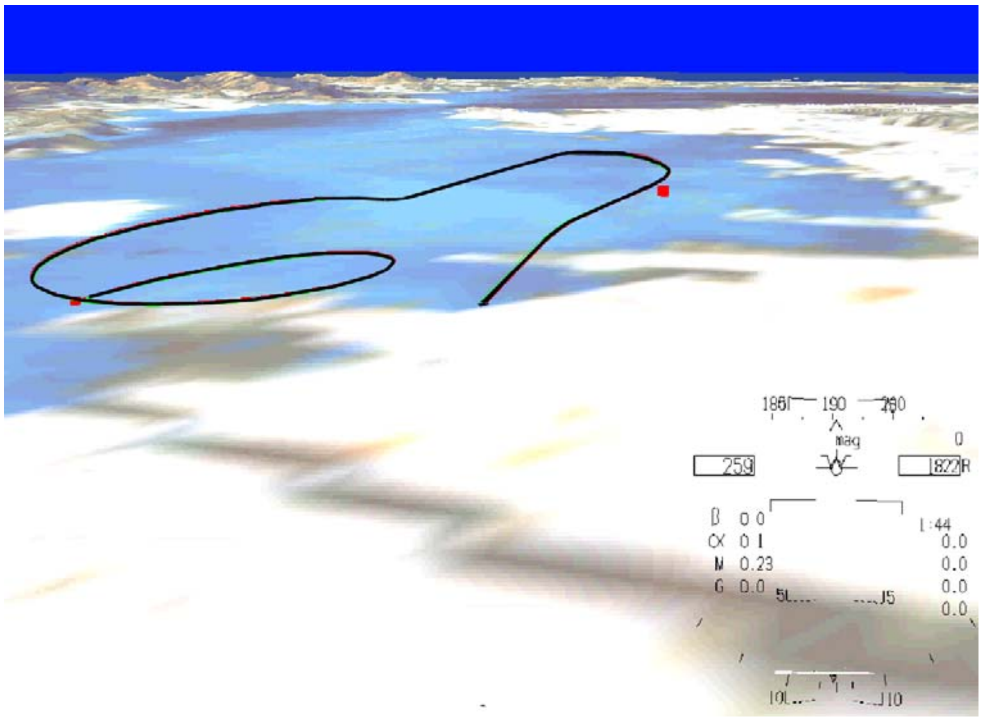


Figure 4-3: Trajectory Planning and Implementation of Reconfigurable Guidance System on UCAV Model

waypoint. Currently, we are investigating ways of improve the performance of this system and the vehicle's trajectory tracking between waypoints.

4.4 Feedback around Trim States

Another technique that can be used to reduce errors in trajectory planning is feedback around the trim trajectories. Since the trim trajectories are equilibria for the system, we can design the system to provide some control authority at each trim state in order to correct for errors in the vehicle's trajectory. Therefore, we can feedback the current vehicle's position and compare it with the actual vehicle's position to correct for heading and position errors in the trajectory en route to the goal state. This allows the system to take advantage of the system's robustness at each trim state, while allowing the system to correct for small errors in the vehicle's trajectory without replanning. For large heading and position errors, the system can replan the vehicle's trajectory from the new position.

This technique also changes the planning strategy for the vehicle. Instead of calculating an exact trajectory to the goal, the planner can calculate a reference trajectory from a starting waypoint to a "finishing region" by which the vehicle can use the feedback commands to reach the goal. This technique also can be used to reduce the number of computations required when calculating the value function. Since the limited control authority can used to "fine tune" the vehicle's trajectory, we can reduce the large number of grid points around the goal in the value function - thus decreasing the computation time need to recalculate the value function.

4.5 Task Allocation using the Guidance Value Function

Since the guidance value function is used to plan trajectories with the MA, this function encapsulates many different vehicle parameters and constraints. We have found that this value function can be used as a database for task allocation [18].

