AN ABSTRACT OF THE DISSERTATION OF

Jason A. Kyle for the degree of Doctor of Philosophy in Mechanical Engineering presented on November 17, 2006.

Title: Optimal Soaring by a Small Autonomous Glider.

Abstract approved:

Mark F. Costello

Extending the flight time of an autonomous unmanned air vehicle by soaring is considered. A suboptimal controller is developed and successful static soaring is demonstrated with a 6 degree of freedom glider model. Altitude gain rates of between ¼ and ½ m/s are achieved with this simple implementation.

A hybrid optimal trajectory generation algorithm is developed and used to find optimal closed cycles in typical wind conditions using a point mass model. The algorithm is shown to be robust to a poor initial guess, with computational performance comparable to a common direct shooting algorithm.

A receding horizon optimal controller strategy is investigated for the problem of autonomous soaring. An efficient Riccatti recursion algorithm is used to determine the next step in the Newton Iteration of the Non-Linear optimization problem. A real time strategy for optimal soaring is developed and shown to perform very well for a point mass model, resulting in repeatable trajectories with significant altitude gain. Sensitivity to errors including wind model errors is investigated. The real time algorithm was found to be insensitive to reasonable errors.

Optimal Soaring by a Small Autonomous Glider

by

Jason A. Kyle

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of

the requirements for the

degree of

Doctor of Philosophy

Presented November 17, 2006

Commencement June 2007

TABLE OF CONTENTS

Page

GENERAL INTRODUCTION 2

ATMOSPHERIC WIND ENERGY EXTRACTION BY A SMALL AUTONOMOUS GLIDER 4

ABSTRACT 5

INTRODUCTION 6

GLIDER MATHEMATICAL MODEL 8

NON-LINEAR MODEL PREDICTIVE CONTROL LAW 11

Autonomous Glider Control Law 16

EXAMPLE RESULTS 17

CONCLUSIONS 26

REFERENCES 27

ATMOSPHERIC WIND ENERGY EXTRACTION BY A SMALL AUTONOMOUS GLIDER 28

A HYBRID TRAJECTORY OPTIMIZATION METHOD FOR WIND ENERGY EXTRACTION 29

ABSTRACT 29

NOMENCLATURE 30

INTRODUCTION 33

OPTIMIZATION PROBLEM DEFINITION 35

CONVERSION TO PARAMETER OPTIMIZATION PROBLEM 36

PARTICLE SWARM OPTIMIZATION 39

TABLE OF CONTENTS (Continued)

Page

DIRECT SHOOTING 42

HYBRID SOLVER 45

GLIDER MODEL 47

WIND STRUCTURE MODELS 48

COMPARISON RESULTS 51

CONCLUSIONS 59

REFERENCES 60

OPTIMAL RECEDING HORIZON MODEL PREDICTIVE CONTROL FOR REAL TIME AUTONOMOUS SOARING 63

ABSTRACT 63

NOMENCLATURE 64

INTRODUCTION 67

RECEDING HORIZON NON-LINEAR MODEL PREDICTIVE CONTROL 71

Sequential Quadratic Programming Algorithm 72

State Inequality Constraints 78

Receding Horizon Implementation 79

GLIDER MODEL 80

WIND STRUCTURE MODELS 83

EXAMPLE RESULTS 86

SENSITIVITY INVESTIGATION 95

TABLE OF CONTENTS (Continued)

Page

CONCLUSIONS 98

REFERENCES 99

GENERAL CONCLUSIONS 101

LIST OF FIGURES

Figure Page

1-1: Thermal Velocity Profile 10

1-2: Glider Position and Thermal Profile Contour 18

1-3: Glider Position Time History 19

1-4: Glider Altitude and Thermal Velocity History 19

1-5: Control Flap Deflection 20

1-6: Desired and Actual Heading Angle 20

1-7: Pitch Angle Time History 21

1-8: Glider Bank Angle 21

1-9: Glider Body Frame x Velocity 22

1-10: Glider Body Frame y Velocity 22

1-11: Glider Body Frame z Velocity 23

2-1: Multiple Low Order Polynomial Control Parameterization 37

2-2: Structure of Hybrid Optimal Search Method (HS) 45

2-3: Thermal Model Shape 48

2-4: Wind Gradient Profile 50

2-5: Thermal Model Profile 51

2-6: Single Thermal Scenario Optimal Closed Trajectory 53

2-7: HS Solution Altitude and Wind Velocity Time History 53

2-8: DS Solution Altitude and Wind Velocity Time History 54

2-9: Single Thermal Scenario Bank Angle Time History 54

2-10: Thermal and Wind Gradient Scenario Optimal Trajectory 56

LIST OF FIGURES

Figure Page

2-11: Thermal and Wind Gradient Scenario Altitude Time History.......................... 57

2-12: Thermal and Wind Gradient Scenario Bank Angle Time History 57

2-13: Thermal and Wind Gradient Scenario Lift Coefficient Time History.............. 58

3-1: Boundary Function for a Range of C Values... 78

3-3: Thermal Model Profile 84

3-4: Wind Gradient Model Profile 85

3-5: Single Thermal Scenario Trajectory 86

3-6: Single Thermal Scenario Altitude and Thermal Velocity Time History............ 87

3-7: Single Thermal Scenario Airspeed Time History.. 87

3-8: Single Thermal Scenario Energy Gain Time History... 88

3-9: Single Thermal Scenario Bank Angle Time History.. 88

3-10: Single Thermal Scenario Lift Coefficient Time History 89

3-11: Oval Thermal Scenario Trajectory 90

3-12: Oval Thermal Scenario Altitude and Thermal Velocity Time History 90

3-13: Thermal and Wind Gradient Trajectory 91

3-14: Thermal and Wind Gradient Scenario Altitude and Thermal Wind Velocity

Time History 92

3-15: Thermal and Wind Gradient Scenario Airspeed Time History 92

3-16: Thermal and Wind Gradient Scenario Energy Gain Time History 93

LIST OF TABLES

Table Page

1-1: Wind Model Parameters 23

1-2: Glider Physical Parameters 24

1-3: Glider Aerodynamic Parameters 24

1-4: Controller Parameters 24

1-5: Trade Study Results 25

1-6: Physical Properties Trade Study 25

2-1: Optimization Problem 1 (OP1) 35

2-2: Optimization Problem 2 (OP2) 36

2-3: Optimization Problem 3 (OP3) 38

2-4: Particle Swarm Optimization Algorithm (PSO) 40

2-5: Direct Shooting Algorithm (DS) 43

2-6: Hybrid Optimal Search Algorithm (HS) 46

2-7: Glider Model Parameters 47

2-8: Thermal Model Parameters 51

2-9: Method Performance Comparison 55

2-10: HS Algorithm Parameters 55

2-11: Wind Gradient Model Parameters 55

2-12: Thermal and Wind Gradient Scenario Algorithm Performance 58

2-13: PSO Algorithm Parameters 59

3-1: Optimization Problem 1 (OP1) 71

LIST OF TABLES (Continued)

Table | Page

3-2: Optimization Problem 2 (OP2) 72

3-3: Quadratic Programming Problem 1 (QP1) 75

3-4: SQP Optimal Control Algorithm 77

3-5: Glider Model Parameters 82

3-6: Round Thermal Model Parameters 93

3-7: Oval Thermal Model Parameters 93

3-8: Horizontal Wind Gradient Model Parameters 94

3-9: Thermal Parameter Sensitivity Investigation 95

3-10: Wind Gradient Parameter Sensitivity Investigation 96

3-11: Glider Aerodynamic Parameter Sensitivity Investigation 96

3-12: Glider Parameter Sensitivity Investigation 97

OPTIMAL SOARING BY A SMALL AUTONOMOUS GLIDER

GENERAL INTRODUCTION

Large birds commonly use wind currents as a free energy source to remain aloft without expending energy through flapping. Columns of rising air called thermals routinely found in the lower atmosphere provide an excellent energy source. Altitude or potential energy can be increased simply by flying circles in the thermal as long as possible. Flight paths tend to be relatively static with nearly constant airspeed, bank, and pitch angles. For this reason, this type of soaring is called "Static Soaring." "Semi Dynamic" soaring trajectories can be more efficient by extracting energy from the vertical wind source as in "Static Soaring," and from the change in wind velocity over the flight path. Airspeed, bank, and pitch angles oscillate over the flight path. Although significant technological challenges must be solved, unmanned air vehicles could also benefit greatly from this free energy source.

One strategy for optimally extracting energy from wind structures involves determination of optimal repeatable trajectories for a range of scenarios off line. After the UAV identifies a suitable wind structure, an optimal closed cycle is selected by interpolation of the available solutions. The UAV can then execute the trajectory or use it to initialize an onboard optimization algorithm to refine the trajectory in flight. This strategy requires significant pre-flight work to build an extensive library of optimal closed trajectories, and large data storage space on the aircraft. With the dynamic nature of winds, the online algorithm must be able to update the trajectory more often than once a cycle. This work assumes the UAV has some knowledge of local wind structures.

The first work of this dissertation investigates the feasibility of gaining energy in thermals with a small UAV by simple static soaring. A six degree of freedom model with first order aerodynamics is used. A multiple input multiple output nonlinear suboptimal controller is developed for use with the 6DOF model. A typical scenario is investigated and the glider successfully gains altitude by static soaring.

In the second work of this dissertation a robust optimization algorithm is developed and used to determine optimal closed trajectories. The tool allows new scenarios to be investigated where the optimal solution and a good initial guess is unknown. Several common scenarios are investigated and performance of the algorithm with a direct shooting algorithm is compared. The solver is shown to be robust to initial guess while performing nearly as fast as direct shooting alone.

The third work in this dissertation investigates a new onboard strategy for the problem of autonomous soaring. A receding horizon optimal control algorithm is extended to the soaring problem. The main strategic difference is that a closed optimal solution is not required in the optimization problem. Initial states and controls are known and the algorithm selects the controls which result in the maximal energy gain over a fixed prediction horizon. This problem is relaxed from the closed cycle formulation, and can be solved efficiently for online use. Fast update rates are achieved resulting in a controller that is robust to un-modeled disturbances and modeling errors. Several common scenarios are investigated and the algorithm is shown to perform exceptionally well.

1 ATMOSPHERIC WIND ENERGY EXTRACTION BY A SMALL AUTONOMOUS GLIDER

Jason Kyle, Katie Evans, and Mark Costello

AIAA Atmospheric Flight Mechanics Conference and Exhibit
San Francisco, CA. Aug. 15-18, 2005

ATMOSPHERIC WIND ENERGY EXTRACTION BY A SMALL AUTONOMOUS GLIDER

Jason Kyle[*] Katie Evans[♣] Mark Costello[†]

Department of Mechanical Engineering

Oregon State University

Corvallis, Oregon

ABSTRACT

Extracting energy from thermal wind conditions with a small autonomous air vehicle is considered. A non-linear model predictive controller is developed that embeds a standard glider model and tracks roll, pitch, and yaw angles. Given knowledge of the local wind structure, the flight control system increases the potential energy of the aircraft through autonomous soaring. A typical energy extracting trajectory is investigated through simulation.

[*] Graduate Research Assistant, Member AIAA.
[♣] Research Associate.
[†] Associate Professor, Member AIAA.

INTRODUCTION

Micro air robots are small, autonomous, intelligent aircraft designed to focus on a specific task. The range of applications envisioned for future micro air robots in both the civilian and military sectors is truly staggering. Micro air robots promise to change the world around us by providing unparalleled situation awareness and data gathering opportunities in a wide variety of scenarios. Micro air robots may be used by environmentalists for detailed wildlife monitoring and surveying tasks. Micro air robots could fly through factory smokestack emissions to measure released chemical concentrations. With gradient sensors and flight control system feedback, micro air robots could map the size and shape of hazardous clouds and provide real time tracking of their location. Forestry management could be aided by sending micro air robots into remote areas of a forest that are difficult to access to gather important forest health and growth data. Other applications include monitoring concentrations of chemical spills and measuring ammonia concentration in agriculture, to name just a couple.

Micro air robots could be used to assess situations too dangerous for direct human intervention. For example, after a natural or man made disaster micro air robots could maneuver through damaged buildings looking for survivors. Fighting forest fires could be enhanced by detailed information on the progress of a fire obtained by micro air robots. Other micro air robot applications include situations where explosive devices are planted in structures and determining the health and location of hostages.

In urban areas, micro air robots can be used by law enforcement for pursuing criminals in a safe yet close manner. Micro air robots could provide flexible traffic monitoring. In rural areas, micro air robots could be employed to efficiently monitor large expanses of land for applications such as border patrol and power line inspection. Swarms of micro air robots could be used in search and rescue missions to rapidly search a large area.

A plethora of military applications exist for micro air robots. Perhaps the most obvious application is reconnaissance. Current operational concepts suggests that reconnaissance micro air robots have a range capability of about 10 km, endurance of up to an hour, speeds of 10 to 20 m/s, and be capable of real time day/night imagery. Micro air robots with these performance characteristics could look over the next hill in combat situations and also perform targeting missions. More aggressive micro air robots could tag targets to aid weapons with improved target recognition in the terminal guidance phase. Groups of micro air robots could be used to seed a future battlefield with sensors. Groups of micro air robots could also be used to form a communication network where each individual micro air robot acts as a relay.

The potential of micro air robots is astonishing, yet significant technical obstacles must be overcome to realize this potential. The Achilles heel of micro air robots is power required for mobility. Micro air robots consume a significant amount of power just to remain aloft. When considering practical micro air robot configurations that carry sensors, power requirement problems become more acute. These power requirements curtail the feasibility of micro air robots for many of the amazing potential missions mentioned above. To remedy this situation, many research groups are actively engaged in research and development on small, low weight, high power output propulsion technologies. An alternate and complementary concept for powering micro air robots is to harvest energy from the environments in which they fly. In straight and level aircraft flight, atmospheric wind updrafts rotate the relative aerodynamic velocity vector downward, causing drag to point aft and slightly upward and lift to point up and slightly forward. When the atmospheric wind updraft is sufficiently large, straight and level flight and even climbing flight is possible without power. Conventional sailplane soaring is founded on this type of atmospheric wind energy extraction. The research detailed in this paper develops a nonlinear model predictive control law that tracks energy harvesting trajectories autonomously. Performance of the control law is investigated through dynamic simulation of an exemplar micro air glider.

GLIDER MATHEMATICAL MODEL

The glider is modeled as a rigid body, and undergoes three-dimensional motion described by three inertial position coordinates and three Euler angles. Hence, 12 state variables are required to describe motion of the glider at a given instance in time. The glider mathematical model is developed for a general fixed wing air vehicle. However, the configuration used in the results section employs a polyhedral main wing with no flaps, and a v-tail with standard trailing edge flaps.

In the equations that follow the ground frame is assumed to be a satisfactory reference frame. The body frame is defined by the standard aerospace rotation sequence, where the matrix $\lfloor T_{B/I} \rfloor$ relates the body frame to the Inertial frame given in Equation (1).

$$\left[T_{B/I}\right] = \begin{bmatrix} c_{\psi_B} c_{\theta_B} & s_{\phi_B} s_{\theta_B} c_{\psi_B} - c_{\phi_B} s_{\psi_B} & c_{\phi_B} s_{\theta_B} c_{\psi_B} + s_{\phi_B} s_{\psi_B} \\ s_{\psi_B} c_{\theta_B} & c_{\psi_B} c_{\phi_B} + s_{\psi_B} s_{\theta_B} s_{\phi_B} & c_{\phi_B} s_{\theta_B} s_{\psi_B} - s_{\phi_B} c_{\psi_B} \\ -s_{\theta_B} & s_{\phi_B} c_{\theta_B} & c_{\theta_B} c_{\phi_B} \end{bmatrix} \tag{1}$$

The common shorthand for trigonometric functions is used throughout the paper (sin(b) = s_b). The kinematic translational and rotational differential equations of motion are given by Equations (2) and (3).

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} = \left[T_{B/I}\right] \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \tag{2}$$

$$\begin{Bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{Bmatrix} = \begin{bmatrix} 0 & s_{\phi_B}/c_{\theta_B} & c_{\phi}/c_{\theta_B} \\ 0 & c_{\phi_B} & -s_{\phi_B} \\ 1 & t_{\theta_B} s_{\phi_B} & t_{\theta_B} c_{\phi_B} \end{bmatrix} \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} = \left[K_{RM}\right] \vec{\omega} \tag{3}$$

The velocity (u,v,w) and angular velocity (p,q,r) states are defined in the body reference frame. The dynamic differential equations of motion are written with respect to the body reference frame and are given by Equations (4) and (5).

$$\begin{Bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{Bmatrix} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} + (1/m) \begin{Bmatrix} U_A \\ V_A \\ W_A \end{Bmatrix} + (1/m) \begin{Bmatrix} U_C \\ V_C \\ W_C \end{Bmatrix} \tag{4}$$

$$\begin{Bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{Bmatrix} = [I]^{-1} \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} [I] \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} + [I]^{-1} \begin{Bmatrix} L_A \\ M_A \\ N_A \end{Bmatrix} + [I]^{-1} \begin{Bmatrix} L_C \\ M_C \\ N_C \end{Bmatrix} \tag{5}$$

The vector $\{X_A\ Y_A\ Z_A\}$ is the aerodynamic forces acting on the airplane, excluding the effects of the control inputs. The vector $\{X_C\ Y_C\ Z_C\}$ is the aerodynamic forces acting on the airplane due to wing flap deflections. These are modeled as the following lumped parameter effects:

$$\begin{Bmatrix} X_A \\ Y_A \\ Z_A \end{Bmatrix} = PV^2 \begin{Bmatrix} (C_{xo} + C_{xa}\alpha + (c/V)C_{xq}q) \\ (C_{yo} + C_{yb}\beta + (b/2V)C_{yp}p + (b/2V)C_{yr}r) \\ (C_{zo} + C_{za}\alpha + (c/V)C_{zq}q) \end{Bmatrix} \tag{6}$$

$$\begin{Bmatrix} X_c \\ Y_c \\ Z_c \end{Bmatrix} = PV^2 \begin{Bmatrix} (C_{xde}\delta_e + C_{xdal}\delta_{al} + C_{xdar}\delta_{ar}) \\ (C_{ydal}\delta_{al} + C_{ydar}\delta_{ar}) \\ (C_{zde}\delta_e + C_{zdal}\delta_{al} + C_{zdar}\delta_{ar}) \end{Bmatrix} \tag{7}$$

It will be useful to note that the forces due to wing flap deflections, for most model gliders, are small in comparison to the forces due to other aerodynamic effects. Similarly, the aerodynamic moments are modeled as follows:

$$\begin{Bmatrix} L_A \\ M_A \\ N_A \end{Bmatrix} = PV^2 \begin{Bmatrix} (b/2)(C_{lo} + C_{lb}\beta + (b/2V)C_{lp}p + (b/2V)C_{lr}r) \\ c(C_{mo} + C_{ma}\alpha + (c/V)C_{mq}q) \\ (b/2)(C_{no} + C_{nb}\beta + (b/2V)C_{zp}p + (b/2V)C_{nr}r) \end{Bmatrix} = \vec{\mathrm{M}}_{AERO} \tag{8}$$

$$\begin{Bmatrix} L_c \\ M_c \\ N_c \end{Bmatrix} = PV^2 \begin{Bmatrix} (b/2)C_{lda}\delta_a \\ c(C_{me}\delta_e + C_{mf}\delta_f) \\ (b/2)C_{nda}\delta_a \end{Bmatrix} = PV^2 \begin{bmatrix} (b/2)C_{lda} & 0 & 0 \\ 0 & cC_{mf} & cC_{me} \\ (b/2)C_{nda} & 0 & 0 \end{bmatrix} \begin{Bmatrix} \delta_a \\ \delta_f \\ \delta_e \end{Bmatrix} = [\mathrm{M}_c]\vec{\delta} \tag{9}$$

where the following parameters from Equations (8) through (9) are defined as:

$$V^2 = \left((u - W_u)^2 + (v - W_v)^2 + (w - W_w)^2\right) \tag{10}$$

$$\alpha = a\tan((w - W_w)/(u - W_u)) \tag{11}$$

$$\beta = a\tan((v - W_v)/V^2) \tag{12}$$

$$P = (1/2)\rho S \tag{13}$$

The glider model uses three independent control surfaces elevator, aileron, and flaps. Wind is modeled in two parts, horizontal wind acts in the x-y plane and thermal wind acts in the vertical direction. Horizontal wind is described in Equation (14).

$$\begin{Bmatrix} u_w \\ v_w \end{Bmatrix} = V_w \begin{Bmatrix} c_\beta \\ s_\beta \end{Bmatrix} \tag{14}$$

where V_w is the average wind velocity, and the angle β determines the direction of the general wind. The thermal wind is described in Equation (15).

$$w_w = M\cos\left(\frac{\pi}{2r^2}D^2\right)e^{(fD)} : \left\{0, \quad D > 3r^2\right\} \tag{15}$$

where D is the distance from the thermal center, r is the thermal radius, f is the decay rate, and M is the maximum value of the thermal occurring at the thermal center. The condition described constrains the thermal to within three times the radius squared. Wind velocity is determined for the mass center of the glider. The thermal described by Equation (15) is characterized by a small area of sink surrounding the thermal core. An example thermal profile is depicted in Figure 1-1 for the parameters in Table 1-1.

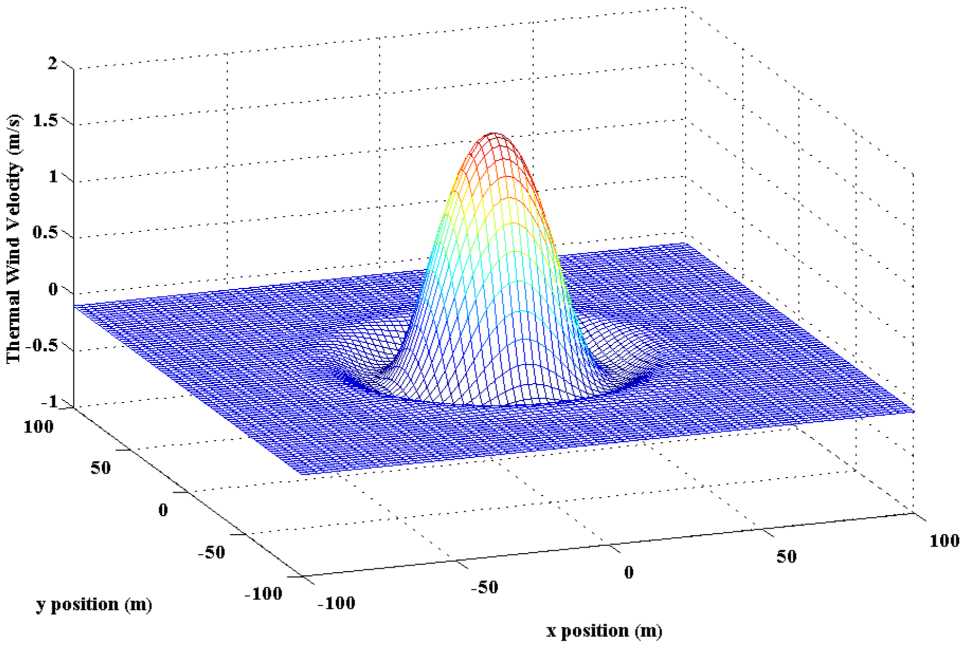

Figure 1-1: Thermal Velocity Profile

Body frame wind is required for the aerodynamic forces:

$$\vec{W} = \left[T_{B/I}\right]^T \begin{Bmatrix} u_w \\ v_w \\ w_w \end{Bmatrix} \tag{16}$$

NON-LINEAR MODEL PREDICTIVE CONTROL LAW

For simplicity, a control affine non-linear system is considered.

$$\dot{x} = f(x) + b(x)u \tag{17}$$

Assuming the functions f(x) and b(x) are sufficiently smooth, each output can be approximated by a Taylor series polynomial of order R_i.

$$y_i(t+\tau) = y(t) + \tau\frac{dy}{dt} + \frac{\tau^2}{2}\frac{d^2y}{dt^2} + \cdots + \frac{\tau^{R_i}}{R_i!}\frac{d^{R_i}y}{dt^{R_i}} \tag{18}$$

The control input does not appear for the first ρ_ι terms in the Taylor series polynomial. Derivatives of the control up to order $R_i - \rho_i$ appear in the expansion. This is written compactly as:

$$y_i(t+\tau) \cong \mathrm{T}_i \mathrm{Y}_i \tag{19}$$

$$\mathrm{T}_i \equiv \begin{bmatrix} 1 & \tau & \dfrac{\tau^2}{2!} & \cdots & \dfrac{\tau^{R_i}}{R_i!} \end{bmatrix} \tag{20}$$

$$\mathrm{Y}_i \equiv \begin{Bmatrix} y_i(t) \\ \dot{y}_i(t) \\ \vdots \\ \dfrac{d^{R_i} y_i(t)}{dt^{R_i}} \end{Bmatrix} \tag{21}$$

Consider the entire output vector, approximated by a Taylor series up to order R_i (possibly different orders for each output).

$$\mathrm{Y}(t+\tau) = \begin{bmatrix} \mathrm{T}_1 & & & 0 \\ & \mathrm{T}_2 & & \\ & & \ddots & \\ 0 & & & \mathrm{T}_M \end{bmatrix} \begin{Bmatrix} \mathrm{Y}_1 \\ \mathrm{Y}_2 \\ \vdots \\ \mathrm{Y}_M \end{Bmatrix} = \mathrm{T}_T \mathrm{Y} \tag{22}$$

Note: the matrix containing T_i is [M x T_T], where M is the number of outputs and T_T is the sum of the expansion orders. The desired trajectory is approximated in the same manner.

$$\mathrm{Y}_D(t+\tau) = \mathrm{T}_T \mathrm{Y}_D \tag{23}$$

Expanding the inputs in a similar manner we have:

$$u_i(t+\tau) \cong \mathrm{T}_s U_i \tag{24}$$

$$U_i \equiv \left\{ \begin{array}{c} u_i(t) \\ \dot{u}_i(t) \\ \vdots \\ \dfrac{d^s u_i(t)}{dt^s} \end{array} \right\} \tag{25}$$

The inputs are approximated by an s<u>th</u> order Taylor polynomial, where s is $R_i - r_i$. Each R_i is selected such that s for each output is equal. This converts the optimal control problem, which in general is solved with calculus of variations, into a discrete parameter optimization problem.

In model predictive control we seek to minimize a cost function over a finite horizon, the cost function is selected as follows.

$$J = \int_{T_1}^{T_2} e(t+\tau)^T Q e(t+\tau) d\tau \tag{26}$$

where Q is a diagonal, positive definite weighting matrix [M x M].

$$Q(\tau) = \begin{bmatrix} q_1(\tau) & & & 0 \\ & q_2(\tau) & & \\ & & \ddots & \\ 0 & & & q_M(\tau) \end{bmatrix} \tag{27}$$

The cost function can be approximated using the Taylor series approximations.

$$J \cong \int_{T_1}^{T_2} (\mathrm{Y}_D - \mathrm{Y}) \mathrm{T}_T^T Q \mathrm{T}_T (\mathrm{Y}_D - \mathrm{Y}) d\tau \tag{28}$$

The only components that depend on τ are T and Q.

$$J \cong (\mathrm{Y}_D - \mathrm{Y})^T \Pi (\mathrm{Y}_D - \mathrm{Y}) \tag{29}$$

$$\Pi = \int_{T_1}^{T_2} \begin{bmatrix} q_1 \mathrm{T}_1^T \mathrm{T}_1 & & & 0 \\ & q_2 \mathrm{T}_2^T \mathrm{T}_2 & & \\ & & \ddots & \\ 0 & & & q_3 \mathrm{T}_M^T \mathrm{T}_M \end{bmatrix} d\tau \tag{30}$$

Each $q_i \mathrm{T}^T \mathrm{T}$ is a square block matrix [R_i x R_i]. This integral is easily determined in closed form. The necessary conditions for optimality are

$$\frac{\partial J}{\partial u} = 0 \tag{31}$$

Application of the necessary condition yields

$$\frac{\partial J}{\partial u} = -(\mathrm{Y}_D - \mathrm{Y})^T \Pi \frac{\partial \mathrm{Y}}{\partial u} = [0] \tag{32}$$

This condition is an M*s row vector. The structure of Y_i for a control affine system follows.

$$\mathrm{Y}_i = \left\{ \begin{array}{c} {}_i\alpha_0(x) \\ \vdots \\ {}_i\alpha_{\rho_i-1}(x) \\ {}_i\alpha_{\rho_i}(x) + {}_i\beta^1(x)u_1 + \cdots + {}_i\beta^M(x)u_M \\ {}_i\alpha_{\rho_i+1}(x, u_{1-M}) + {}_i\beta^1(x)\dot{u}_1 + \cdots + {}_i\beta^M(x)\dot{u}_M \\ \vdots \\ {}_i\alpha_R(x, u_{1-M}, \ldots, d^{(s-1)}u_{1-M}/dt^{(s-1)}) + {}_i\beta^1(x)u_1^s + \cdots + {}_i\beta^M(x)u_M^s \end{array} \right\} \tag{33}$$

The structure for dY/du takes on a convenient form.

$$\frac{\partial Y_i}{\partial u} = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ {}_i\beta^1 & 0 & 0 & \cdots & {}_i\beta^M & 0 & 0 \\ & \ddots & 0 & \cdots & & \ddots & 0 \\ \frac{\partial\, {}_i\alpha_R}{\partial u_1} & \cdots & {}_i\beta^1 & \cdots & \frac{\partial\, {}_i\alpha_R}{\partial u_M} & \cdots & {}_i\beta^M \end{bmatrix} \tag{34}$$

The first ρ_i rows of this matrix are zero. For convenience we define Π_i as the ith block of the Π matrix.

$$\Pi_i = \int_{T_1}^{T_2} q_i \mathrm{T}_i^T \mathrm{T}_i \, d\tau \tag{35}$$

Combining these definitions with the necessary condition for optimality results in the Equation (36).

$$[\mathrm{Y}_D - \mathrm{Y}]^T \begin{bmatrix} \Pi_i & & 0 \\ & \ddots & \\ 0 & & \Pi_M \end{bmatrix} \begin{bmatrix} \partial \mathrm{Y}_1/\partial u \\ \vdots \\ \partial \mathrm{Y}_M/\partial u \end{bmatrix} = 0 \tag{36}$$

An equivalent expression is found by eliminating the columns of Π that multiply the zero rows of $\partial Y_i / \partial u$.

$$\Pi_{iR} = \Pi_i(:, \rho_i + 1 : end) \tag{37}$$

$$\Pi_R = \begin{bmatrix} \Pi_{1R} & & 0 \\ & \ddots & \\ 0 & & \Pi_{MR} \end{bmatrix} \tag{38}$$

Eliminating the zero rows of dY_R/du we get the following.

$$\left[d\mathrm{Y}_R \right] = \begin{bmatrix} \partial \mathrm{Y}_1 / \partial u(\rho_1 + 1 : R_i, :) \\ \vdots \\ \partial \mathrm{Y}_M / \partial u(\rho_M + 1 : R_i, :) \end{bmatrix} \tag{39}$$

With our selection of R_i such that s for each output is equal, dY_R is a square s*M block lower triangular matrix. Also, Π_R is an s*M square matrix. The equivalent optimal condition is then expressed as follows.

$$\left[\mathrm{Y}_D - \mathrm{Y} \right]^T \Pi_R d\mathrm{Y}_R = [0] \tag{40}$$

where dY_R^{-1} exists, we have the following condition.

$$\left[\mathrm{Y}_D - \mathrm{Y} \right]^T \Pi_R = [0] \tag{41}$$

Due to the block structure of Π_R this condition can be expanded.

$$\left[\left(\mathrm{Y}_{1D} - \mathrm{Y}_1 \right)^T \Pi_{1R} \quad | \cdots | \quad \left(\mathrm{Y}_{MD} - \mathrm{Y}_M \right)^T \Pi_{MR} \right] = [0] \tag{42}$$

These conditions are similar to the single input single output case, in that we have M equations similar to the SISO form. Note, that unlike the SISO case, Y_i contains each of the controls $U_1 – U_M$. With a little work, we can solve for the control parameters. First split the Π_{iR} matrix to be conformal with $(Y_{iD} – Y_i)^T$.

$$\Pi_{iR} = \begin{bmatrix} \Pi_{iR1} \\ \Pi_{iR2} \end{bmatrix} \tag{43}$$

The split is made such that Π_{iR1} is $[1:\rho_i,(\rho_i+1):R_i]$ and Π_{iR2} is an $(\pi_i+1):R_i$ square matrix. The error vector is split up similarly.

$$\mathrm{Y}_{iDU} = \mathrm{Y}_{iD}(1 : \rho_i) \tag{44}$$

$$\mathrm{Y}_{iDL} = \mathrm{Y}_{iD}(\rho_i + 1 : R_i) \tag{45}$$

$$\Upsilon_{iU} = \Upsilon_i(1:\rho_i) \tag{46}$$

$$\Upsilon_{iL} = \Upsilon_i(\rho_i + 1 : R_i) \tag{47}$$

Writing our necessary conditions with these definitions we have

$$\Upsilon_{iDU}\Pi_{iR1} + \Upsilon_{iDL}\Pi_{iR2} = \Upsilon_{iU}\Pi_{iR1} + \Upsilon_{iL}\Pi_{iR2} \tag{48}$$

Solve for Υ_{iL} to get:

$$\Upsilon_{iL} = \Pi_{iR2}^{-1}\Pi_{iR1}^{T}(\Upsilon_{iDU} - \Upsilon_{iU}) + \Upsilon_{iDL} \tag{49}$$

If we are interested only in the current control input, the first component of Υ_{iL} can be extracted. We will have M equations for M unknowns. Combining these equations and solving for the control input, we get the following equations.

$$B = \begin{bmatrix} {}_1\beta^1 & \cdots & {}_1\beta^M \\ \vdots & \ddots & \vdots \\ {}_M\beta^1 & \cdots & {}_M\beta^M \end{bmatrix} \tag{50}$$

$$A = \begin{bmatrix} {}_1\alpha_{\rho_1} & \cdots & {}_M\alpha_{\rho_M} \end{bmatrix}^T \tag{51}$$

$$\Upsilon^1{}_{DL} = \begin{bmatrix} \Upsilon_{1DL}(1) & \cdots & \Upsilon_{MDL}(1) \end{bmatrix}^T \tag{52}$$

$$\Upsilon_{DU} = \begin{bmatrix} \Upsilon_{1DU} & \cdots & \Upsilon_{MDU} \end{bmatrix}^T \tag{53}$$

$$\Upsilon_{U} = \begin{bmatrix} \Upsilon_{1U} & \cdots & \Upsilon_{MU} \end{bmatrix}^T \tag{54}$$

$$\bar{C} = \begin{bmatrix} u_1(t) & \cdots & u_M(t) \end{bmatrix}^T \tag{55}$$

$$K_i = \Pi_{iR2}^{-1}\Pi_{iR1}^{T}(1,:) \tag{56}$$

$$K_T = \begin{bmatrix} K_1 & & & 0 \\ & K_2 & & \\ & & \ddots & \\ 0 & & & K_M \end{bmatrix} \tag{57}$$

Using this notation, our control law is:

$$\bar{C} = B^{-1}\left(K_T(\Upsilon_{DU} - \Upsilon_U) + \Upsilon_{DL} - A\right) \tag{58}$$

If the B matrix is singular, in some cases the Moore-Penrose pseudo inverse may be used.

Autonomous Glider Control Law

An angle tracking controller for the glider system described is considered.

$$h = [\psi \quad \theta \quad \phi] \tag{59}$$

While it is desirable to control heading and pitch angles directly, the roll angle is also included to fully specify the trajectory. The relative degree of each output is 2, therefore two derivatives of the output are required for control affine predictive control. The first derivative of the output is given in Equation (3), the second derivative follows:

$$\ddot{h} = \dot{K}_{RM}\omega + K_{RM}\dot{\omega} \tag{60}$$

The derivative of angular velocity is given by Equation (5), and K'$_{RM}$ is not shown for simplicity. The full control vector is found in the derivative of angular velocity. Since only the current control input is desired Equation (15) is Y$_L$ from the control law development.

$$\mathrm{Y}_L = \left[\dot{K}_{RM}\right]\omega - \left[K_{RM}\right][\mathrm{I}]^{-1}\left([S_w][\mathrm{I}]\omega - \vec{\mathrm{M}}_{AERO} - \left[\mathrm{M}_C\right]\vec{\delta}_c\right) \tag{61}$$

or:

$$\mathrm{Y}_L = (\mathrm{Y}^T{}_{DU} - \mathrm{Y}^T{}_U)\mathrm{K} + \mathrm{Y}_{DL} \tag{62}$$

Solving for the control vector gives the following control law:

$$\vec{\delta}_c = \left[M_c{}^{-1}\right]\left(\left[S_\omega\right][\mathrm{I}]\omega - \vec{M}_{AERO} + [\mathrm{I}]\left[K_{RM}\right]^{-1}\left(\mathrm{Y}_{DL}^T + \left(\mathrm{Y}_{DU}^T - \mathrm{Y}_U^T\right)\mathrm{K} - \dot{\mathrm{K}}_{RM}\omega\right)\right) \tag{63}$$

Two matrix inverses are required in the control law and fortunately one is easily determined in closed form:

$$\left[K_{RM}\right]^{-1} = \begin{bmatrix} -s_\theta & 0 & 1 \\ s_\phi c_\theta & c_\phi & 0 \\ c_\theta c_\phi & -s_\phi & 0 \end{bmatrix} \tag{64}$$

The matrix M$_c$ contains the systems control moment coefficients and may be singular. In this case the pseudo inverse may be used to find the least magnitude best fit solution.

EXAMPLE RESULTS

The simulation is exercised with physical parameters depicted in Table 1-2, aerodynamic parameters in Table 1-3, and wind model parameters in Table 1-1. Controller parameters are shown in Table 1-4. In order to increase the potential energy of the glider, a circle at the core of the thermal is tracked. Initially the glider is directed to the tangent of the circle, when it enters the desired circle it is directed around the circle in a smooth trajectory according to Equations (65) and (66).

$$\varphi_D = \tan^{-1}\left(Dy, Dx\right) \tag{65}$$

$$\dot{\varphi}_D = V_{xy} / r_{dc} \tag{66}$$

where Dy and Dx are the y and x components of the difference vector between the current airplane position and the center of the thermal, rdc is the radius of the desired circle at the core of the thermal, and Vxy is the velocity of the glider in the x-y plane. This heading angle mapping results in a smooth transition into the thermal, where the desired circle is quickly converged upon.

For the numerical example considered here, two known thermals exist in the immediate area, the glider first engages the close thermal until it reaches an altitude of 200 m, when it is directed toward the second known thermal.

The airplane begins aligned with the I_I axis, with a pitch angle of 0 degrees, an I_B velocity of 12.60 m/s, a K_B velocity of -0.25 m/s, at an altitude of 40 m. The position of the glider mass center in the x-y plane is depicted in Figure 1-2 where the contour lines show the thermal magnitude. The 3 dimensional flight path is shown in Figure 1-3. The corresponding altitude and control deflection history is shown in Figures 1-4 and 1-5. The glider completes 31 circles in the first thermal where it reaches 200 m of altitude. The glider then heads toward the second thermal, losing 40 m of altitude in the process, and begins climbing the second thermal. The corresponding attitude history is shown in Figures 1-6 through 1-8. Desired heading is shown as the dashed line in Figure 1-6. Body frame velocities are depicted in Figures 1-9 through 1-11.