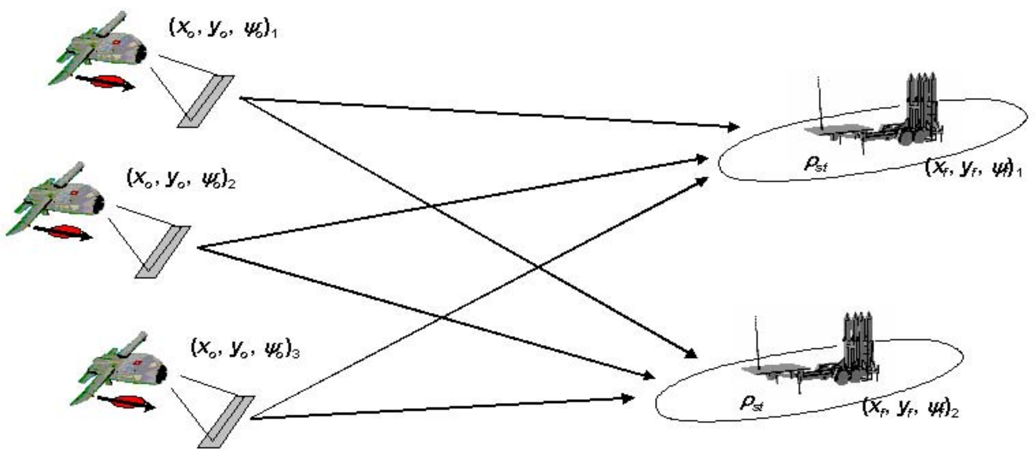


Figure 4-4: Task Allocation Scenario for Three Vehicles and Two Targets

4.5.1 Current Task Allocation Strategy

Researchers at the Air Force Research Laboratory (AFRL) have been investigating cooperative control and task allocation issues over the past four years. Currently, this work has focused on task allocation methods for Wide Area Search Munitions (WASM) random scenarios (as shown in Figure 4-4) using a decision-making algorithms based upon a capacitated transshipment network model [16]. This system allocates vehicle tasks (comprised of Search, Classify, Attack, and Battle Damage Assessment - BDA) by finding the best solution using a matrix of each vehicle's cost to perform each target. To calculate these costs, the system uses the vehicle's path planning system, which uses the TST/TTT method described above, to find the shortest path option for the vehicle based on the algorithms described by [21, 17].

Primitive Number	Primitive Description	Velocity (ft/sec)	Flight Path Angle (deg)	Turn Rate (deg/sec)	Switch to Primitive Number
1	Straight	393.7	0	0	2,3
2	Turn Right	393.7	0	13.269	1,3
3	Turn Left	393.7	0	13.269	1,2

Table 4.1: Maneuver Primitives for AFRL Path Planner for Capacitated Transshipment Network Research

Currently, the TST/TTT path planning algorithm uses the maneuver primitives shown in Table 4.1. These three primitives are used to develop the shortest path based on initial/final position, initial/final headings, and sensor standoff distance. During task allocation, the vehicle's path planner is required to calculate vehicle trajectories to the target to return the shortest path to the target for each set of initial conditions. Initial conditions change to fit each task. For example, during the Classification task, the LOCAAS vehicle must approach the target using four "cardinal" headings to maximize the probability of target identification (as illustrated in Figure 4-5), while the BDA task does not specify the final vehicle approach heading to the target.

Additionally, the trajectories associated with each cost are not stored by the planner. Therefore, after the tasks are allocated to each vehicle, the vehicles must recalculate their path to the target. Finally, since the current system uses kinematic