As the glider reaches the core of the first thermal characterized by a positive vertical airflow, altitude of the glider levels off and begins to increase. A nearly constant rate of increase in altitude of 0.5 m/s is reached. After approximately 3 1/2 minutes of flight time, the gliders target altitude of 200 m is achieved. The glider achieves a constant rate of altitude increase in the second thermal of 0.25 m/s, a net altitude increase of 140 m. is achieved in the duration of the simulation.

While optimal energy extraction trajectories were not explored in this work, a simple trade study of bank angles pitch angles, and radius of the desired circle can provide insight into improving our soaring tactics. The first trade study was conducted on the first thermal from Table 1-1, where the glider is directed into the thermal for a duration of 6 minutes, where it reaches a steady state rate of altitude gain. The three parameters, desired radius, prescribed bank angle and pitch angle were varied and climb rates recorded. The results are shown in Table 1-5.

Another trade study involving glider mass and wing area was conducted. A simulation of 2 minutes was conducted where the final potential energy was recorded. Results are shown in Table 1-6.

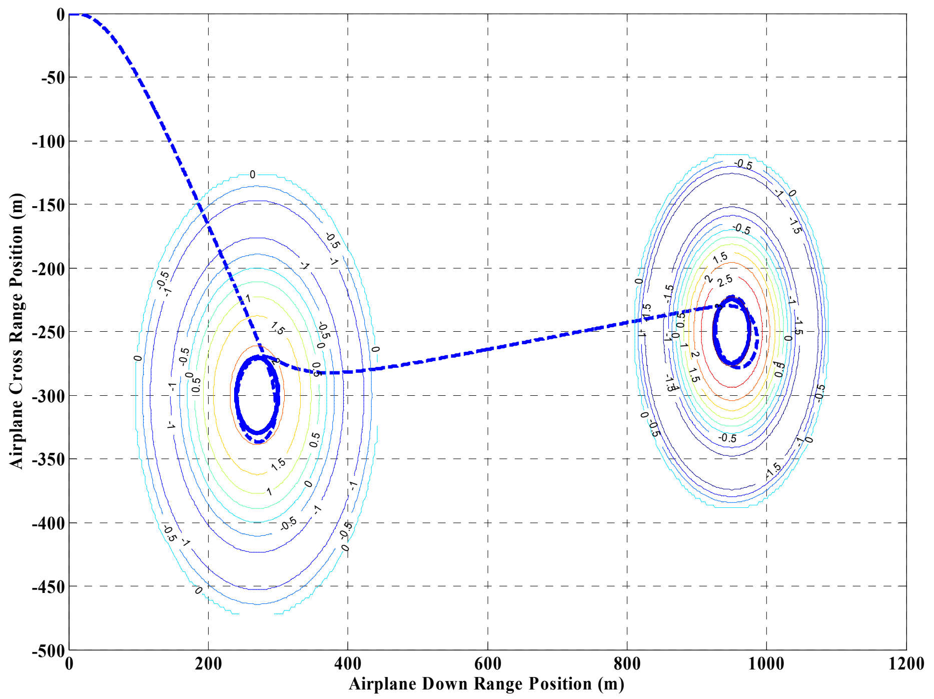

Figure 1-2: Glider Position and Thermal Profile Contour

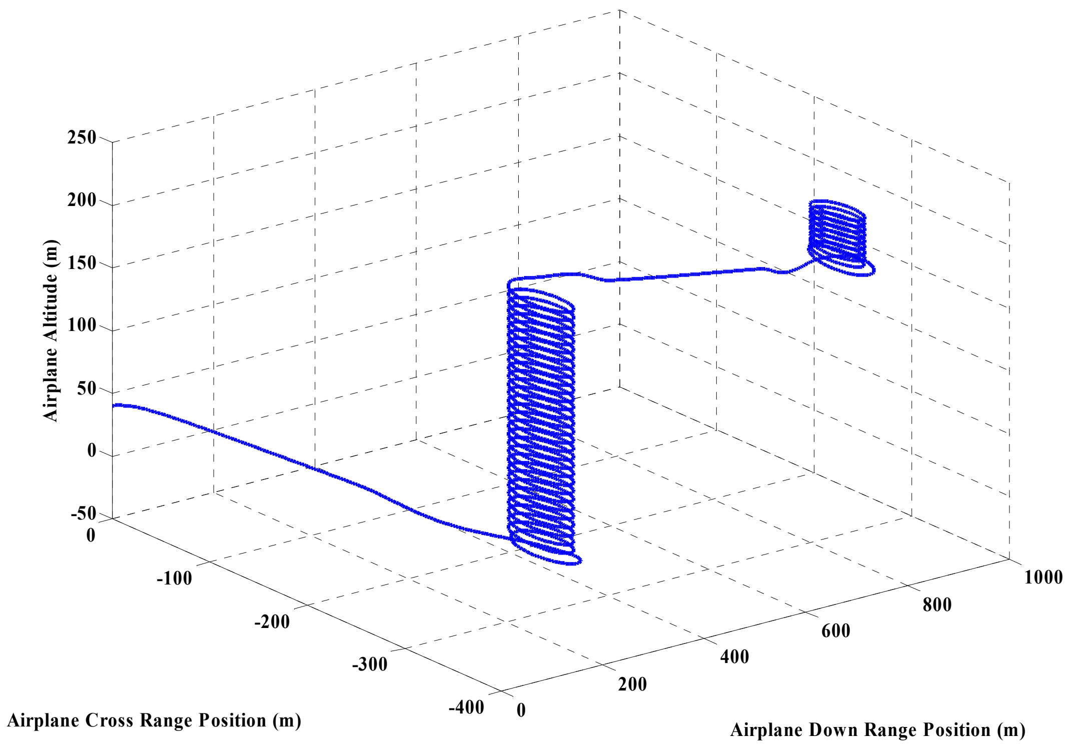

Figure 1-3: Glider Position Time History

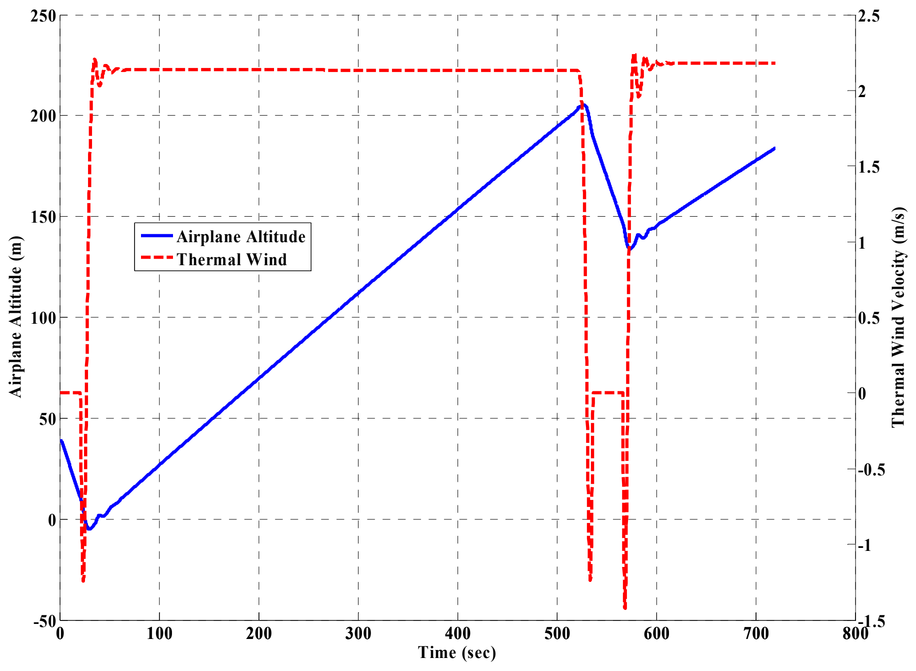

Figure 1-4: Glider Altitude and Thermal Velocity History

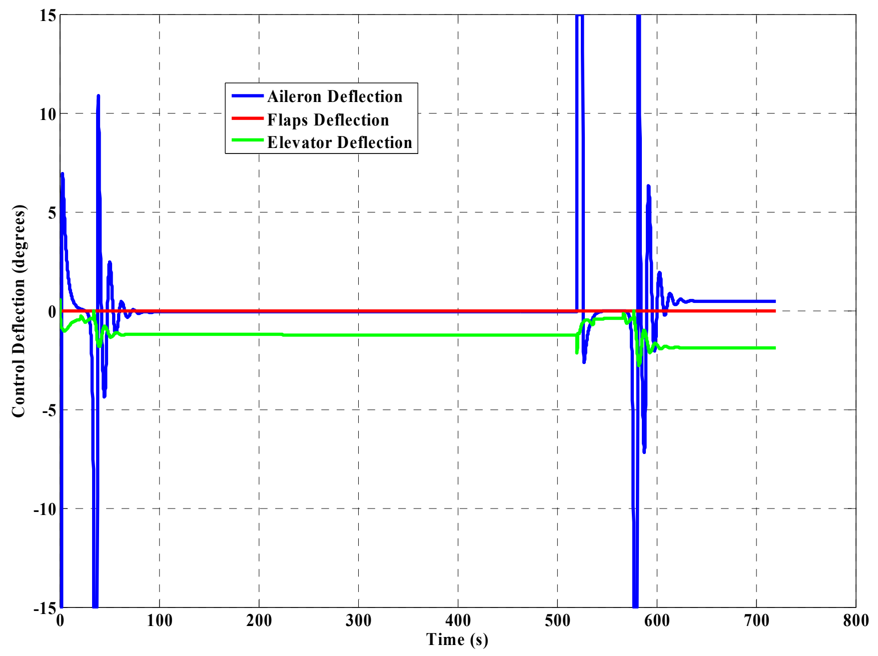

Figure 1-5: Control Flap Deflection

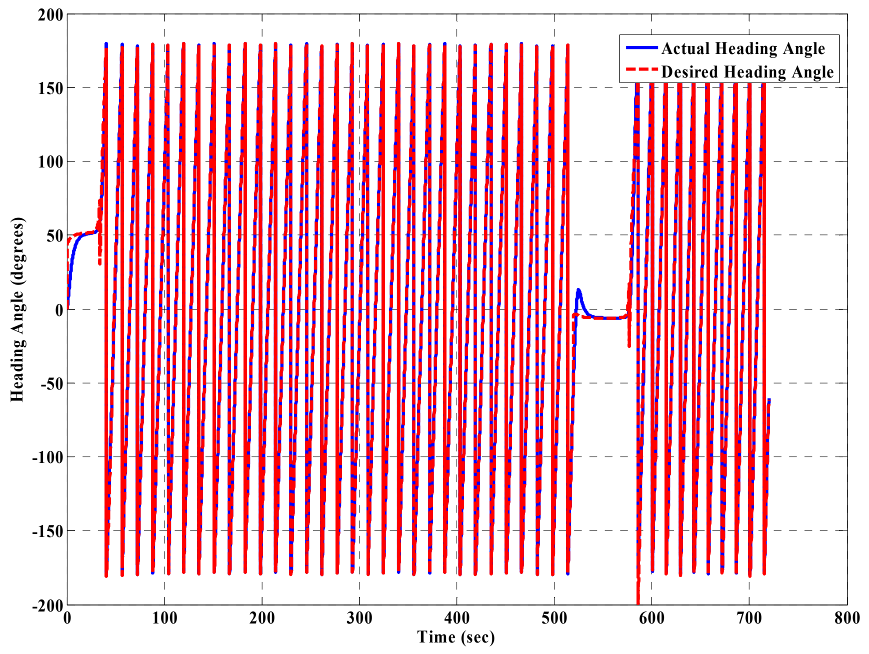

Figure 1-6: Desired and Actual Heading Angle

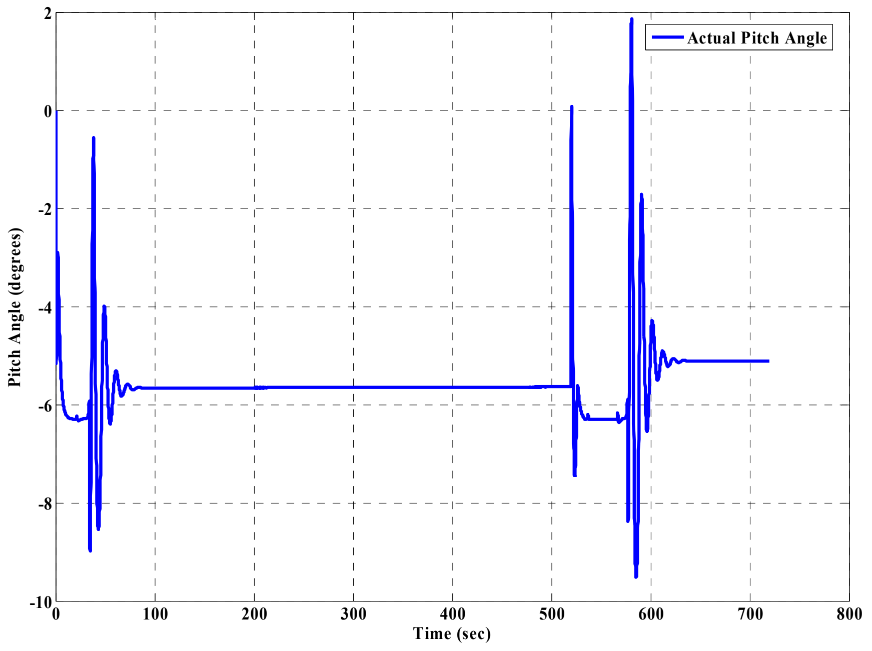

Figure 1-7: Pitch Angle Time History

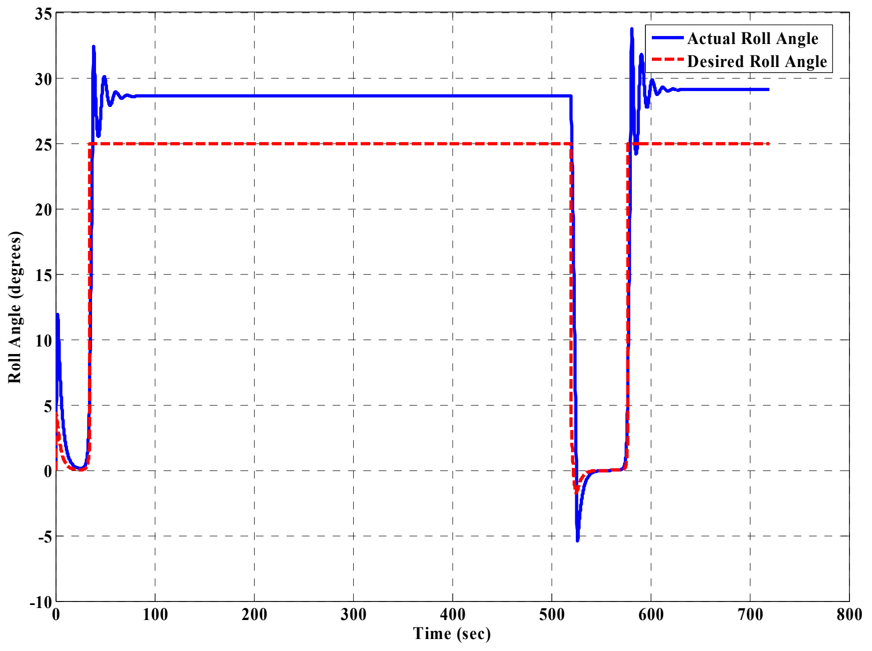

Figure 1-8: Glider Bank Angle

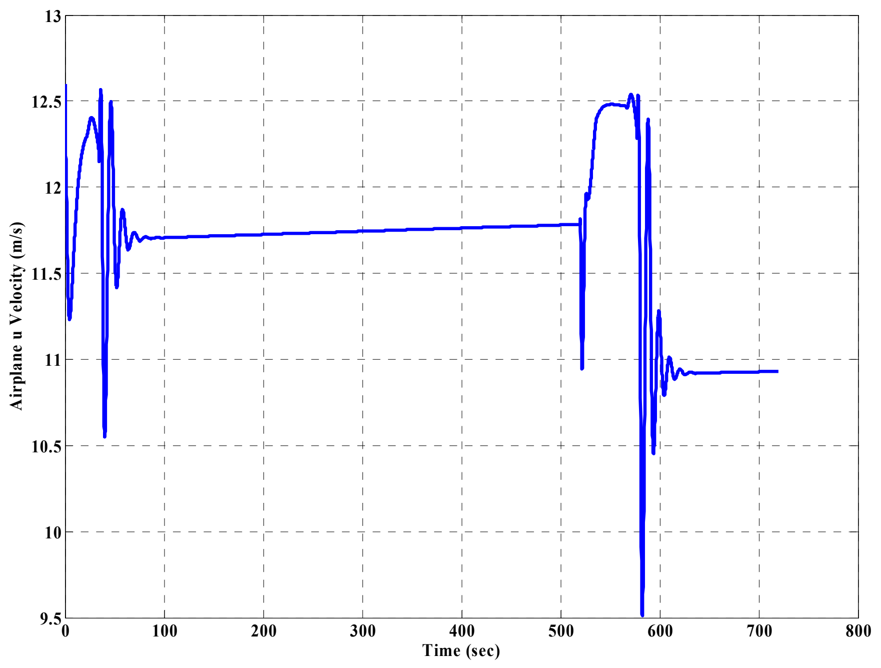

Figure 1-9: Glider Body Frame x Velocity

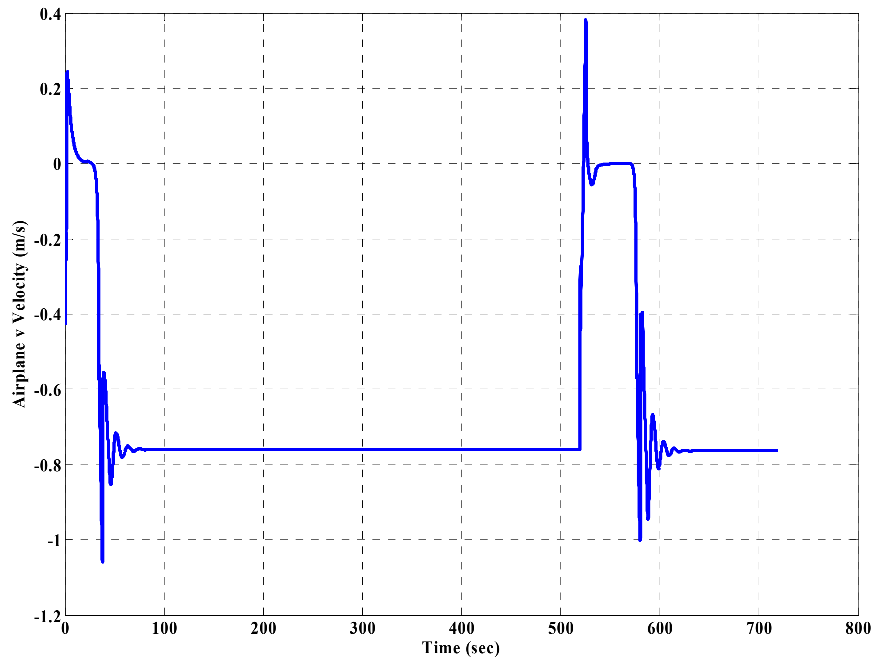

Figure 1-10: Glider Body Frame y Velocity

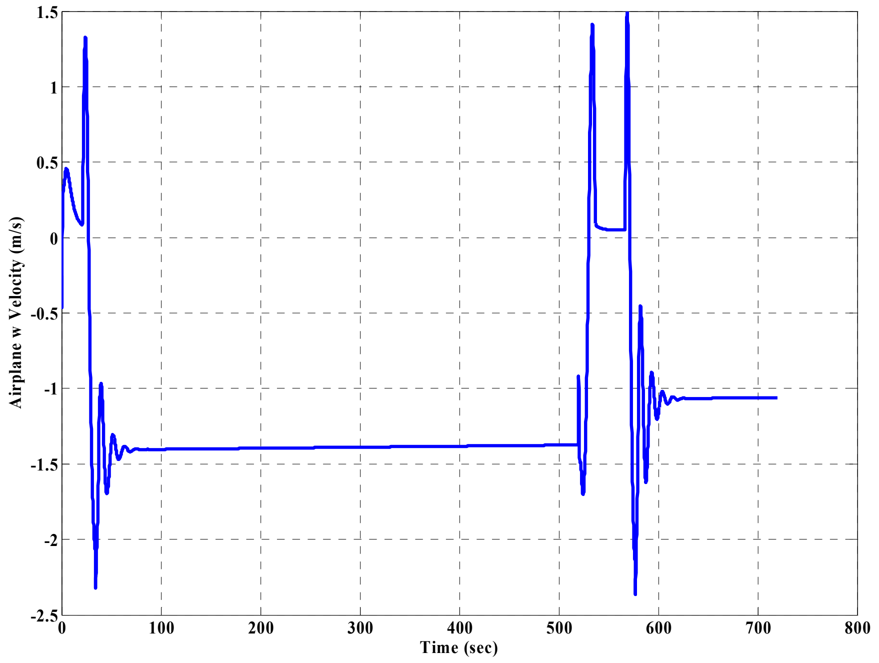

Figure 1-11: Glider Body Frame z Velocity

Table 1-1: Wind Model Parameters

General Wind Magnitude (m/s)	0.0
General Wind Direction (deg.)	-17.00
1st Thermal Magnitude (m/s)	2.50
1st Thermal Radius (m)	180.0
1st Thermal Decay Factor	-0.005
1st Thermal x Inertial Position (m)	340.0
1st Thermal y Inertial Position (m)	250.0
2nd Thermal Magnitude (m/s)	2.50
2nd Thermal Radius (m)	180.0
2nd Thermal Decay Factor	-0.005
2nd Thermal x Inertial Position (m)	340.0
2nd Thermal y Inertial Position (m)	250.0