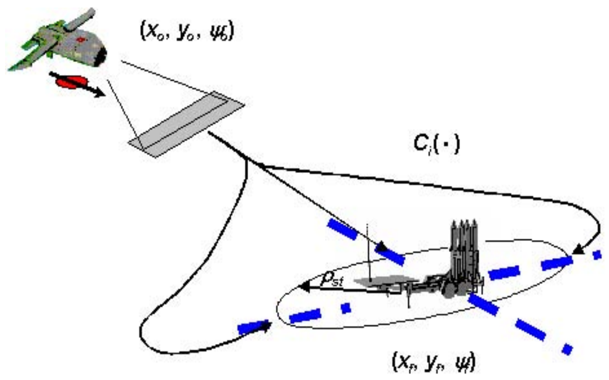


Figure 4-5: Target Cardinal Headings (dashed lines) for Munition to Maximize Probability of Target Identification

geometry to generate the vehicle's cost to the target, when the vehicle enters regions where path discontinuities (as described by Walker [21]), the vehicle maybe forced to elongate its path unnecessarily because it calculates trajectories to the established cardinal headings for target identification reasons.

With the current technique, three issues may arise:

Issue 1: Sensor Standoff Considerations

Since the sensor footprint is rather large, the vehicle can approach the target from angels within 5° to 10° of the cardinal heading and still achieve high probabilities for target recognition. For this reason, it would be helpful to survey more approach angles for each target before deciding on the best approach path.

Issue 2: Additional Primitives/Performance

Currently, the path planner is required to calculate the cost to the target. Therefore, if additional primitives are considered for each task, more computation time will be required.

Issue 3: Additional Approach Angles

If the vehicle is required to consider more approach angles to avoid path discontinuities, more computation time will be required.

4.5.2 Task Allocation with Coordination Function

To help resolve this problem, the vehicle's guidance value function can be used to quickly provide the cost-to-goal information required by the task allocation algorithm. For the LOCAAS problem, the MA is composed of three trims (Straight, Turn Right, Turn Left) and six maneuvers as described in Table 4.1. Since the maneuvers are direct transitions (i.e. we are not considering vehicle dynamics in the transitions between trim states), we can calculate a Coordination Function for Task Assignment $C(x)$ using Eq. 4.1:

$$C(x) = \arg \min_{\mathrm{u} \in \mathrm{U(x)}} \mathbf{p}(\mathbf{x}, \mathbf{u}, \mathbf{t}) \tag{4.1}$$

where $p(\cdot)$ represents the path planning algorithm which depends on the state x, the commands u. This function can be calculated directly off-line by saving the best cost for each initial condition calculated by the path planning algorithm.

There are two major benefits in using the coordination function:

Advantage 1: Calculation Time Reduction

As the number of maneuver primitives increase, the time to calculate the vehicle's cost-to-goal *does not change.* Therefore, the task assignment algorithm can determine the vehicle's final destination without calculating the path.

Advantage 2: Footprint Considerations

With this function, the task allocation can probe for the best costs using a lookup function. Therefore, the algorithm will be able to consider more trajectories in less computation time (as illustrated in Figure 4-6

This coordination function is called by the Task Assignment algorithm to provide the cost-to-goal information for a given initial/final position, initial/final headings, and sensor standoff distance.

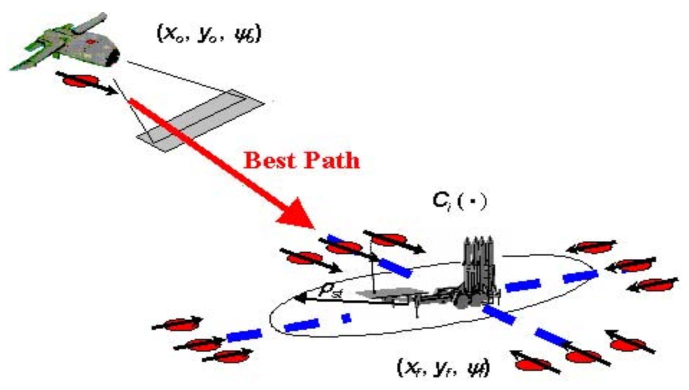


Figure 4-6: Multiple Cost Consideration (red circles) without Calculating Path

4.5.3 Kinematic Planner and Coordination Function Comparison

To test the accuracy of the coordination function for the LOCAAS scenario, we compared the values saved in the coordination function against the costs calculated by the Kinematic Planner (as shown in Table 4.2).

Trial Number	$\Delta\Psi_R$ (deg)	Δx (ft)	Δy (ft)	Kinematic Pln. (sec)	Coord. Fnct. (sec)	Δt_{coast} (sec)
1	0	100	200	27.699	27.699	0.000
4	135	100	200	44.717	44.717	0.000
7	-90	100	200	40.223	40.223	0.001
10	45	1500	1000	31.628	31.561	0.067
13	180	1500	1000	47.884	47.885	-0.001
16	-45	1500	1000	30.721	30.720	0.001
18	45	2500	100	33.271	32.636	0.636
19	90	2500	100	32.080	32.080	0.000
22	-135	2500	100	29.395	29.400	-0.004
27	0	-2500	1000	6.980	6.982	-0.002
29	180	-2500	-1000	49.500	49.501	-0.001

Table 4.2: Results Comparing the Cost-to-Goal generated by the Kinematic Planner Algorithm and the Coordination Function for the Capacitated Transshipment Network Research Path Planner

Initial comparisons showed that the values calculated using the function were very close to the values calculated by the planner function. Larger differences in cost were observed around path discontinuities for some destination points a radius of 3400 ft

(2*Radius of Curvature) from the goal, as described by Walker [21]. The calculation time for cost calculations was minimal.

From this demonstration, we have shown that the guidance value function can be used in task assignment. This demonstration provides a link between mission planning and mission execution. Since the guidance value function encapsulates such notions as fuel consumption, mission execution time, and furtivity constraints, the task allocation algorithm can use values from the guidance value function to optimize vehicle tasks using values representative of the vehicle's current state.

Chapter 5

Conclusion

In this thesis, we have shown that the reconfiguration strategy described in Eq. (2.2) can be used to plan trajectories for the vehicle after a change in the MA structure. This observation provides us with a foundation for understanding the robustness of the nominal value function. By properly developing the maneuver automaton (i.e. satisfying performance, controllability and connectivity requirements), the robust value function can be used or slightly modified to create a tool for finding a good estimate for calculating the reconfigured system's path. We have also shown how adding and removing performance affects the nominal value function. When we add performance to the system, the cost-to-go never increases. Likewise, when we remove performance from the system, the optimal cost-to-go is never reduced. These results are valuable for understanding the effect of vehicle reconfiguration on the guidance value function.

We discussed the trade-offs between system performance and robustness. By adding grid points, we can reduce the interpolation errors encountered when calculating trajectories, thus improving the performance of the system. We proved that under the reconfiguration strategy, if the system can find a sequence of motion primitives or controls that move the system to a region of the value function that will guide the vehicle to the goal state under the reconfigured MA, then the nominal value function can be used to plan trajectories from the initial hybrid state. We demonstrated the risks in using the reconfiguration strategy without a strategy that determines if there is a feasible trajectory to the goal state after a decision is made under the reconfiguration strategy. We showed that by combining the reconfiguration strategy with a “look ahead” technique, the cost-to-goal can be reduced while ensuring that a feasible trajectory to the goal exists.

We discussed some of the issues encountered when we implement the value function-based guidance systems. We described some of the techniques that can be used to implement reconfiguration strategies and the use of feedback to reduce errors in the trajectory of the vehicle. Finally, we showed that this guidance value function can also be used in task planning since the cost-to-go in the value function is representative of the time for the system to reach the goal state.

5.1 Recommendations for Future Work

Currently, we are exploring the theoretical issues relating to the value function’s robustness to perturbations and removal / additions of automaton primitives. Some of these issues include developing better proofs for the stability and robustness of the reconfiguration strategy for different changes in the plant model and maneuver automaton. Specifically, we are examining these issues for more complex models (i.e. continuous non-linear models without using the maneuver automaton). To address these issues, we are building our understanding of they complex systems by using simplified low-order system models to develop a stronger understanding of these issues.