Table 1-2: Glider Physical Parameters

Mass (Kg)	0.9015148
Inertia, Ixx (Kg m^2)	0.074240596
Inertia, Iyy (Kg m^2)	0.039028809
Inertia, Izz (Kg m^2)	0.11019321
Main Wing Span (m)	0.9144
T-Tail Span (m)	0.5300
T-Tail Chord (m)	0.1255

Table 1-3: Glider Aerodynamic Parameters

Main Wing Airfoil	RG-15
Main Wing Flaps	Trailing Edge Individual Control
T-Tail Airfoil	NACA 0009
T-Tail Flaps	Trailing Edge Elevator Only

Table 1-4: Controller Parameters

Psi Error Weighting	1.0
Theta Error Weighting	2.0
Phi Error Weighting	3.0
Expansion Order	5
Prediction Times	0.0s – 5.0s

Table 1-5: Trade Study Results

Bank Angle Deg.	Pitch Angle Deg.	Tracked Radius (m)	Rate of Climb (m/s)
10	-7	28.6	0.06783
15	-7	28.6	0.34458
20	-7	28.6	0.37809
25	-7	28.6	0.31413
15	-7	25	0.34069
15	-7	22	0.31777
15	-7	33	0.32600
15	-7	40	0.28118
10	-5	28.6	0.08516
15	-5	28.6	0.17000
20	-5	28.6	0.11213
25	-5	28.6	-0.00837
15	-5	25	0.10694
15	-5	22	0.03236
15	-5	33	0.21902
15	-5	40	0.25142
5	-5	67	0.25507
15	-5	100	0.22027
10	-9	28.6	-0.22117
15	-9	28.6	0.27955
20	-9	28.6	0.42190
25	-9	28.6	0.42545
15	-9	25	0.34337
15	-9	22	0.37935
15	-9	33	0.18301
15	-9	40	0.0481

Table 1-6: Physical Properties Trade Study

Glider Mass (Kg)	Wing Area (m^2)	Final PE
0.9515148	0.4044	568
1.1339809	0.4044	240
1.3607771	0.4044	-397
0.6803886	0.4044	767
0.2267962	0.4044	398
0.9515148	0.5055	928
0.9515148	0.38	472

CONCLUSIONS

A multiple input multiple output model predictive controller was developed with the purpose of enabling autonomous aircraft energy extraction from atmospheric winds. The flight control law requires that external information is provided on the local wind structure and full state feedback has been assumed. Numerical results exercising this control law show an aircraft altitude gain in two thermals of known location and size. Climb rates of ½ m/s and ¼ m/s were achieved in the subsequent thermals. Controller parameters were varied to study the effects of bank angle, pitch angle and desired radius in a thermal. It was found that for smaller thermals more aggressive bank angles still resulted in impressive climb rates. A trade study involving glider mass and wing area showed increased system mass will result in decreased potential energy gain unless wing area is also increased.

REFERENCES

[1] R.K. Mutha, W.R. Cluett, A. Penlidis, “Nonlinear Model-based Predictive Control of Control Nonaffine Systems,” *Automatica*, Vol. 33, No. 5, pp 907-913, 1997.

[2] W.H. Chen, “Predictive control of general nonlinear systems using approximation,” IEE, *Control Theory Appl.*, Vol. 151, No. 2, March 2004.

[3] P. Lissaman, “Wind Energy Extraction by Birds and Flight Vehicles”, *43rd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2005-241, 10-13 January, 2005.

[4] M.J, Allen, “Autonomous Soaring for Improved Endurance of a Small Uninhabited Air Vehicle,” *43rd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2005-1025, 10-13 January 2005.

[5] Y. J. Zhao, “Optimal patterns of glider dynamic soaring,” *Optimal control applications and methods,* Vol. 25, pp. 67-89, 2004.

[6] G. Sachs, O. da Costa, “Optimization of Dynamic Soaring at Ridges,” *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 11-14 August 2003, AIAA 2003-5303.

[7] Y. J. Zhao, Y. C. Qi, “Minimum fuel powered dynamic soaring of unmanned aerial vehicles utilizing wind gradients,” *Optimal control applications and methods*, Vol. 25, pp. 211-233, 2004.

[8] N. Goto, H. Kawable, “Direct optimization methods applied to a nonlinear optimal control problem,” *Mathematics and Computers in Simulation*, Vol. 51, pp. 557-577, 2000.

2 A HYBRID TRAJECTORY OPTIMIZATION METHOD APPLIED TO AUTONOMOUS SOARING

Jason Kyle and Mark Costello

A HYBRID TRAJECTORY OPTIMIZATION METHOD APPLIED TO AUTONOMOUS SOARING

Jason Kyle[υ]

Department of Mechanical Engineering

Oregon State University

Corvallis, OR.

Mark Costello[β]

School of Aerospace Engineering

Georgia Institute of Technology

Atlanta, Georgia

ABSTRACT

Determination of optimal trajectories for nonlinear dynamic systems is considered. Standard gradient based methods are fast, but fail for cases where a good initial guess is not known. Particle swarm optimization methods are robust to a poor initial guess but slow. This paper combines the two techniques to form a fast and robust trajectory optimization solver. Example results to substantiate this claim are shown for an autonomous soaring trajectory optimization problem.

[υ] Graduate Research Assistant, Member AIAA

[β] Sikorsky Associate Professor, School of Aerospace Engineering, Associate Fellow AIAA

NOMENCLATURE

$\vec{x}$	= continuous time state vector
$\vec{u}$	= continuous time control vector
ϕ_s	= initial and final state vector penalty function
ϕ_u	= initial and final control vector penalty function
J	= scalar cost function to minimize
$\vec{f}$	= continuous time system governing dynamic equations
$\vec{u}_{\min}, \vec{u}_{\max}$	= control vector bounds
t_f	= optimization problem final time
$t_{f\min}, t_{f\max}$	= final time bounds
$\vec{x}_{\min}, \vec{x}_{\max}$	= state vector bounds
$\vec{x}_i$	= i^{th} discrete state vector
$\vec{u}_i$	= i^{th} discrete control vector
$\vec{G}$	= discrete time system governing dynamic equations
A_j	= parameter matrix of the j^{th} control approximation
$\vartriangle t$	= discrete system time step
$\vec{v}^k$	= vector of unknown parameters at the k^{th} iteration
$\vec{v}_{\min}, \vec{v}_{\max}$	= simple bounds on unknown parameters
$particle_i$	= i^{th} vector of optimization parameters for particle swarm optimization algorithm
$pbest_i$	= vector of optimization parameters with the lowest cost in the history of $particle_i$
$lbest_i$	= vector of optimization parameters with the lowest cost of the neighbors to $particle_i$
$gbest$	= vector of optimization parameters with the lowest cost in the history of all particles
V_i	= velocity of the solution trajectory for $particle_i$
I_w	= inertia weight for particle swarm optimization algorithm
c_1, c_2	= acceleration parameters for particle swarm optimization algorithm
$r()$	= random number function
$[\cdot]_\#$	= projection onto simple bounds
N_{PSO}	= number of particle swarm optimization iterations to be executed
$T_\%$	= percentage of the total solution space
k	= solution iteration index
α^k	= line search parameter for the k^{th} iteration

$\vec{d}^k$	=	search direction for the k^{th} iteration
∇	=	gradient operator
v^*	=	optimal vector of parameters
I	=	set of optimization parameters with inactive bound constraints
A_l	=	set of optimization parameters with active lower bound constraints
A_u	=	set of optimization parameters with active upper bound constraints
g^k	=	gradient of cost function at the k^{th} iteration
N_{cg}	=	number of gradient algorithm iterations to be executed
D	=	positive definite diagonal scaling matrix
$\langle \cdot \rangle_{I^k}$	=	dot product over the parameters in the set I^k
μ^k	=	scalar parameter to compute the k^{th} conjugate direction
$\tilde{d}^k$	=	unaltered gradient search direction at the k^{th} iteration
β	=	Armijo line search rule parameter
J_k	=	cost at iteration k
J_{PG_0}	=	cost at the first evaluation of algorithm DS
x, y, z	=	inertial frame position vector components of the glider mass center
$\psi_{pm}, \theta_{pm}, \phi_{pm}$	=	glider heading, pitch and bank angles
W_x, W_y, W_z	=	wind velocity components represented in the inertial frame
V_{pm}	=	glider airspeed magnitude
C_L	=	glider lift coefficient
C_{do}	=	glider parasitic drag coefficient
k	=	glider induced drag coefficient
S	=	glider wing area
ρ	=	density of air
g	=	gravitational acceleration constant
m	=	glider mass
D, L	=	glider drag and lift forces
w_{th}	=	thermal wind velocity at the current position
w_{peak}	=	wind velocity at the core of the thermal
k_1, k_2, k_3, k_4	=	thermal model shape parameters
r_2	=	thermal outer radius
r	=	distance from the current position to the thermal core
w_D	=	thermal model downdraft velocity
w_e	=	environmental sink velocity

w_{eff}	=	effective wind velocity at the current glider location
w_{hg}	=	horizontal wind velocity
w_g	=	average slope of the horizontal wind gradient
h	=	current height of glider
$h_{\min}$	=	altitude the horizontal wind gradient begins
$h_{\max}$	=	altitude the horizontal wind gradient ends
A	=	horizontal wind shape parameter
$w_{h\min}$	=	minimum horizontal wind velocity

INTRODUCTION

The majority of numerical trajectory algorithms are based on gradient information [1]. These methods are able to handle complex problems and tend to rapidly converge near a local minimum. However convergence may be slow or even fail for problems with many local minima, singular minima or where good initial guesses are not known. Alternatively, swarm intelligence based optimization methods exhibit a high level of success on various difficult optimization problems [8]. Computationally these methods are not competitive with local search techniques for trajectory optimization problems. However they exhibit significantly different failure modes. A method composed of a local search and a swarm based algorithm is presented as a technique for difficult trajectory optimization problems.

Many variants of local optimization techniques exist; the most popular methods can be categorized as direct or indirect. Betts gives an overview of some of these methods in [1]. He concludes that indirect methods suffer from three major drawbacks including the requirement of analytic expressions for the necessary conditions which is problem dependant and can be a difficult task, a surprisingly small region of convergence, and if path inequalities are required the sequence of constrained and unconstrained subarcs must be guessed prior to solving. Direct methods do not suffer from any of these problems, however they do require reasonable initial guesses, only approximations of local minima are guaranteed, and convergence can be slow for some problems. Often direct methods are chosen for difficult optimization problems.

Numerous methods for converting a continuous optimal control problem into a discrete parameter optimization problem are summarized by Hull [2]. The resulting nonlinear programming problem (NLP) can then be solved by an existing nonlinear programming code.

Swarm intelligence techniques make up a fundamentally different approach to optimization problems. Typically swarm solvers are modeled after swarm behavior in nature, such as bird flocking, fish schooling, and ant colonies. Populations of solutions interact locally to decide which direction to move in. Computation per iteration is usually very small because cost information is shared and gradient information is not required. Typically many iterations are required for convergence, and in general swarm techniques are not competitive with local search methods when applied to trajectory optimization problems. Often referred to as global optimization techniques, swarm solvers are robust to a poor initial guess of the solution.

A specific example of a swarm based technique is particle swarm optimization (PSO). Introduced by Eberhart and Kennedy [3], PSO is modeled after the social behavior of bird flocking or fish schooling, this solver is very robust to local minima and is often referred to as a global solver. In it's original form, global or even local convergence cannot be guaranteed for PSO [4]. Bergh [4,5] identified and addressed this issue with a simple modification to PSO resulting in a guaranteed locally convergent method. Many difficult problems have been successfully solved by PSO methods and an overview and discussion of applications can be found in [6].

Several strategies to combine global and local search strategies have been proposed. A typical design strategy attempts to maximize the ability to find a global minimum while minimizing the number of iterations required. A method combining genetic algorithms and Quasi-Newton local search was proposed by Renders and Flasse [7]. Applied to a system identification problem, one of the resulting combination methods was shown to perform well. A general methodology for combining global and local direct search algorithms was described by Syrjakow and Szczerbicka [8]. Multiple minima were successfully investigated on a well known function.

A combination method is proposed in this work, which takes advantage of the efficiency of a local gradient method and the global search strategy of a swarm method. A strategy for combining these methods is investigated and compared with direct shooting. The example scenario considered investigates paths of maximal altitude gain for a micro air robot in naturally occurring wind structures. Soaring in a single small thermal is considered, the hybrid algorithm is found to be marginally slower than direct shooting. When a horizontal wind gradient is added, the direct shooting method fails to converge but the hybrid method converges.

OPTIMIZATION PROBLEM DEFINITION

The optimization problem considered in this work is to determine the control history $\vec{u}$, initial conditions $\vec{x}(t_0)$, and final time t_f which minimize the scalar cost function given by Equation (67), subject to the dynamic system given by Equation (68), and the inequality constraints given in Equations (69) through (71).

Table 2-1: Optimization Problem 1 (OP1)

Minimize:

$$J(\vec{x},\vec{u}) = \phi_s\left(\vec{x}(t_0),\vec{x}(t_f)\right) + \phi_u\left(\vec{u}(t_0),\vec{u}(t_f)\right) \tag{67}$$

Subject to:

$$\dot{\vec{x}} = \vec{f}(\vec{x},\vec{u}) \tag{68}$$

$$\vec{u}_{\min} \le \vec{u} \le \vec{u}_{\max} \tag{69}$$

$$t_{f\min} \le t_f \le t_{f\max} \tag{70}$$

$$\vec{x}_{\min} \le \vec{x}(t_0) \le \vec{x}_{\max} \tag{71}$$

Non-Linear inequality constraints on the state vector are dealt with by defining a new state as the integral of the constraint violation squared. The final values of the extended states are penalized with a quadratic function in the cost. Problem 1 differs from a standard optimal control problem in that the initial conditions are unknown, and the initial and final states and controls are nonlinearly coupled in the cost function. Investigation into closed trajectories with unknown initial states would be one example. The states and controls in Problem 1 are continuous in time; typically the calculus of variations is employed to solve this type of problem.

CONVERSION TO PARAMETER OPTIMIZATION PROBLEM

For implementation with the solution methods considered in this work, Problem 1 is converted to a discrete parameter optimization problem given by Problem 2.

Table 2-2: Optimization Problem 2 (OP2)

Minimize:

$$J(\vec{X},\vec{U}) = \phi_s(\vec{x}_1,\vec{x}_{L+1}) + \phi_u(\vec{u}_1,\vec{u}_L) \tag{72}$$

Subject to:

$$\vec{x}_{i+1} = \vec{x}_i + G(\vec{x}_i,\vec{u}_i(A_j)) \qquad i = 1\ldots L \tag{73}$$

$$\vec{u}_{\min} \le \vec{u}_i \le \vec{u}_{\max} \qquad i = 1\ldots L \tag{74}$$

$$t_{f\min} \le t_f \le t_{f\max} \tag{75}$$

$$\vec{x}_{1\min} \le \vec{x}_1 \le \vec{x}_{1\max} \tag{76}$$

where $\vec{x}_i$ and $\vec{u}_i(A)$ are the ith discrete state and control vectors, and $G(\vec{x}_i,\vec{u}_i(A_j))$ depends on the explicit integration method chosen. Fixed step 4th order explicit Runge Kutta is chosen for this work. In OP2 the first subscript denotes the discrete node index, the second subscript denotes the vector component index, and hats are employed over all vector values to reduce confusion.
The control is parameterized as a continuous function of the coefficient matrix A and time. The optimization problem is now to determine the discrete control parameters and initial states of Problem 2. Fitness or cost of a particle is evaluated by forward propagation of the system states according to Equation (73), followed by evaluation of the cost given by Equation (72). The control parameters A, initial state vector $\vec{x}^1$, and final time t_f make up the unknown parameters OP2.

Choice of the control function is highly problem dependant; options may include polynomials, sine and cosine functions or a combination of both. For the example investigated in this work, the controls are approximated by a fixed number of low order polynomials valid over a discrete time segment. Continuity between polynomials is enforced in closed form. An example of control parameterization is shown in Figure 2-1.

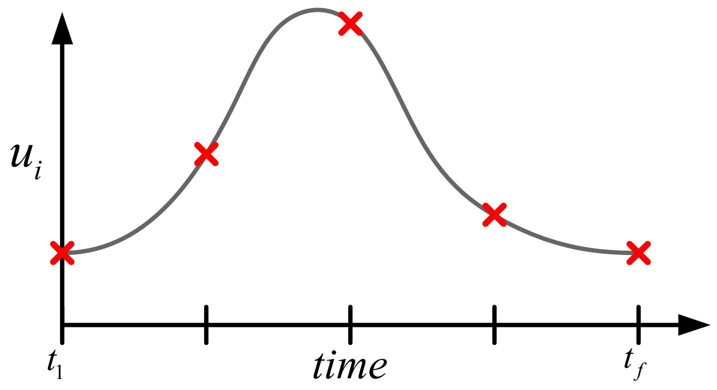

Figure 2-1: Multiple Low Order Polynomial Control Parameterization

In this example the control is split into four 3rd order polynomials shown in Equation (77), continuity in the control and derivative of control is strictly enforced at each node resulting in a first order continuous control approximation.

$$u_i = \sum_{l=1}^{4} a_{i,l}^{j} \tau^{l-1} \qquad \{t_j \le t \le t_{j+1}\} \tag{77}$$

where τ is a normalized time parameter, Equation (77) is valid over the time range indicated. The coefficients of the polynomial a^j can be represented as control and control derivatives at the discrete time points.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{Bmatrix} a_{i,1}^{j} \\ a_{i,2}^{j} \\ a_{i,3}^{j} \\ a_{i,4}^{j} \end{Bmatrix} = \begin{Bmatrix} u_i^1 \\ \dot{u}_i^1 \\ u_i^2 \\ \dot{u}_i^2 \end{Bmatrix} \tag{78}$$

Control, and control rate at the discrete points now become our control parameters, and first order continuity is achieved.