We are also working to include feedback and uncertainty models for each maneuver primitive in our current navigation model to improve the system’s trajectory planning and reduce the time required to calculate the value function for the vehicle. Since most of the planning in this thesis is done open-loop (i.e. after the path is planned for the vehicle, the system commands the vehicle to the desired waypoint using pre-determined command inputs), we are working to include trajectory and state tracking to improve vehicle performance during highly-complex maneuvers. This feedback command allows the system to use proportional navigation techniques in a ”final region” of the trajectory. In addition, by including uncertainty in the trajectory planning, the vehicle may choose to fly paths using maneuver primitives that are "well-defined", thus improving the vehicle's trajectory planning capability. Currently, we are investigating how feedback commands and uncertainty affect the reconfiguration of these value function based guidance systems.

Next, we are investigating how parameters in the development of the value function affect the path planning capabilities of the vehicle. In Chapter 3 and 4, we showed that the number of grid points influences the vehicle's trajectory. In addition, we would like to investigate the sensitivity of the vehicle's trajectory with respect to the grid size for different grid variables from a theoretical and practical perspective.

Another important area for research is examining how the lateral/directional value function for a vehicle is affected by changes in altitude and other environmental conditions (i.e. wind speed). Though these effects can be considered as uncertainties in the plant model, one may be able to reconfigure the original system based on the vehicle's position and current environmental conditions and use the nominal value function to plan precision paths.

This research could also be examined in the context of multiple vehicle reconfiguration. Since each vehicle in a vehicle formation may have a value function-based guidance system, how can we calculated an optimal path for the group? In addition, how does the group compensate for a change or reconfiguration in one vehicle? If we model the group as a single entity and develop a guidance value function for this entity, how can this nominal guidance value function used when there are changes in this entity or any of the parts of this entity?

From these theoretical and practical issues, we hope to have a more complete framework for using this technique in a practical environment.

Appendix A

Tables: Maneuver Automaton Listings for Case Study Examples

Trim	Velocity (ft/sec)	Maneuver to Trim Number	Trim	Velocity (ft/sec)	Maneuver to Trim Number
1	-10.0000	{2, ..., 23}	2	-4.6416	{1, 3, ..., 23}
3	-3.0000	{1, 2, 4, ..., 23}	4	-2.1544	{1, ..., 3, 5, ..., 23}
5	-1.0000	{1, ..., 4, 6, ..., 23}	6	-0.4642	{1, ..., 5, 7, ..., 23}
7	-0.2154	{1, ..., 6, 8, ..., 23}	8	-0.1000	{1, ..., 7, 9, ..., 23}
9	-0.0464	{1, ..., 8, 10, ..., 23}	10	-0.0215	{1, ..., 9, 11, ..., 23}
11	-0.0100	{1, ..., 10, 12, ..., 23}	12	0.0000	{1, ..., 11, 13, ..., 23}
13	0.0100	{1, ..., 12, 14, ..., 23}	14	0.0215	{1, ..., 13, 15, ..., 23}
15	0.0464	{1, ..., 14, 16, ..., 23}	16	0.1000	{1, ..., 15, 17, ..., 23}
17	0.2154	{1, ..., 16, 18, ..., 23}	18	0.4642	{1, ..., 17, 19, ..., 23}
19	1.0000	{1, ..., 18, 20, ..., 23}	20	2.1544	{1, ..., 19, 21, ..., 23}
21	3.0000	{1, ..., 20, 22, 23}	22	4.6416	{1, ..., 21, 23}
23	10.0000	{1, ..., 22}			

Table A.1: Nominal MA for Double Integrator Reconfiguration Example

Trim	Velocity (ft/sec)	Maneuver to Trim Number	Trim	Velocity (ft/sec)	Maneuver to Trim Number
1	Removed	None	2	Removed	None
3	Removed	None	4	Removed	None
5	-1.0000	{6, ..., 23}	6	-0.4642	{5, 7, ..., 23}
7	-0.2154	{5, 6, 8, ..., 23}	8	-0.1000	{5, 6, 7, 9, ..., 23}
9	-0.0464	{5, ..., 8, 10, ..., 23}	10	-0.0215	{5, ..., 9, 11, ..., 23}
11	-0.0100	{5, ..., 10, 12, ..., 23}	12	0.0000	{5, ..., 11, 13, ..., 23}
13	0.0100	{5, ..., 12, 14, ..., 23}	14	0.0215	{5, ..., 13, 15, ..., 23}
15	0.0464	{5, ..., 14, 16, ..., 23}	16	0.1000	{5, ..., 15, 17, ..., 23}
17	0.2154	{5, ..., 16, 18, ..., 23}	18	0.4642	{5, ..., 17, 19, ..., 23}
19	1.0000	{5, ..., 18, 20, ..., 23}	20	2.1544	{5, ..., 19, 21, ..., 23}
21	3.0000	{5, ..., 20, 22, 23}	22	4.6416	{5, ..., 21, 23}
23	10.0000	{5, ..., 22}			

Table A.2: Reconfigured MA for Double Integrator Test 1: Trim Removal

Trim	Velocity (ft/sec)	Maneuver to Trim Number	Trim	Velocity (ft/sec)	Maneuver to Trim Number
1	-10.0000	{2, ..., 25}	2	-4.6416	{1, 3, ..., 25}
3	-3.4500	{1, 2, 4, ..., 25}	4	-3.0000	{1, ..., 3, 5, ..., 25}
5	-2.1544	{1, ..., 4, 6, ..., 25}	6	-1.0000	{1, ..., 5, 7, ..., 25}
7	-0.4642	{1, ..., 6, 8, ..., 25}	8	-0.2154	{1, ..., 7, 9, ..., 25}
9	-0.1000	{1, ..., 8, 10, ..., 25}	10	-0.0464	{1, ..., 9, 11, ..., 25}
11	-0.0215	{1, ..., 10, 12, ..., 25}	12	-0.0100	{1, ..., 11, 13, ..., 25}
13	0.0000	{1, ..., 12, 14, ..., 25}	14	0.0100	{1, ..., 13, 15, ..., 25}
15	0.0215	{1, ..., 14, 16, ..., 25}	16	0.0464	{1, ..., 15, 17, ..., 25}
17	0.1000	{1, ..., 16, 18, ..., 25}	18	0.2154	{1, ..., 17, 19, ..., 25}
19	0.4642	{1, ..., 18, 20, ..., 25}	20	1.0000	{1, ..., 19, 21, ..., 25}
21	2.1544	{1, ..., 20, 22, ..., 25}	22	3.0000	{1, ..., 21, 23, 24, 25}
23	3.4500	{1, ..., 22, 24, 25}	24	4.6416	{1, ..., 23, 25}
25	10.0000	{1, ..., 24}			

Table A.3: Reconfigured MA for Double Integrator Test 2: Trim Addition

Trim	Velocity (ft/sec)	Maneuver to Trim Number	Trim	Velocity (ft/sec)	Maneuver to Trim Number
1	-10.0000	{2, ..., 23}	2	-4.6416	{1, 3, ..., 23}
3	-3.0000	{1, 2, 4, ..., 23}	4	-2.1544	{1, ..., 3, 5, ..., 23}
5	-1.0000	{1, ..., 4, 6, ..., 23}	6	-0.4642	{1, ..., 5, 7, ..., 23}
7	-0.2154	{1, ..., 6, 8, ..., 23}	8	-0.1000	{1, ..., 7, 9, ..., 23}
9	-0.0464	{1, ..., 8, 10, ..., 23}	10	-0.0215	{1, ..., 9, 11, ..., 23}
11	-0.0100	{1, ..., 10, 12, ..., 23}	12	0.0000	{1, ..., 11, 13, ..., 23}
13	0.0100	{1, ..., 12, 14, ..., 23}	14	0.0215	{1, ..., 13, 15, ..., 23}
15	0.0464	{1, ..., 14, 16, ..., 23}	16	0.1000	{1, ..., 15, 17, ..., 23}
17	0.2154	{1, ..., 16, 18, ..., 23}	18	0.4642	{1, ..., 17, 19, ..., 23}
19	1.0000	{1, ..., 18, 20, ..., 23}	20	2.1544	{1, ..., 19, 21, ..., 23}
21	3.0000	{1, ..., 20, 22, 23}	22	4.6416	{1, ..., 21, 23}
23	10.0000	{1, ..., 22}			