Explicit discretization of the governing dynamic equations allows the state vector time history to be determined by directly solving Equation (73), given the control parameters, initial state vector, and final time. The optimization problem OP2 can be described by Equations (79) and (80).

Table 2-3: Optimization Problem 3 (OP3)

Minimize:

$$J(\vec{X}(\vec{v}),\vec{v}) = \phi_v(\vec{X}(\vec{v}),\vec{v}) \tag{79}$$

Subject to:

$$\vec{v}_{\min} \le \vec{v} \le \vec{v}_{\max} \tag{80}$$

where v is a vector containing the unknown controls, initial state vector, and final time. The optimal control problem (OP3) is to determine the unknown parameter vector v, in the region defined by Equation (80), which minimizes the nonlinear cost function given by Equation (79).

PARTICLE SWARM OPTIMIZATION

Designed by Eberhart and Kennedy [3], particle swarm optimization (PSO) is a relatively new technique based on the social behavior of bird flocking or fish schooling. Characterized by a simple structure, and few tuning parameters PSO has been used to effectively solve many difficult optimization problems [6].

The algorithm begins with a fixed number of possible solution vectors, or particles randomly initialized within the solution space. Each particle keeps track of it's best location (personal best) as measured by the cost to be minimized, also the best particle in the neighborhood of each particle is recorded (local best). An iteration consists of updating each particle by a combination of it's personal best, local best, and a record of the last update direction (velocity) according to Equations (81) through (83).

$$V_{i,j} = I_w * V_{i,j} + c_1 * r(\bullet) * (pbest_{i,j} - particle_{i,j}) + c_2 * r(\bullet) * (lbest_{i,j} - particle_{i,j}) \quad (81)$$

$$V_{i,j} = \left[particle_{i,j} + V_{i,j} \right]_{\#} \quad (82)$$

$$particle_{i+1} = particle_i + V_i \quad (83)$$

where c_1 and c_2 are positive constants usually called acceleration coefficients, $r()$ generates random numbers in the range [0,1], I_w is the inertia weight, V_i is the velocity of $particle_i$, and $[\cdot]_{\#}$ denotes projection onto simple bounds defined in Equation (84).

$$\left[z \right]_{\#} = \left\{ \begin{matrix} b_l \;\; for \; z \le b_l \\ z \;\; for \; b_l \le z \le b_u \\ b_u \;\; for \; z \ge b_u \end{matrix} \right\} \quad (84)$$

The best particle that $particle_i$ has achieved so far is denoted $pbest_i$, the best particle in each particles neighborhood is denoted $lbest_i$. Typically both acceleration coefficients are set to 2.0, and the inertia weight is varied linearly from 1.2 to 0.4 [4]. The neighborhood of each particle is usually selected as a fixed value denoting the level of interaction between particles. For a neighborhood of 2 each particle is only attracted to the best of a single neighbor to the left and right as determined by the particle index.

Selecting a small neighborhood allows a more thorough search of the solution space, while usually requiring more iterations. The global best version of the PSO algorithm (GBest) increases the neighborhood of each particle to the total number of particles. The GBest algorithm is a special case of the local best (LBest) algorithm described above. The LBest PSO algorithm is outlined in Table 2-2.

Table 2-4: Particle Swarm Optimization Algorithm (PSO)

```
Given: $N_{PSO}, T_{\%}, k, \vec{v}^k$
Initialize: all $p_i$ randomly in region $T_{\%}$ of total feasible region

For $N_{PSO}$

    For all $p_i$
      calculate fitness of $p_i$
      If fitness of pi is less than $pbest_i$
        update $pbest_i$
      For all j in the Neighborhood of particle i
        If fitness of $pbest_j$ is less than $lbest_i$
          update $lbest_i$
      End
    End

    For all $p_i$
      update velocity of $p_i$ according to Equations (81)
      and (82)
      update $p_i$ according to Equation (83)
    End

    If convergence criterion is met
      return optimal solution

    $k = k + 1$
End
```

As the algorithm progresses the particles “fly” through the solution domain as dictated by the velocity. The global best particle attracts each particle causing a swarming effect. As the method progresses the particles converge on an optimum.

Convergence can be achieved numerous ways depending on the problem, for example if the global optimal cost is known, the algorithm should terminate when this cost is achieved. Alternatively the algorithm can be run for a specified number of iterations, taking the final global best particle as the optimal. For most problems the particles will begin to converge on an optimal solution and the algorithm should terminate when the error term, or distance from the global best particle falls below a specified value.

PSO has many desirable properties including the ability to handle poor initial guesses, a relatively simple structure, few tuning parameters, and the possibility of achieving globally optimal solutions. PSO does not easily stick at local minima as most gradient methods do. The stochastic nature of PSO makes the decision of how to parameterize the control much more limited especially in optimal control problems. Often the control is parameterized into a number of discrete points applied at specific time instances; continuity between points is often implied. While most gradient methods easily handle implied control continuity, stochastic algorithms do not.

DIRECT SHOOTING

Direct shooting is a common method for solving optimization problems of the Problem 2 form [1,2]. The cost function, or fitness of a solution is evaluated in the same way as for PSO, forward propagation of the system states according to Equation (73), followed by evaluation of the cost given by Equation (72). The system cost is then minimized by using a gradient method such as Newton's method, or an approximate Newton method. Gradients can be evaluated numerically by finite difference approximation of the cost function, or analytically by solving for the co-states with the adjoint equations [1]. Many Newton based techniques are available for solving problems of the form Problem 2, a diagonally scaled conjugate direction method was selected for this application due to the low computation per iteration, and robust behavior on difficult problems. Simple bounds are treated by a projection method as proposed in [11]. A diagonally scaled version of the conjugate projected direction method developed by Schwartz [10] is used in this work. The algorithm is briefly described below, readers are referred to [10] for a complete development.

Gradient methods are iterative; they require an initial guess of the solution parameters which are updated by new parameters with a lower cost. New parameters are identified by using gradients to search in the direction of decreasing cost. Successive updates of the solution parameters eventually result in a local minimum provided that each iteration sufficiently decreases the cost. Projected gradient methods can be described by Equation (85).

$$\vec{v}^{k+1} = \vec{v}^{k}(\alpha^{k}, \vec{d}^{k}) = \left[\vec{v}^{k} + \alpha^{k} \cdot \vec{d}^{k} \right]_{\#} \tag{85}$$

where $\vec{v}$ is the vector of unknown parameters, $\vec{d}$ is the search direction, α is a line search parameter, the superscript k denotes the k^{th} iteration, and $[\cdot]_{\#}$ denotes projection onto the simple bounds defined in Equation (84). The line search parameter is selected to insure that the cost is sufficiently reduced in each iteration. The necessary conditions for a minimum of Problem 2 are given in Equation (86).

$$\begin{aligned}&\left[\nabla J(X(v^*),v^*)\right]_I = \vec{0}\\&\left[\nabla J(X(v^*),v^*)\right]_{A_l} \geq \vec{0}\\&\left[\nabla J(X(v^*),v^*)\right]_{A_u} \leq \vec{0}\end{aligned} \tag{86}$$

where the superscript $*$ denotes the optimal values of the unknown parameters, subscript I denotes the set of parameters with inactive bound constraints, subscript A_l denotes the set of parameters with active lower bound constraints, and subscript A_u denotes the set of parameters with active upper bound constraints. A direct shooting projected gradient algorithm is described in Table 2-4 [10].

Table 2-5: Direct Shooting Algorithm (DS)

```
Given: $N_{cg}$, $k$, $\vec{v}^k$

    For $N_{cg}$
        Step 1: Compute gradient $g^k = \nabla J(\vec{v}^k)$
                Identify Active and Inactive Constraint set
                ($A_l^k$, $A_u^k$, and $I^k$)
                If Convergence Criterion is Met
                    stop

        Step 2: Calculate the search direction $\vec{d}^k$

        Step 3: Calculate the distance $\alpha$ to move along $\vec{d}^k$
                update $\vec{v}^{k+1} = \left[\vec{v}^k + \alpha\vec{d}^k\right]_\#$
                update iteration index $k = k+1$
    End
```

where N_{cg} is the number of gradient steps to be taken.

The search direction in algorithm DS can be selected as the steepest descent direction, Newton or Quasi-Newton direction, or conjugate gradient direction. The conjugate gradient direction is described by Equations (87) through (89).

$$\mu_k = \frac{\left\langle g^k, D(g^k - g^{k-1}) \right\rangle_{I^k}}{\left\langle g^k, Dg^k \right\rangle_{I^k}} \tag{87}$$

$$\tilde{d}_i^k = -D_{i,j} g_i^k + \mu_k d_i^{k-1} \tag{88}$$

$$d_i^k = \begin{cases} \tilde{d}^k & \text{if } \left\langle \tilde{d}^k, Dg^k \right\rangle_{I^k} \le \sigma_1 \left\| Dg^k \right\|_{I^k} \text{ and } \left\| \tilde{d}^k \right\|_{I^k} \le \sigma_2 \left\| Dg^k \right\|_{I^k} \\ -D_{i,j} g_i^k & \textit{otherwise} \end{cases} \tag{89}$$

where the matrix D is a diagonal positive definite scaling matrix, $\langle\cdot\rangle_{I^k}$ denotes a dot product operation over the set of inactive constraints, and $\|\cdot\|_{I^k}$ denotes the 2-norm operation over the set of inactive constraints. The condition in Equation (89) insures that $\left\|d^k\right\|_{I^k}$ is bounded below by $\sigma_1 \left\|g^k\right\|_{I^k}$ and bounded above by $\sigma_2 \left\|g^k\right\|_{I^k}$, and that d^k does not become orthogonal to g^k.

In step 3 the first α_a^k which satisfies a modified Armijo rule given in Equation (90) is accepted.

$$J(\vec{v}^k(\alpha^k, \vec{d}^k)) - J(\vec{v}^k) \le \eta \left\{ \alpha^k \left\langle D\vec{g}^k, \vec{d}^k \right\rangle_{I_k} - \left\langle D\vec{g}^k, \vec{v}^k - \vec{v}^k(\alpha^k, d^k) \right\rangle_{A_k} \right\} \tag{90}$$

where $\alpha_a^k = \beta^m$, and m is the smallest integer such that Equation (90) is satisfied.

A cubic polynomial interpolate of $J(\alpha)$ between 0 and α_a is then minimized. Convergence of algorithm DS is met when Equations (86) are approximately satisfied. Development and comparison with other Newton like methods can be found in [10].

HYBRID SOLVER

A hybrid solver composed of PSO and DS optimization strategies is proposed as a fast and more robust method than either technique individually. The solution quality and convergence rate of DS may vary significantly with initial guess, on the other hand, PSO is robust to initial guess but exhibits a slow convergence rate. Careful combination of these methods results in a robust algorithm that performs nearly as fast as algorithm DS alone. Success of the combination solver design lies in the careful balance between convergence speed and solution space coverage. The following flow diagram shows the structure of the hybrid method.

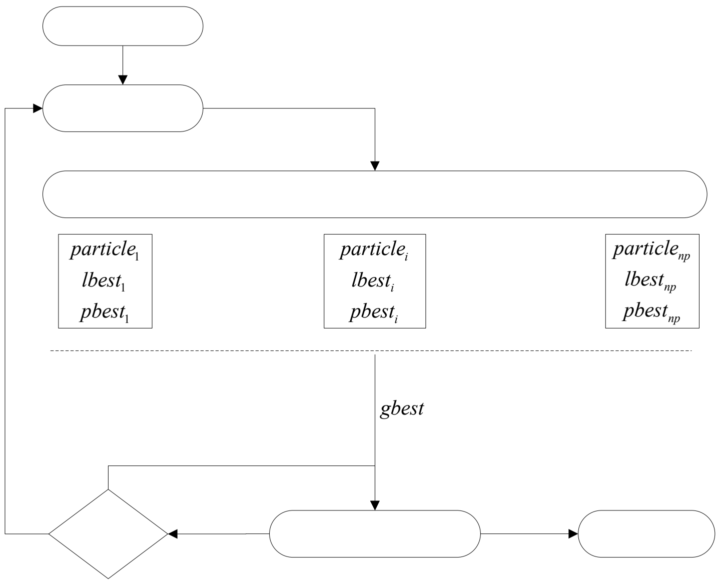
START
Initialize PSO
Partic
$particle_1$ $lbest_1$ $pbest_1$
$particle_i$ $lbest_i$ $pbest_i$
$particle_{np}$ $lbest_{np}$ $pbest_{np}$
$gbest$

Figure 2-2: Structure of Hybrid Optimal Search Method (HS)

Algorithm HS does not require an initial guess of the solution, however performance can be significantly increased if the initial search is limited to a subset of the feasible parameter space. Function "*Initialize PSO"* in Figure 2-2 accepts a vector of parameters, and a percentage of the search space to cover. The particles are then randomly initialized within a fixed region equal to the desired percentage of the total feasible space, centered about a "best guess" vector of parameters. A hybrid algorithm is described in Table 2-6.

Table 2-6: Hybrid Optimal Search Algorithm (HS)

```
Given: Initial guess $\vec{v}^0$, $T_{\%}, k = 0$
For $N_{HS}$

  Step 1: Perform Algorithm $PSO(N_{PSO}, T_{\%}, k, \vec{v}^k)$
          update $\vec{v}^k = gbest$

  Step 2: Perform Algorithm $DS(N_{PG}, k, \vec{v}^k)$
    Decision 1:
      If algorithm cost is sufficiently high
        update $T_{\%}$
        update $N_{PSO}$
        return to Step 1
      Else
        return to Step 2

End
```

where N_{HS} is the number of total iterations, N_{PSO} is the number of PSO iterations, N_{PG} is the number of gradient steps to be taken, and $T_{\%}$ is the size of the search region to be explored by PSO in percentage of the total feasible space. Search space coverage by algorithm PSO is updated based on performance of algorithm DS, Equation (91).

$$T_{\%} = T_{\%} \cdot {}^{J_k}\!/_{J_{PG_0}} \tag{91}$$

where J_k is the current cost, and J_{PG_0} is the cost at the first evaluation of algorithm DS. The number of PSO iterations (N_{PSO}) can also be updated at step 2.

GLIDER MODEL

The glider is modeled as a 3 degree of freedom (DOF) point mass described by three inertial position coordinates. Forces acting on the mass include aerodynamic lift drag, and gravity. Control surfaces are taken to be bank angle and lift coefficient. In the equations that follow the ground frame is assumed to be a satisfactory reference frame.

$$\begin{Bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{Bmatrix} = \begin{bmatrix} c_{\psi_{pm}} c_{\theta_{pm}} & c_{\phi_{pm}} s_{\theta_{pm}} c_{\psi_{pm}} + s_{\phi_{pm}} s_{\psi_{pm}} \\ s_{\psi_{pm}} c_{\theta_{pm}} & c_{\phi_{pm}} s_{\theta_{pm}} s_{\psi_{pm}} - s_{\phi_{pm}} c_{\psi_{pm}} \\ -s_{\theta_{pm}} & c_{\theta_{pm}} c_{\phi_{pm}} \end{bmatrix} \begin{Bmatrix} -D/m \\ -L/m \end{Bmatrix} - \begin{Bmatrix} 0 \\ 0 \\ g \end{Bmatrix} \tag{92}$$

where, x y and z are the position vector components described in the inertial frame. The common shorthand for trigonometric functions is used throughout the paper ($\sin\theta = s_\theta$). Forces and angles are described by Equations (93) through (97).

$$L = \tfrac{1}{2} \rho S V_{pm}^2 C_L \tag{93}$$

$$D = \tfrac{1}{2} \rho S V_{pm}^2 \left(C_{do} + k C_L^2 \right) \tag{94}$$

$$V_{pm}^2 = (\dot{x} - W_x)^2 + (\dot{y} - W_y)^2 + (\dot{z} - W_z)^2 \tag{95}$$

$$\sin_{\theta_{pm}} = (W_z - \dot{z}) / V_{pm} \tag{96}$$

$$\tan_{\psi_{pm}} = (\dot{y} - W_y) / (\dot{x} - W_x) \tag{97}$$

where W_x, W_y and W_z are wind velocities described in the inertial frame. Model parameters are selected to represent the performance of a typical micro air robot with a small payload, and are given in Table 2-7.

Table 2-7: Glider Model Parameters

Mass (kg)	1.55
Aerodynamic Area (m^2)	0.3406
Max Lift to Drag	38
Profile Drag Coefficient	0.010

WIND STRUCTURE MODELS

The basic scenario investigated in this work involves thermals and wind gradients. Optimal trajectories in wind gradients alone are classified as dynamic soaring, and have been investigated by numerous authors including Sachs, Zhao, and Lissaman [12,13,14]. Static and semi-dynamic soaring in stationary thermals has been studied by Wharington [15]. Wharington argues that many local minima exist in static soaring, and shows that, under certain conditions, maneuvers classified as semi-dynamic result in larger altitude gain per cycle than the typical static trajectories.

A thermal model developed by Allen [16] is used in this investigation. It is designed based on balloon and surface measurements made at Desert Rock, Nevada. The general shape is described in Figure 2-3.

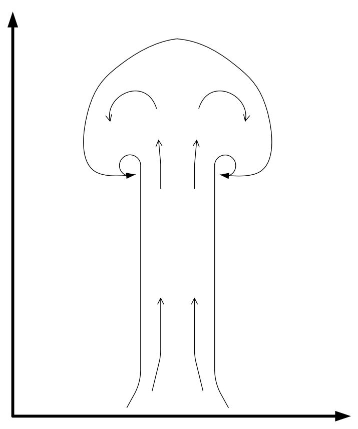

Figure 2-3: Thermal Model Shape

The lift distribution is a revolved trapezoid given by Equation (98).

$$w_{th} = w_{peak}\left(\frac{1}{1+\left|k_1\frac{r}{r_2}+k_3\right|^{k_2}}+k_4\frac{r}{r_2}+w_D\right) \tag{98}$$

where k_1 through k_4 are shape parameters, r is the distance from the thermal core, r_2 is the thermal outer radius, w_D is the downdraft velocity, and w_{peak} is the velocity at the center of the thermal. Downdraft wind velocity is determined by Equations (99) and (100).

$$w_1 = \begin{cases} \dfrac{-\pi}{6}\sin(\pi r/r_2) & for\ r_1 < r < r_2 \\ 0 & else \end{cases} \tag{99}$$

$$w_D = \begin{cases} 2.5 w_1 \left(h/h_i - 0.5\right) & for\ 0.5 < h/h_i < 0.9 \\ 0 & else \end{cases} \tag{100}$$

where h is the altitude in the thermal and h_i is the height of the thermal. Environmental sink is determined by enforcing conservation of air mass in the surrounding area, the transition between the thermal and surrounding sink is smoothed according to Equation (101).

$$w_{eff} = w_{th}(1 - w_e/w_{peak}) + w_e \tag{101}$$

where the environmental sink is w_e, and w_{th} is the vertical wind velocity. Typical values of peak velocity, outer radius, and thermal height are given in [16].

A wind gradient model developed by Zhao [13] is used in this study. The general form allows linear, exponential and logarithmic gradient profiles to be modeled. Horizontal wind is given as a function of altitude by Equation (102).

$$w_{hg} = w_g \left[A(h - h_{\min}) + \frac{1-A}{h_{\max} - h_{\min}}(h - h_{\min})^2 \right] + w_{h\min} \tag{102}$$

Where w_g is the average slope, $h_{\min}$ is the height the gradient begins, $h_{\max}$ is the height the gradient ends, $w_{h\min}$ is the minimum horizontal wind velocity, and A is the shape parameter in the range 0 to 2. The gradient profile is shown for different values of A in Figure 2-4.