Table A.4: Reconfigured MA for Double Integrator Test 3: Changes in Maneuver Command Limits

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,3,4,5,6,7,8,9
2	300	-5	1,3,4,5,6,7,8,9
3	400	-5	1,2,4,5,6,7,9
4	300	0	1,2,3,5,6,7,8,9
5	400	0	1,2,3,4,6,7,8,9
6	500	0	1,2,3,4,5,7,8,9
7	300	+5	1,2,3,4,5,6,8,9
8	400	+5	1,2,4,5,6,7,9
9	300	+10	1,2,3,4,5,6,7,8

Table A.5: Nominal 9-Trim Longitudinal MA - Case Study II

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,3,4,5,6,7,8
2	300	-5	1,3,4,5,6,7,8
3	400	-5	1,2,4,5,6,7
4	300	0	1,2,3,5,6,7,8
5	400	0	1,2,3,4,6,7,8
6	500	0	1,2,3,4,5,7,8
7	300	+5	1,2,3,4,5,6,8
8	400	+5	1,2,4,5,6,7
9	300	+10	Not Achievable

Table A.6: Reconfigured 9-Trim Longitudinal MA - Case Study II: Test 1 - Part 1

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,3,4,5,6,7,8
2	300	-5	1,3,4,5,6,7,8
3	400	-5	1,2,4,5,6,8
4	300	0	1,2,3,5,6,7,8
5	400	0	1,2,3,4,6,7,8
6	500	0	Not Achievable
7	300	+5	1,2,3,4,5,7,8
8	400	+5	1,2,4,5,6,8
9	300	+10	1,2,3,4,5,6,7

Table A.7: Reconfigured 9-Trim Longitudinal MA - Case Study II: Test 1 - Part 2

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,4
2	300	-5	1,3,4
3	400	-5	2,4,5
4	300	0	2,3,5,7,8
5	400	0	3,4,6,8
6	500	0	5
7	300	+5	4,8,9
8	400	+5	4,5,7
9	300	+10	4,7

Table A.8: Reconfigured 9-Trim Longitudinal MA - Case Study II: Test 2

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,4
2	300	-5	1,3,4
3	400	-5	2,4,5
4	300	0	2,3,5,7,8
5	400	0	3,4,6,8
6	500	0	5
7	300	+5	4,8,9
8	400	+5	4,5,7
9	300	+10	4,7

Table A.9: Nominal 9-Trim Longitudinal MA - Case Study II: Test 3

Trim Number	Velocity (ft/sec)	Flight Path Angle (deg)	Maneuver to Trim Number
1	300	-10	2,4
2	300	-5	1,3,4
3	400	-5	2,4,5,6
4	300	0	2,3,5,7,8
5	400	0	3,4,6,8
6	500	0	3,4,5,8
7	300	+5	4,8,9
8	400	+5	4,5,6,7
9	300	+10	4,7

Table A.10: Reconfigured 9-Trim Longitudinal MA - Case Study II: Test 3

Trim Number	Velocity (ft/sec)	Turn Rate (deg)	Maneuver to Trim Number
1	50	-20	1,2
2	50	0	1,3
3	50	+20	2,3

Table A.11: Nominal 3-Trim Lateral/Directional MA - Case Study III

Trim Number	Velocity (ft/sec)	Turn Rate (deg)	Maneuver to Trim Number
1	50	-20	1,2
2	50	0	1,3
3	50	+20	Not Achievable

Table A.12: Reconfigured 3-Trim Lateral/Directional MA - Case Study III: Test 1

Bibliography

[1] AFRL Air Vehicles Directorate. *Control Science Center of Excellence Brochure.* 2002.