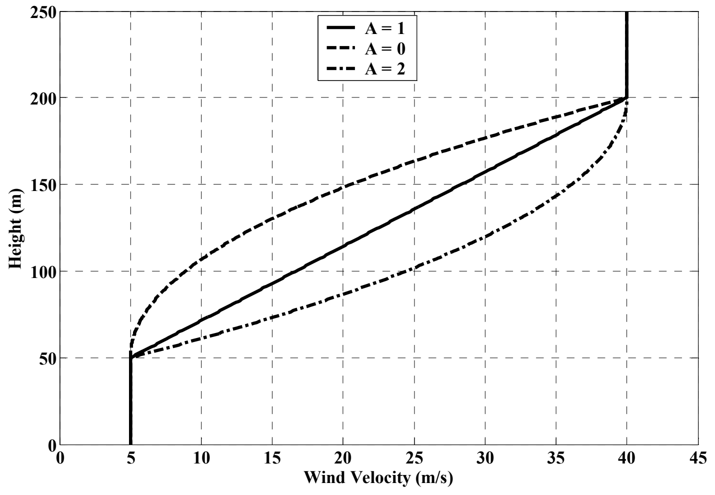

Figure 2-4: Wind Gradient Profile

COMPARISON RESULTS

The first scenario considered consists of a single thermal located at the origin as shown in Figure 2-5. Wind model values are given in Table 2-8.

Table 2-8: Thermal Model Parameters

Peak Velocity (m/s)	1.55
Outer Radius (m)	41.2
k1	1.93
k2	2.63
k3	-0.015
k4	-0.050
k5	0.0015
Environmental Sink (m/s)	-0.10

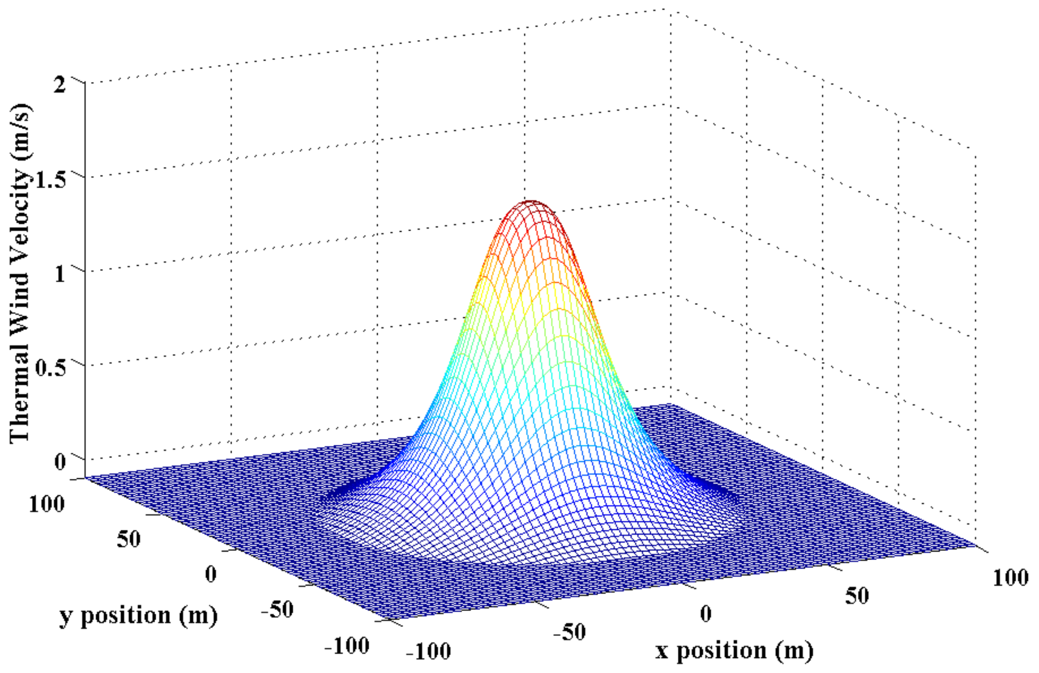

Figure 2-5: Thermal Model Profile

A closed trajectory with maximal altitude gain and zero speed loss is sought subject to the bounds given in Equations (103) through (105).

Control Bounds:

$$\begin{gathered}|\phi| \leq 130^{\circ} \\ |\dot{\phi}| \leq 45^{\circ} \\ -0.2 \leq C_L \leq 1.2 \\ |\dot{C}_L| \leq 0.5\end{gathered} \tag{103}$$

Initial State Vector Bounds:

$$\begin{gathered}3.0 \leq V_1 \leq 30.0 \\ |\psi_1| \leq 90^{\circ} \\ |\theta_1| \leq 50^{\circ} \\ 0.0 \leq y_1 \leq 80.0 \\ 2.0 \leq t_f \leq 20.0\end{gathered} \tag{104}$$

Path Constraints:

$$3.0 \leq V_w \leq 30.0 \tag{105}$$

where V_w is the glider air speed. Initial heading angle, pitch angle, velocity, and position along the y axis are sought in the optimization problem. Initial altitude and x position are given as 50 and 0 meters respectively.

Both search algorithms are exercised on this scenario resulting in two similar locally optimal trajectories shown in Figures 2-6 through 2-9. Initial runs indicated that a constant maximum lift coefficient was always optimal in this scenario. For the first scenario, lift coefficient is represented by a constant value and bank angle is split into fifteen polynomials. Bank angle and bank rate at the initial time are constrained to be equal to the bank angle and bank rate at the final time.

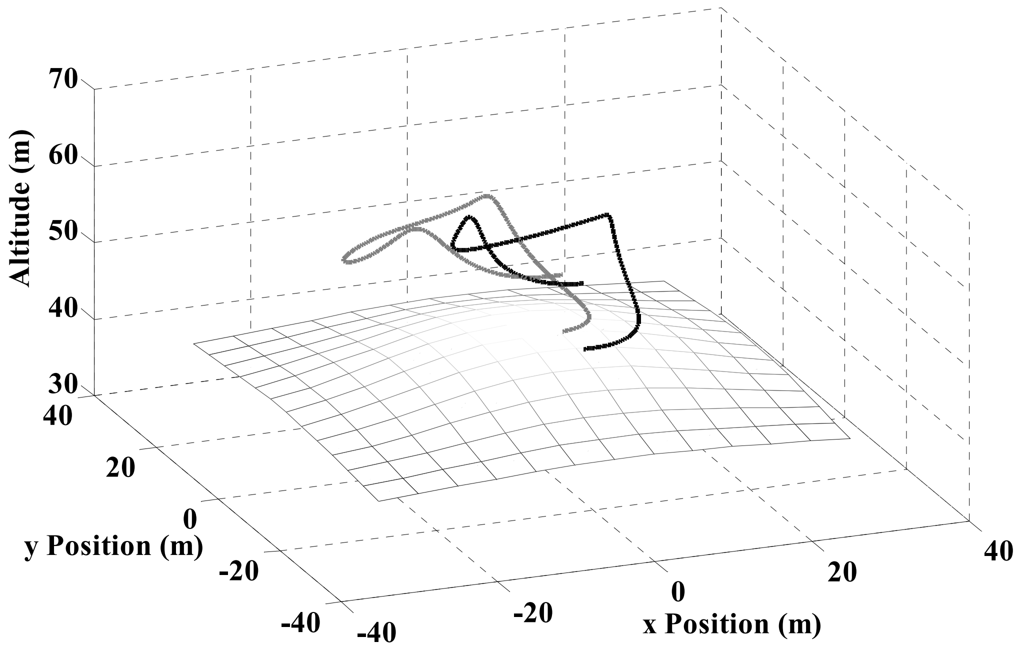

Figure 2-6: Single Thermal Scenario Optimal Closed Trajectory

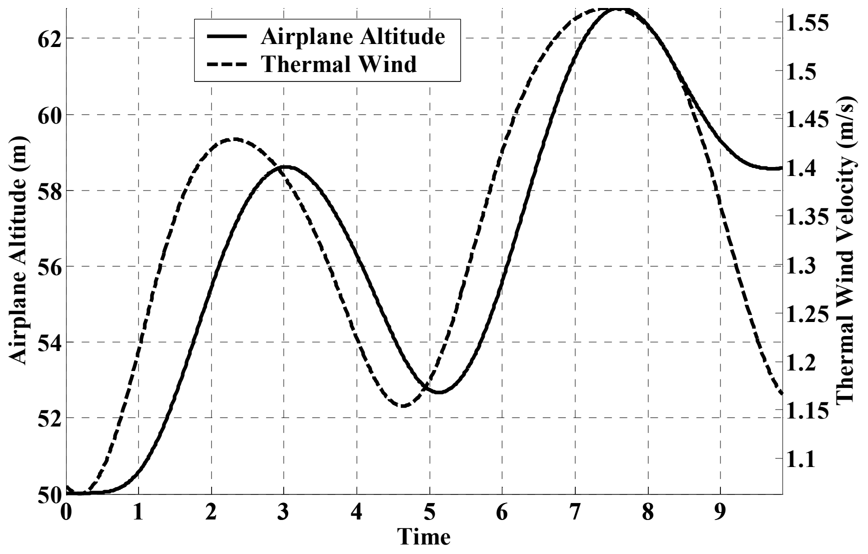

Figure 2-7: HS Solution Altitude and Wind Velocity Time History

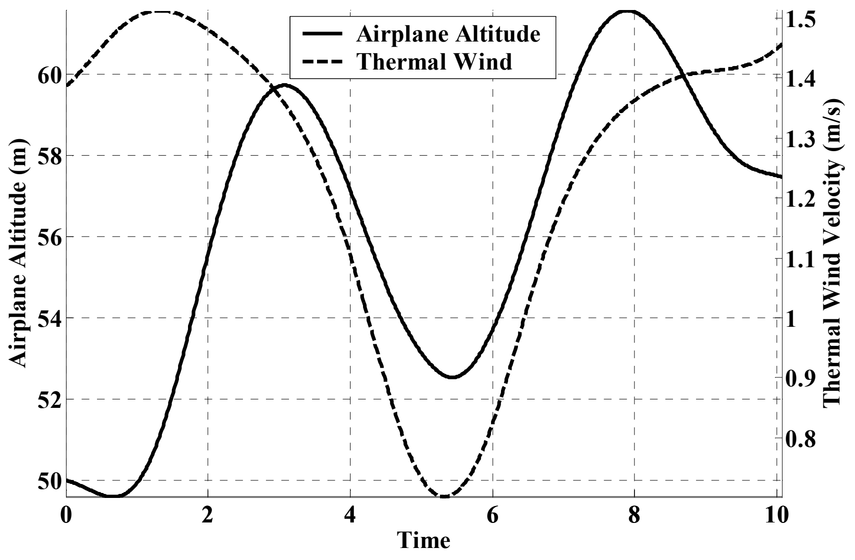

Figure 2-8: DS Solution Altitude and Wind Velocity Time History

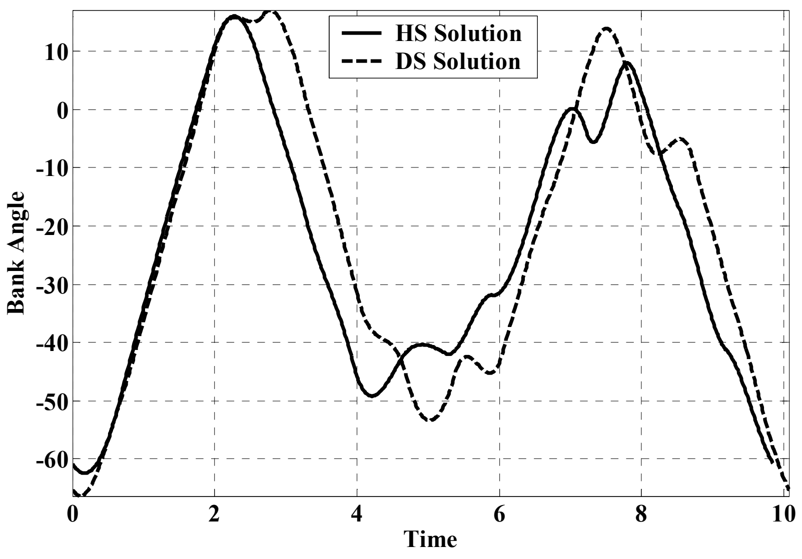

Figure 2-9: Single Thermal Scenario Bank Angle Time History

Solution time between the methods is very similar. However the hybrid solver finds a better solution in this case. Performance of the two algorithms is compared in Table 2-9.

Table 2-9: Method Performance Comparison

Method	Solution Time (s)	Altitude Gain (m)
HS	27	8.6
DS	26	7.5

Both methods are initialized with the same values. Parameters used in algorithm HS are shown in Table 2-10.

Table 2-10: HS Algorithm Parameters

Initialization Percentage $T_{\%}$	5.0%
Initial Number of PSO N_{PSO}	40
Number of DS N_{PG}	20
Number of Subsequent PSO N_{PSO}	30

Values in Table 2-10 were selected based on extensive experimentation, general selection guidelines are discussed in the conclusions chapter.

The second scenario involves a single thermal and a horizontal wind gradient, wind model values are given in Table 2-7 and 2-8. Typically the formation of thermals is significantly degraded as the horizontal wind increases beyond some level. Allen [16] uses 12.87 m/s as an upper limit to the formation of thermals. We limit our investigation to horizontal winds below this threshold. Horizontal wind gradient model parameters are given in Table 2-11.

Table 2-11: Wind Gradient Model Parameters

Minimum Height (m)	40.0
Maximum Height (m)	70.0
Maximum Wind Velocity (m/s)	9.0
Minimum Wind Velocity (m/s)	4.0
Shape Parameter A	1.0

Initial altitude and x position are given as 50 and 0 meters respectively. Bank angle is represented with 15 polynomials, and lift coefficient is represented by 10 polynomials. The total number of unknown parameters is 55.

The addition of a horizontal wind gradient causes problems for convergence with algorithm DS for some initial guess', however algorithm HS seems to be robust to this addition. An example of this difficulty is demonstrated with this scenario. Both algorithms are initialized with the same values, DS fails to converge while HS converges in a reasonable time. The resulting trajectory is shown in Figures 2-11 through 2-14.

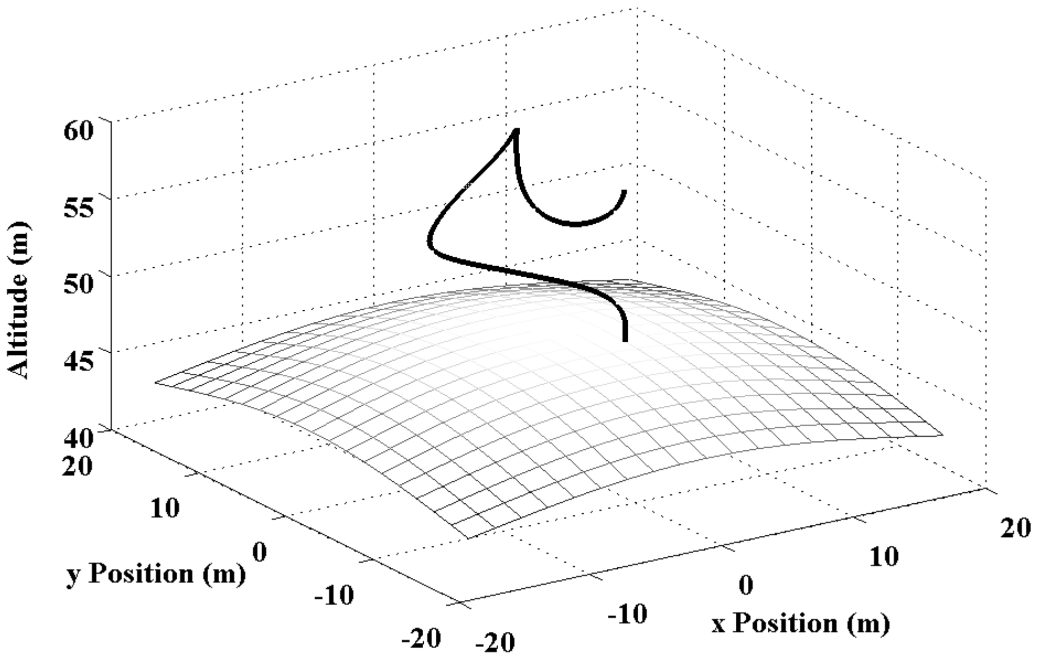

Figure 2-10: Thermal and Wind Gradient Scenario Optimal Trajectory

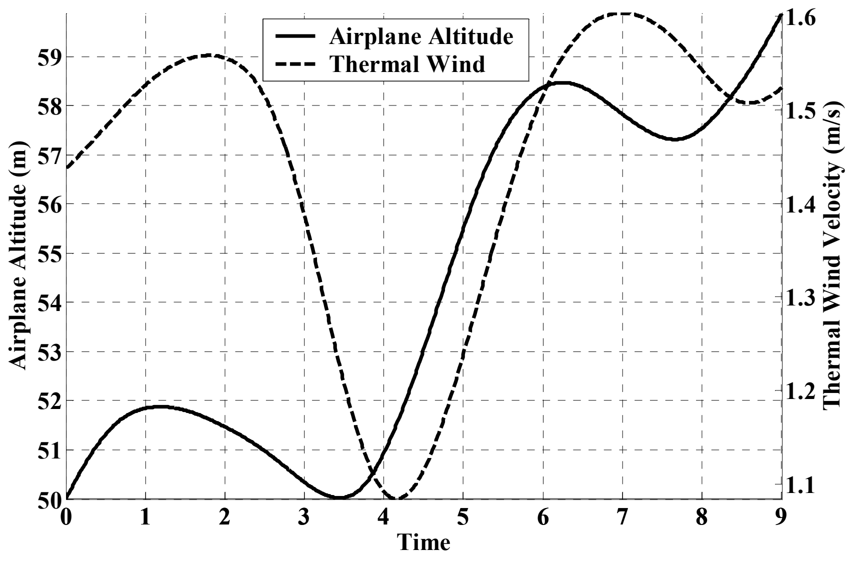

Figure 2-11: Thermal and Wind Gradient Scenario Altitude Time History

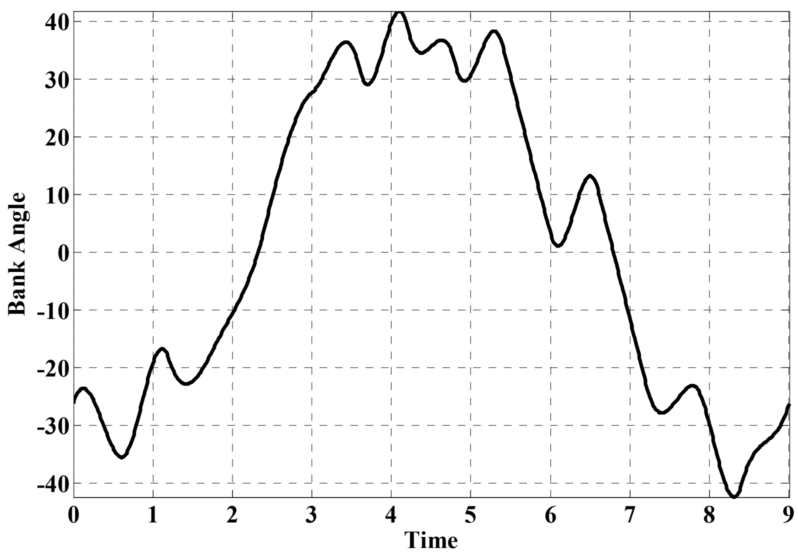

Figure 2-12: Thermal and Wind Gradient Scenario Bank Angle Time History

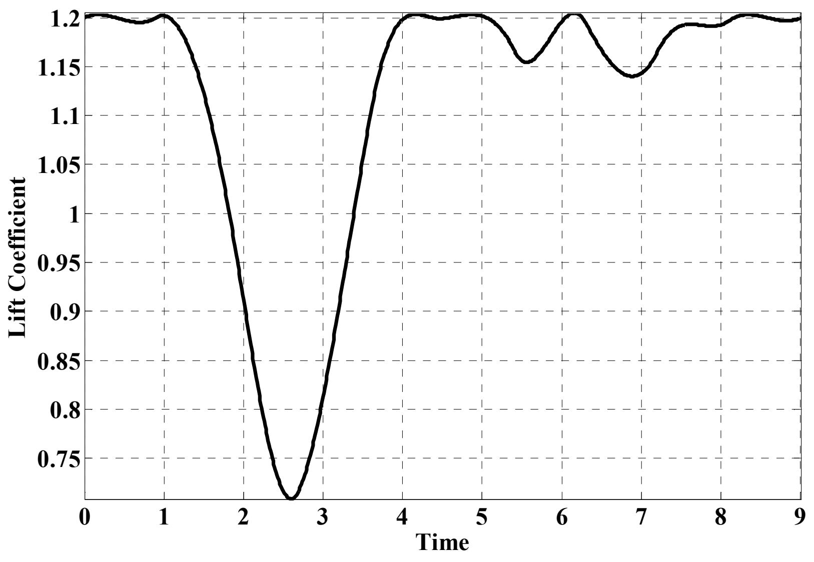

Figure 2-13: Thermal and Wind Gradient Scenario Lift Coefficient Time History

Interestingly, the addition of a wind gradient results in a larger altitude gain, however it is unknown if either trajectory is globally optimal. Performance of both algorithms is compared in Table 2-10.

Table 2-12: Thermal and Wind Gradient Scenario Algorithm Performance

Method	Solution Time (s)	Altitude Gain (m)
HS	57.8	9.9
DS	Failure	NA

The addition of a wind gradient seems to increase the frequency of failed convergence for algorithm DS, yet convergence for algorithm HS is only slowed. This may be important when determining an initialization scheme for DS, care must be taken to insure initial guesses are robust to seemingly small changes in the problem.

The optimal trajectory in this scenario takes the form of a figure eight centered on the thermal from the top view. The bank schedule required in this scenario is different in magnitude and form from the single thermal scenario, and results in a larger altitude gain.

CONCLUSIONS

Performance of algorithm HS depends heavily on the size of the initial search space, and number of initial PSO iterations. A small search space coverage greatly enhanced convergence speed where the initial guess was reasonable. Similarly, a small search space coverage required fewer PSO iterations to reduce the cost significantly. Where the initial guess was poor a larger search space and more PSO iterations were required. Parameters specific to PSO and DS were selected to optimize performance of each individual algorithm. The number of neighbors in PSO was selected at a moderate level, to insure the algorithm would thoroughly explore the desired search region. Values used in this work are given in Table 2-10.

Table 2-13: PSO Algorithm Parameters

Number of Particles	25
Number of Neighbors	8
Inertia Weight Schedule	0.9-0.4
Acceleration Parameters C_1, C_2	2.0, 2.0

The hybrid solver was found to be marginally slower than direct shooting for the single small thermal soaring problem. However a slightly better solution was found. When a horizontal wind gradient is added, the direct shooting method fails to converge but the hybrid method converges.

REFERENCES

[1] Betts, J., "Survey of Numerical Methods for Trajectory Optimization," Journal of Guidance, Control and Dynamics, Vol. 21, No. 2, 1998.

[2] Hull, D., "Conversion of Optimal Control Problems into Parameter Optimization Problems," Journal of Guidance, Control, and Dynamics, Vol. 20, No. 1, 1997.

[3] Eberhart, R., and Kennedy, J., "A New Optimizer Using Particle Swarm Theory," Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya, Japan, pp. 39-43, 1995.

[4] F. van den Bergh, "An Analysis of Particle Swarm Optimizers," PhD thesis, Department of Computer Science, University of Pretoria, South Africa, 2002.

[5] F. van den Bergh, "A New Locally Convergent Particle Swarm Optimiser," IEEE International Conference on Systems, Man and Cybernetics, Vol. 3, pp. 6, 2002.

[6] X. Hu, "Particle Swarm Optimization," July 12, 2006 http://www.swarmintelligence.org/tutorials.php

[7] J. M. Renders, and S. P. Flasse, "Hybrid Methods Using Genetic Algorithms for Global Optimization," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 26, No. 2, 1996.

[8] M. Syrjakow, and H. Szczerbicka, "Combination of Direct Global and Local Optimization Methods," IEEE International Conference on Evolutionary Computation, 1996.

[9] Y. Shi, R. Eberhart, "A Modified Particle Swarm Optimizer," IEEE International Conference on Evolutionary Computation, Anchorage, AK, May 1998.

[10] A. Schwartz, "Theory and Implementation of Numerical Methods Based on Runge-Kutta Integration for Solving Optimal Control Problems," Ph.D Thesis, University of California at Berkeley, 1996.

[11] D. Bertsekas, "Nonlinear Programming," Athena Scientific, Belmont, Mass., 2nd edition, 1999.

[12] G. Sachs, and O. da Costa, "Optimization of Dynamic Soaring at Ridges," AIAA Atmospheric Flight Mechanics Conference and Exhibit, Austin, Texas, 2003.

[13] Y. J. Zhao, "Optimal Patterns of Glider Dynamic Soaring," Optimal Control Applications and Methods, Vol. 25, pp 67-89, 2004.

[14] P. Lissaman, "Wind Energy Extraction by Birds and Flight Vehicles," AIAA 43rd Aerospace Sciences Meeting and Exhibit, Reno NV., 2005.

[15] J. Wharington, and I. Herszberg, "Optimal Semi-Dynamic Soaring," Royal Melbourne Institute of Technology, Melbourne, Australia, October 11, 1998.

[16] M. Allen, "Updraft Model for Development of Autonomous Soaring Uninhabited Air Vehicles," 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 9-12, 20.

3 RECEDING HORIZON TRAJECTORY OPTIMIZATION FOR REAL TIME AUTONOMOUS SOARING

Jason Kyle and Mark Costello

RECEDING HORIZON TRAJECTORY OPTIMIZATION FOR REAL TIME AUTONOMOUS SOARING

Jason Kyle[υ]
Department of Mechanical Engineering
Oregon State University
Corvallis, OR.

Mark Costello[β]
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, Georgia

ABSTRACT

Extracting energy from naturally occurring wind structures with a small autonomous air vehicle is considered. A receding horizon trajectory optimization strategy is presented for real time application. An initial guess of the optimal trajectory is improved with several Newton iterations of a nonlinear optimization problem 20 times a second. Several scenarios involving a small thermal and horizontal wind shear are investigated. The resulting trajectories converge on a repeatable semi-dynamic cycle after a short time. Significant altitude gain is achieved in all scenarios. Sensitivity to wind and glider model errors is investigated.

[υ] Graduate Research Assistant, Member AIAA
[β] Sikorsky Associate Professor, School of Aerospace Engineering, Associate Fellow AIAA

NOMENCLATURE

$\vec{u}_i$	= ith control vector
$\vec{x}_i$	= ith state vector
$J(\vec{u})$	= scalar cost function
L	= index of final state vector
$g_i(\vec{x}_i)$	= penalty function on ith state vector
$q_i(\vec{u}_i)$	= penalty function on ith control vector
D	= diagonal positive definite penalty matrix
$G(\vec{x}_i, \vec{u}_i)$	= discrete governing dynamic equations
$\vec{u}_{\max}, \vec{u}_{\min}$	= upper and lower bounds on the control vector
$\chi_{\min}, \chi_{\max}$	= simple bounds on nonlinear inequality function
$\chi(\vec{x}_i)$	= ith nonlinear inequality function
C	= control extraction matrix
$\mathcal{L}$	= Lagrangian function of the nonlinear optimization problem
H_i	= ith discrete Hamiltonian of the nonlinear optimization problem
$\vec{P}_i$	= ith vector of Lagrange multiplier for nonlinear optimization problem
$[\cdot]_{\#}$	= projection onto simple bounds
α^k	= line search parameter
$\vec{s}^k$	= vector of unknown parameters at the kth iteration
$I^{\#}(\vec{u})$	= indices of controls with active bound constraints
ε	= soft boundary parameter for control inequality constraints
$\nabla\mathcal{L}$	= gradient of the Lagrangian
$\nabla^2\mathcal{L}$	= Hessian of the Lagrangian
$\tilde{\mathcal{L}}(\delta\vec{u})$	= Lagrangian function of the quadratic programming problem

$\tilde{H}_i$	= ith discrete Hamiltonian of the quadratic programming problem
$\delta\vec{u}_i^k$	= ith control vector variation at the kth iteration, $\vec{u}_i^k - \vec{u}_i$
$\vec{\lambda}_i$	= ith co-state vector of the quadratic programming problem
F_B	= interior point boundary function for state inequality constrains
c	= scalar weight parameter for interior point boundary function
x, y, z	= position vector components of the glider expressed in the inertial frame
V	= glider airspeed magnitude
ψ, θ, ϕ	= glider path heading, pitch and bank angles
W_x, W_y, W_z	= wind velocity components expressed in the inertial frame
m	= mass of the glider
g	= gravitational acceleration constant
ρ	= density of air
S	= glider model wing area
C_L	= glider model lift coefficient
C_{D_0}	= glider model profile drag coefficient
k	= glider model induced drag factor
t_{ph}	= controller prediction horizon
V^*	= nominal airspeed
$\overline{x}, \overline{y}, \overline{z}$	= non-dimensional position vector components of the glider mass expressed in the inertial frame
τ	= non-dimensional time parameter
$\overline{V}$	= non-dimensional glider airspeed
$\overline{g}$	= non-dimensional gravitational acceleration constant
D_{p_c}	= non-dimensional dynamic pressure coefficient
$\overline{E}_{sp}$	= non-dimensional specific wind relative energy

$\overline{E}'_{sp}$	= non-dimensional specific wind relative energy rate
$\partial_x, \partial_y, \partial_z$	= partial derivative with respect to x, y, and z
r_g	= distance of the glider from the thermal core
$\overline{x}_T, \overline{y}_T$	= location of the thermal core in the inertial frame
e_T	= thermal elliptical shape parameter
$\overline{w}_T$	= vertical wind velocity at the core of the thermal
r_T	= radius of the thermal
$\overline{f}$	= thermal model decay shape parameter
$\overline{w}_d$	= environmental sink wind velocity
w_{hg}	= horizontal wind velocity
β	= slope of the wind gradient
a	= horizontal wind gradient shape parameter
$h_{\min}$	= minimum altitude of the wind gradient
$h_{\max}$	= maximum altitude of the wind gradient
$w_{h\min}$	= minimum horizontal wind velocity

INTRODUCTION

Micro air robots are small, autonomous, intelligent aircraft designed to focus on a specific task. The range of applications envisioned for future micro air robots in both the civilian and military sectors is truly staggering. Micro air robots may be used by environmentalists for detailed wildlife monitoring and surveying tasks. Micro air robots could fly through factory smokestack emissions to measure released chemical concentrations. With gradient sensors and flight control system feedback, micro air robots could map the size and shape of hazardous clouds and provide real time tracking of their location. Forestry management could be aided by sending micro air robots into remote areas of a forest that are difficult to access to gather important forest health and growth data. Other applications include monitoring concentrations of chemical spills and measuring ammonia concentration in agriculture, to name just a couple.

The potential of micro air robots is astonishing, yet significant technical obstacles must be overcome to realize this potential. The Achilles heel of micro air robots is power required for mobility. Micro air robots consume a significant amount of power just to remain aloft. When considering practical micro air robot configurations that carry sensors, power requirement problems become more acute. These power requirements curtail the feasibility of micro air robots for many of the potential missions mentioned above. To remedy this situation, many research groups are actively engaged in research and development on small, low weight, high power output propulsion technologies. An alternate and complementary concept for powering micro air robots is to harvest energy from the environments in which they fly. In straight and level aircraft flight, atmospheric wind updrafts rotate the relative aerodynamic velocity vector downward, causing drag to point aft and slightly upward and lift to point up and slightly forward. When the atmospheric wind updraft is sufficiently large, straight and level flight and even climbing flight is possible without power.

Thermals, hill lift, and mountain waves are examples of naturally occurring wind structures with a large enough vertical velocity component to support static soaring. A typical static soaring strategy involves remaining in the vertical updraft as long as possible, for example constant circling in a thermal. Trajectories tend to be relatively static, with a nearly constant bank angle, airspeed, and path pitch angle. Due to the simplicity of the trajectory, this mechanism can be very successful where the updraft region is large. As the size of the updraft region decreases larger bank angles are required to remain in the updraft resulting in increased airspeed and greater energy loss' due to large drag forces. In this case semi-dynamic trajectories [1] have been shown to perform significantly better by using multiple energy gain mechanisms. Semi-dynamic soaring requires larger variations in airspeed, path pitch angle, and bank angle however energy gains are typically larger than with static soaring.

Several research groups have investigated optimal static and dynamic soaring trajectories. Allen used a simplified airplane model and thermal measurements in a specific geographic region to estimate flight time gains assuming a soaring controller is available [2]. He found that endurance of a glider could be increased from 2 hrs. up to a maximum of 14 hrs. in the region investigated, while the average flight time gain over a year was found to be 8.6 hrs.. Wharington used a direct shooting method to find a closed cycle which results in the largest altitude gain where single and multiple small thermals are known to exist [1]. Wharington found that semi-dynamic soaring is optimal in small thermals.

Sachs investigated repeatable optimal dynamic soaring trajectories in the wind shear at a ridge [3]. He found several trajectories capable of sustaining flight in a realistic wind shear, and also presented minimum shear required to sustain flight. Lissaman further investigated the problem of dynamic-soaring at a ridge, and the oceanic boundary layer. He developed simplified point mass equations, and derived general lower bounds for the wind shear required to sustain flight [4]. Zhao developed a simple wind gradient model capable of representing an exponential, logarithmic, or linear wind gradient [5]. He used direct collocation to investigate repeatable dynamic-soaring trajectories including a minimum cycle time problem. Zhao and Qi investigated the dynamic-soaring problem while minimizing the power required for maintaining flight [6]. Goto investigated dynamic-soaring trajectories in a linear wind gradient using direct collocation [7]. He addressed the issue of control discontinuity inherent in the solution to discrete optimal control problems.

To estimate trajectories that maximize energy gain; trajectory optimization algorithms are typically employed. Betts gives a summary of some common trajectory optimization methods in [8]. Most methods can be classified as either direct or indirect. Indirect methods use the calculus of variations to derive the first order necessary conditions for an extremum. The result is a two point boundary value problem which is then solved for the optimal solution. An example of an indirect solver using finite elements in time is by Warner and Hodges [9].

Direct methods approach the trajectory optimization problem by first converting from continuous variables to discrete parameters. Parameter optimization or nonlinear programming methods are then used to solve the discrete optimization problem. Hull provides an overview of typical methods for this conversion [10]. Many methods exist for solving the resulting parameter optimization problem Berteskas provides an introduction to standard nonlinear programming techniques in [11].

Collocation is one common method of converting a continuous optimization problem into a discrete parameter optimization problem. An implicit integration technique equivalent to Simpson's 1/3 rule is used to represent the governing equations of motion. Hargraves and Paris give a detailed overview of this collocation in [12]. Conway and Larson show how some of the control parameters can be removed in direct optimization resulting in a reduced problem which they show can be solved more efficiently. An implicit Euler rule is used to represent the governing dynamics [13]. Solving an equivalent reduced order system is considered by Petit, et. al. The reduced system is solved by collocation, and results indicate the solution speed can be increased [14]. A direct shooting method uses an explicit integration technique to represent the governing equations of motion. Schwartz developed a direct shooting method for optimal control problems in [15].

Typically any of the above methods require significant computation, and are used for off line investigation. There are many methods that make significant approximations to the problem which reduce computation time significantly.

Computational requirements and sub-optimality are balanced to obtain a desirable algorithm. Slegers, Kyle, and Costello use a taylor series approximation of the system dynamics to predict the optimal control actuation. Trajectory tracking of a flight vehicle was examined in [16]. A real time approach for trajectory optimization was developed by Yakimenko who approximates system dynamics with a single high order polynomial [17]. Flight tests were conducted for short term spatial maneuvers to validate the technique.

The research detailed in this paper investigates a receding horizon control law for nonlinear trajectory optimization that maximizes glider energy gain over a fixed horizon. An initial guess of the optimal trajectory is improved with several Newton iterations of a nonlinear optimization problem 20 times a second. The glider then executes the resulting trajectory. This is a unique approach to the autonomous soaring problem, and is shown to be capable of real time implementation. An efficient direct trajectory optimization algorithm [11] is extended to handle simple bounds on the control and state vector, and nonlinear inequalities on the states. Robustness to wind and model errors show this method has potential for real time application.

RECEDING HORIZON TRAJECTORY OPTIMIZATION

The optimization problem considered in this work is to determine the control history which minimizes the scalar cost function given by Equation (106), subject to the discrete dynamics given by Equation (107), the control inequality constraints given in Equation (108), and the nonlinear state inequality constraints given in Equation (109).

Table 3-1: Optimization Problem 1 (OP1)

Minimize:

$$J(\vec{u}) = g_L(\vec{x}_L) + q_0(\vec{u}_0) + \sum_{i=1}^{L-1} g_i(\vec{x}_i) + q_i(\vec{u}_i) + \tfrac{1}{2}\Delta\vec{u}_i^T D \Delta\vec{u}_i \quad (106)$$

Subject to:

$$\vec{x}_{i+1} = G(\vec{x}_i, \vec{u}_i) \quad i = 0 \ldots L-1 \quad (107)$$

$$\vec{u}_{\min} \le \vec{u}_i \le \vec{u}_{\max} \quad i = 0 \ldots L-1 \quad (108)$$

$$\chi_{\min} \le \chi(\vec{x}_i) \le \chi_{\max} \quad i = 1 \ldots L \quad (109)$$

where $\Delta\vec{u}$ is defined in Equation (110).

$$\Delta\vec{u}_i = \vec{u}_i - \vec{u}_{i-1} \quad (110)$$

The initial state vector $\vec{x}_0$ and previous control vector $\vec{u}_{-1}$ are given, and the discrete dynamic equations can be solved when the control sequence is given. Problem OP1 arises by converting a continuous optimization problem into a discrete parameter optimization problem using an explicit integration technique on the state equations. A problem of this form is typically solved by a direct shooting method [10].

The algorithm considered in this work restricts coupling in the cost function to the ith control and/or state vector. Problem OP1 is converted to this form by appending the previous control to the state vector, as given in Equation (111).

$$\vec{x}_i = \begin{Bmatrix} \vec{x}_i \\ \vec{u}_{i-1} \end{Bmatrix} \tag{111}$$

Now $\Delta\vec{u}_i$ can be written in terms of the ith state and control vectors given by Equation (112).

$$\Delta\vec{u}_i = \vec{u}_i - C\vec{x}_i \tag{112}$$

Problem OP1 can then be written in the required form given in Table 3-2.