[2] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, Mass., 2000.

[3] M. S. Branicky. Stability of switched and hybrid systems. pages 3498–3503. In Proceedings of the 34th IEEE Conference on Decision and Control, 1994.

[4] M. S. Branicky, T. A. Johansen, I. Petersen, and E. Frazzoli. Maximum likelihood identification of a rotary-wing RPV simulation model from flight-test data. volume 2, pages 1840–1845. In Proceedings of the 39th IEEE Conference on Decision and Control, 2000.

[5] M. Dahleh, M. A. Dahleh, and G. Verghese. *Lectures on Dynamic Systems and Control.* Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2001.

[6] L. E. Dubins. On curve of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 97:497–516, 1957.

[7] N. Faiz, S. K. Agrawal, and R. M. Murray. Trajectory planning of differentially flat systems with dynamics and inequalities. *AIAA Journal of Guidance, Control, and Dynamics*, 24(2):219–227, 2002.

[8] J. M. Fowler. *Coupled Task Planning for Multiple Unmanned Air Vehicles.* Technical Report, AFRL/VACA WPAFB, Dayton, Ohio, 2001.

[9] E. Frazzoli. *Robust Hybrid Control for Autonomous Vehicle Motion Planning.* PhD Thesis, MIT, 2001.

[10] E. Frazzoli, M.A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[11] D. G. Luenberger. *Introduction to Dynamic Systems: Theory, Models, and Applications.* John Wiley and Sons, New York, 1979.

[12] B. Mettler, M. Valenti, T. Schouwenaars, and E. Feron. Rotorcraft motion planning for agile maneuvering using a maneuver automaton. Montreal, Canada, June 2002. 58th Forum of AHS International.

[13] S. J. Rasmussen and P. R. Chandler. Multiuav: A multiple uav simulation for investigaton of cooperative control. San Diego, California, December 2002. Winter Simulation Conference.

[14] M. Safonov and M. Athans. Gain and phase margin for multiloop LQG regulators. *IEEE Transactions on Automatic Control*, 22(2):690–696, 1977.

[15] J.D. Schierman, J.R. Hull, and D.G. Ward. Adaptive guidance with trajectory reshaping for reusable launch vehicles. Monterey, California, August 2002. AIAA Guidance, Navigation, and Control Conference and Exhibit.

[16] C. Schumacher, P. Chandler, and S. Rasmussen. Task allocation for wide area search munitions via iterative network flow. Monterey, California, August 2002. AIAA Guidance, Navigation, and Control Conference and Exhibit.

[17] D. Swaroop. *A Method of Cooperative Classification and Attack for LOCAAS Vehicles.* Technical Report, AFRL/VACA WPAFB, Dayton, Ohio, 2000.

[18] M. Valenti. *Trajectory Reconfiguration and Task Assignment for an Unmanned Aircraft.* Technical Report, AFRL/VACA WPAFB, Dayton, Ohio, 2002.

[19] M. Valenti, B. Mettler, T. Schouwenaars, E. Feron, and J. Paduano. Trajectory reconfiguration for an unmanned aircraft. Monterey, California, August 2002. AIAA Guidance, Navigation, and Control Conference and Exhibit.

[20] A. J. Verma and J. L. Junkins. Trajectory generation for transition from VTOL to wing-borne flight using inverse dynamics. Reno, Nevada, January 2000. 38th AIAA Aerospace Sciences Meeting and Exhibit.

[21] D. H. Walker. *Geometric Path Planning with Length Constraints.* Technical Report, AFRL/VACA WPAFB, Dayton, Ohio, 2002.

[22] K. A. Wise, J. S. Brinker, A. J. Calise, D. F. Enns, M. R. Elgersma, and P. Voulgaris. Direct adaptive reconfigurable flight control for a tailless advanced fighter aircraft. *International Journal of Robust and Nonlinear Control*, 9:999–1012, 1999.

[23] K. Zhou. *Essentials of Robust Control.* Prentice Hall, Upper Saddle River, New Jersey, 1998.