Table 3-2: Optimization Problem 2 (OP2)

Minimize:

$$\begin{aligned} J(\vec{u}) = {} & g_L(\vec{x}_L) + q_0(\vec{u}_0) + \\ & \sum_{i=1}^{L-1} g_i(\vec{x}_i) + q_i(\vec{u}_i) + \tfrac{1}{2}\vec{u}_i^T D\vec{u}_i + \tfrac{1}{2}\vec{x}_i^T C^T DC\vec{x}_i - \vec{x}_i^T C^T D\vec{u}_i \end{aligned} \tag{113}$$

Subject to:

$$\vec{x}_{i+1} = G(\vec{x}_i, \vec{u}_i) \quad i = 0\ldots L-1 \tag{114}$$

$$\vec{u}_{\min} \le \vec{u}_i \le \vec{u}_{\max} \quad i = 0\ldots L-1 \tag{115}$$

$$\chi_{\min} \le \chi(\vec{x}_i) \le \chi_{\max} \quad i = 1\ldots L \tag{116}$$

Although the size of the state vector is extended in OP2, the additional computation is minimal with careful implementation.

Sequential Quadratic Programming Algorithm

Many solution techniques exist for OP2, an efficient sequential quadratic programming algorithm (SQP) is selected for this work. A brief development can be found in [11]. Problem OP2 can be solved by finding a stationary point of the Lagrangian function given in Equations (117) through (119).

$$\mathcal{L} = g_L + \sum_{i=0}^{L-1} \left(H_i - \vec{P}_{i+1}^T \vec{x}_{i+1} \right) \tag{117}$$

$$H_i = g_i(\vec{x}_i) + q(\vec{u}_i) + \tfrac{1}{2}\vec{u}_i^T D\vec{u}_i + \tfrac{1}{2}\vec{x}_i^T C^T DC\vec{x}_i - \vec{x}_i^T C^T D\vec{u}_i + \vec{P}_{i+1}^T\left(\vec{G}_i(\vec{x}_i,\vec{u}_i)\right) \tag{118}$$

$$H_0 = q_0(\vec{u}_0) + \tfrac{1}{2}\vec{u}_0^T D\vec{u}_0 + \tfrac{1}{2}\vec{x}_0^T C^T DC\vec{x}_0 - \vec{x}_0^T C^T D\vec{u}_0 + \vec{P}_1^T\left(\vec{G}_0(\vec{x}_0,\vec{u}_0)\right) \tag{119}$$

where H_i is the discrete Hamiltonian, and $\vec{P}_i$ are the system co-states. A stationary point of the Lagrangian is given in Equation (120).

$$\nabla \pounds = 0 \tag{120}$$

where ∇ denotes the gradient.

An iteration of Newton's method with simple bounds is given by Equation (121).

$$\vec{s}^{\,k+1} = \left[\vec{s}^{\,k} + \alpha^k \delta\vec{s}^{\,k}\right]_{\#} \tag{121}$$

where $\vec{s}^{\,k}$ is the vector of unknown parameters which include the states, co-states, and controls at the k^{th} iteration, α is a line search parameter, and $[\cdot]_{\#}$ denotes a projection onto simple bounds. The search direction $\delta\vec{s}^{\,k}$ can be found by minimizing the quadratic function given by Equation (122).

$$\text{minimize } \mathrm{F}_{\mathrm{qp}}(\delta \mathrm{s}) = \pounds(\vec{s}^{\,k}) + \nabla\pounds(\vec{s}^{\,k})^T(\delta\vec{s}) + \tfrac{1}{2}(\delta\vec{s})^T D^k(\delta\vec{s}) \tag{122}$$

$$D^k = \begin{bmatrix} [\nabla^2\pounds(\vec{s}^{\,k})]_A & 0 & \cdots & 0 \\ 0 & [\nabla^2\pounds(\vec{s}^{\,k})]_{i,i} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & [\nabla^2\pounds(\vec{s}^{\,k})]_{i,i} \end{bmatrix} \tag{123}$$

The subscript A denotes the set of parameters with inactive bound constraints, and the i indices span the parameters with active bound constraints. In words, matrix D^k is the Hessian of the Lagrangian modified to be diagonal for parameters with active bound constraints. This modification of the Hessian matrix is required to insure descent of the cost function for an iteration of the form in Equation (121), proof of this condition was introduced by Berteskas [11]. The set of inactive controls is defined by Equation (124).

$$I^{\#}(u) = \left\{ \begin{matrix} i/\ \ u_i \ge u_{i\max} - \varepsilon,\ \dfrac{\delta\pounds}{\delta u_i} < 0 \\ i/\ \ u_i \le \varepsilon - u_{i\min},\ \dfrac{\delta\pounds}{\delta u_i} > 0 \end{matrix} \right\} \tag{124}$$

Treatment of simple bounds in this way will be limited to the control variables, while bounds on the state will be addressed later.

For clarity, the structure of the gradient and Hessian of the Lagrangian function is given in Equations (126) and (127). The order of the vector of unknown parameters is given in Equation (125).

$$\vec{s}^{\,k} = \begin{bmatrix} \vec{u}_0^T & \vec{P}_1^T & \vec{x}_1^T & \vec{u}_1^T & \vec{P}_2^T & \cdots & \vec{x}_{L-1}^T & \vec{u}_{L-1}^T & \vec{P}_L^T & \vec{x}_L^T \end{bmatrix}^T \tag{125}$$

$$\nabla \mathcal{L} = \begin{Bmatrix} \vec{b}_0 + B_0^T \vec{P}_1 \\ \vec{G}_0 - \vec{x}_1 \\ \vec{a}_1 + A_1^T \vec{P}_2 - \vec{P}_1 \\ \vec{b}_1 + B_1^T \vec{P}_2 \\ \vec{G}_1 - \vec{x}_2 \\ \ddots \\ a_i + A_i^T \vec{P}_{i+1} - \vec{P}_i \\ b_i + B_i^T \vec{P}_{i+1} \\ \vec{G}_i - \vec{x}_{i+1} \\ \ddots \\ a_L - \vec{P}_L \end{Bmatrix} \tag{126}$$

$$\nabla^2 \mathcal{L} = \begin{bmatrix} R_0 & B_0^T & & & & & & & & & \\ B_0 & 0 & -I_x & & & & & & & & \\ & -I_x & Q_1 & M_1 & A_1^T & & & & & & \\ & & M_1^T & R_1 & B_1^T & & & & & & \\ & & A_1 & B_1 & 0 & -I_x & & & & & \\ & & & & -I_x & \ddots & \ddots & & & & \\ & & & & & \ddots & \ddots & -I_x & & & \\ & & & & & & -I_x & Q_{L-1} & M_{L-1} & A_{L-1}^T & \\ & & & & & & & M_{L-1}^T & R_{L-1} & B_{L-1}^T & \\ & & & & & & & A_{L-1} & B_{L-1} & 0 & -I_x \\ & & & & & & & & & -I_x & Q_L \end{bmatrix} \tag{127}$$

$$\begin{aligned} R_i &\equiv \nabla_{uu}^2 H_i & \quad M_i &\equiv \nabla_{xu}^2 H_i \\ Q_i &\equiv \nabla_{xx}^2 H_i & \quad Q_L &\equiv \nabla_{xx}^2 g_L \end{aligned} \tag{128}$$

$$\begin{aligned} A_i &\equiv \left(\nabla_u \vec{G}_i\right)^T \\ B_i &\equiv \left(\nabla_x \vec{G}_i\right)^T \\ a_i &\equiv \nabla_x g_i + (C^T DC)\vec{x}_i - (C^T D)\vec{u}_i \\ b_i &\equiv \nabla_u q_i + D\vec{u}_i - DC\vec{x}_i \end{aligned} \tag{129}$$

where matrices B_i and M_i are appropriately modified such that Equation (127) is diagonal for controls with active bound constraints. This amounts to setting to zero the columns of B_i, M_i, and off diagonal columns and rows of R_i for controls with active bound constraints.

The typical solution strategy involves solving for the state and co-state trajectories given a control history, this is done by selecting the states and co-states to satisfy the first order necessary conditions given by Equation (120). The state trajectory is found by forward propagation of Equation (114) with known initial conditions, and the co-state trajectory is found by backward propagation of Equation (130).

$$\begin{aligned} \vec{P}_L &= a_L \\ \vec{P}_i &= a_i + A_i^T \vec{P}_{i+1} \quad i = L-1 \ldots 1 \end{aligned} \tag{130}$$

Multiplying out Equation (122) and collecting terms results in the equivalent quadratic programming problem given by QP1.

Table 3-3: Quadratic Programming Problem 1 (QP1)

Minimize: $$\begin{aligned} \tilde{J}(\delta \mathrm{u}) = {} & \tfrac{1}{2}\delta\vec{x}_L^T Q_L \delta\vec{x}_L + a_L^T \delta\vec{x}_L + \\ & \sum_{i=0}^{L-1} b_i^T \delta\vec{u}_i + \tfrac{1}{2}\delta\vec{u}_i^T R_i \delta\vec{u}_i + \\ & \sum_{i=1}^{L-1} a_i^T \delta\vec{x}_L + \tfrac{1}{2}\delta\vec{x}_i^T Q_i \delta\vec{x}_i + \tfrac{1}{2}\delta\vec{u}_i^T M_i \delta\vec{x}_i \end{aligned} \tag{131}$$
Subject to: $$\begin{aligned} \delta\vec{x}_0 &= 0 \\ \delta\vec{x}_{i+1} &= A_i \delta\vec{x}_i + B_i \delta\vec{u}_i \quad i = 1 \ldots L-1 \end{aligned} \tag{132}$$

Problem QP1 is solved for $\delta\vec{u}$, which is used in Equation (121) to find the exact Newton iteration for the original nonlinear programming problem OP2. Efficient solution to a problem of the form QP1 can be found by a Riccatti recursion, a brief outline is given presently.

System QP1 is solved by finding a stationary point of the Lagrangian given in Equation (133) and (134).

$$\tilde{L}(\delta\vec{u}) = \tfrac{1}{2}\delta\vec{x}_L^T Q_L \delta\vec{x}_L + a_L^T \delta\vec{x}_L + \sum_{i=0}^{L-1} \tilde{H}_i - \lambda_{i+1}^T \vec{x}_{i+1} \qquad (133)$$

$$\begin{aligned} \tilde{H}_i &= b_i^T \delta\vec{u}_i + \tfrac{1}{2}\delta\vec{u}_i^T R_i \delta\vec{u}_i + a_i^T \delta\vec{x}_i + \tfrac{1}{2}\delta\vec{x}_i^T Q_i \delta\vec{x}_i + \delta\vec{u}_i^T M_i \delta\vec{x}_i + \ldots \\ &\quad \vec{\lambda}_{i+1}^T (A_i \delta\vec{x}_i + B_i \delta\vec{u}_i) \quad i = 1 \ldots L-1 \\ \tilde{H}_0 &= b_0^T \delta\vec{u}_0 + \tfrac{1}{2}\delta\vec{u}_0^T R_0 \delta\vec{u}_0 \end{aligned} \qquad (134)$$

where $\vec{\lambda}_i$ are the co-states of the quadratic program QP1. A stationary point of QP1 is found by satisfying the quadratic program state equations given in Equation (135), the co-state equations given in Equation (136), and the stationary conditions given in Equation (137).

$$\begin{aligned} \delta\vec{x}_{i+1} &= A_i \delta\vec{x}_i + B_i \delta\vec{u}_i \quad i = 0 \ldots L-1 \\ \delta x_0 &= 0 \end{aligned} \qquad (135)$$

$$\begin{aligned} \vec{\lambda}_L &= a_L + Q_L \delta\vec{x}_L \\ \vec{\lambda}_i &= a_i + Q_i \delta\vec{x}_i + M_i^T \delta\vec{u}_i + A_i^T \vec{\lambda}_{i+1} \quad i = 1 \ldots L-1 \end{aligned} \qquad (136)$$

$$\begin{aligned} & b_0 + R_0 \delta\vec{u}_0 = 0 \\ & b_i + R_i \delta\vec{u}_i + M_i \delta\vec{x}_i + B_i^T \vec{\lambda}_{i+1} = 0 \quad i = 1 \ldots L-1 \end{aligned} \qquad (137)$$

The set of Equations (135) through (137) represent the necessary conditions for a local minimum, and can be satisfied in closed form resulting in Equations (138) through (140).

$$\delta\vec{u}_i^k = -(R_i + B_i^T S_{i+1} B_i)^{-1}((M_i + B_i^T S_{i+1} A_i)\delta\vec{x}_i^k + b_i + B_i^T \vec{\lambda}_{i+1}) \quad i = 0 \ldots L-1 \qquad (138)$$

$$\begin{aligned} \vec{\lambda}_N &= a_N \\ \vec{\lambda}_i &= a_i + A_i^T S_{i+1} B_i (R_i + B_i^T S_{i+1} B_i)^{-1} (b_i + B_i^T \vec{\lambda}_{i+1}) \quad i = 1 \ldots L-1 \end{aligned} \qquad (139)$$

$$\begin{aligned} & S_L = Q_L \\ & S_i = A_i^T S_{i+1} A_i + Q_i - (B_i^T S_{i+1} A_i + M_i)^T (R_i + B_i^T S_{i+1} B_i)^{-1} (B_i^T S_{i+1} A_i + M_i) \\ & i = 1 \ldots L-1 \end{aligned} \qquad (140)$$

Descent of the original cost function is guaranteed where the matrix $(R_i + B_i^T S_{i+1} B_i)$ is positive definite. If this is not the case, the matrix should be modified by adding a diagonal positive definite matrix such that $(R_i + R_{iM} + B_i^T S_{i+1} B_i)$ is positive definite [11]. The algorithm is summarized in Table 3-4.

Table 3-4: SQP Optimal Control Algorithm

```
Given: $\vec{x}_0, \vec{u}$

For number of iterations
    Step 1: Forward propagate state equations, Equation(114)
    .

    Step 2: Back Propagate for $\vec{P}$ using Equation (130) ,
            $\vec{\lambda}$ using Equation (139),and
            $\vec{S}$ using Equation (140).

    Step 3: Forward Propagate Newton control step $\delta\vec{u}$ using
            Equations (138) and (132).

    Step 4: Update controls using Equation (121).
            update iteration index
End
```

The parameter α^k in Equation (121) is determined by line search defined in Equation (141).

$$\alpha^k = \beta^m \tag{141}$$

where $\beta < 1$, and $m = 0,1,\ldots$. The first m which results in a reduced cost is accepted. Typically a line search technique requires sufficient cost reduction for acceptance to insure convergence in a finite number of steps. In this case, any cost reduction is acceptable because full convergence is not expected but a small computation time per iteration is desired.

State Inequality Constraints

State inequality constraints (Equation (116)) are enforced using interior point boundary functions [11]. Boundary functions are appended to the cost function which penalizes the cost for infeasible states. Typically boundary functions with singularities at the limits are used and initial feasible trajectories are required, however all further solutions then remain feasible. In the problem considered in this work initial feasible trajectories are easily found, and feasible iterates are desirable. Typical form of a boundary function is shown in Figure 3-1.

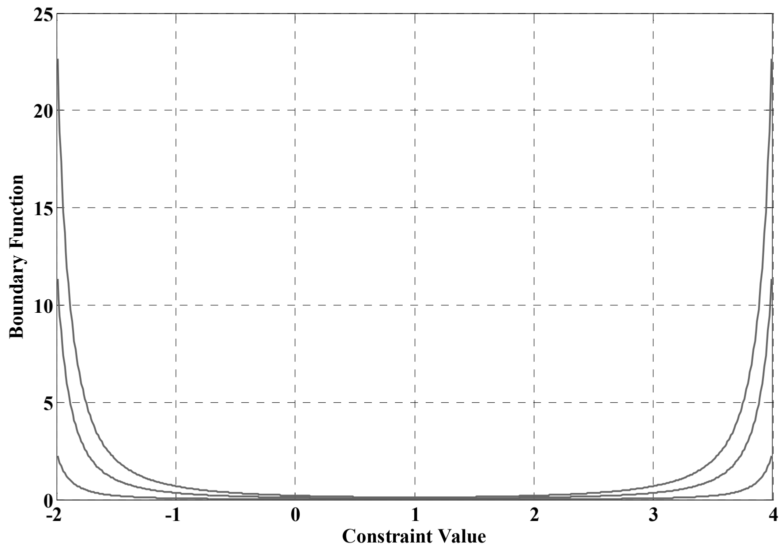

Figure 3-1: Boundary Function for a Range of C Values

where the boundary function is given by Equation (142).

$$F_B = c/(\chi - \chi_{\min}) + c/(\chi_{\max} - \chi) \tag{142}$$

Typically the scalar c is decreased as the optimal solution is approached, allowing the trajectory to approach the constraint more closely. In this work, a small c value is selected and increased if the bound is approached too closely. Care must also be taken that the bounds are not violated due to system unknowns, state noise, or when stepping the prediction horizon forward in time. If the bounds are violated at any point, they should be shifted and the parameter c increased until the original bounds are again satisfied.

Receding Horizon Implementation

The optimal control algorithm described in this work is applied in a receding horizon fashion. Two algorithm iterations are conducted for each control update requiring an average of 0.0125 seconds per iteration for a prediction horizon of 10 seconds. Time per iteration varies due to line searching, however a line search was not required in most cases. A fixed update rate of 0.05 seconds was selected for this work allowing a buffer of 0.025 seconds for line searching or other miscellaneous tasks.

After updating the control, the initial state vector required for the control algorithm is approximated by projecting the model forward in time using the current control input. The control trajectory from the previous iteration is used to start the next iteration as shown in Figure 3-2. Robustness of this algorithm to modeling uncertainties and disturbances is investigated in the results section.

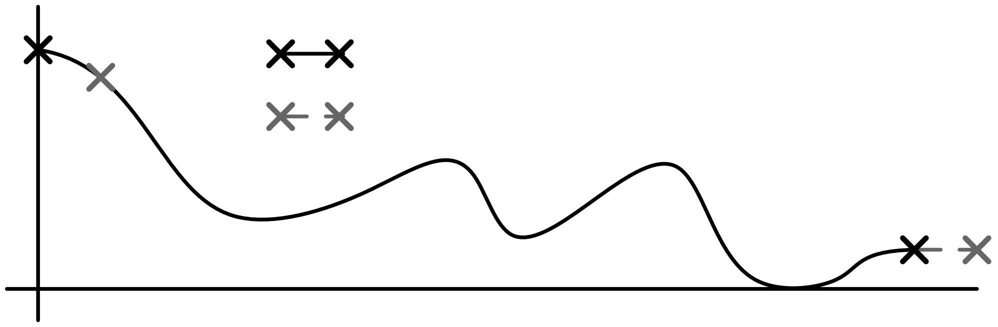

Figure 3-2: Control Trajectory Update

GLIDER MODEL

A point mass model is sufficient for this investigation. In the equations that follow, the ground frame is assumed to be a satisfactory reference frame. Glider inertial position is described in Equations (143) through (145).

$$\dot{x} = Vc_\theta c_\psi - W_x(z) \tag{143}$$

$$\dot{y} = Vc_\theta s_\psi - W_y(z) \tag{144}$$

$$\dot{z} = -Vs_\theta - W_z(x,y) \tag{145}$$

where, V is the glider airspeed, θ is the path pitch angle, and ψ is the path heading measured from the positive x axis. Wind is described in the inertial frame, W_x is wind velocity in the negative x direction, W_y is the wind velocity in the negative y direction, and W_z is the wind velocity in the negative z direction. The horizontal wind velocity varies with altitude, and vertical wind velocity varies with x, and y position. Wind models are discussed in the following section. The common shorthand for trigonometric functions is used throughout the paper ($\sin(\theta) \equiv s_\theta$). Dynamic equations of motion are described in Equations (146) through (148).

$$\dot{V} = -\tfrac{D}{m} - gs_\theta + c_\psi c_\theta \dot{W}_x + s_\psi c_\theta \dot{W}_y - s_\theta \dot{W}_z \tag{146}$$

$$\dot{\psi} V c_\theta = \tfrac{Ls_\phi}{m} - s_\psi \dot{W}_x + c_\psi \dot{W}_y \tag{147}$$

$$\dot{\theta} V = \tfrac{Lc_\phi}{m} - c_\theta g - s_\theta c_\psi \dot{W}_x - s_\theta s_\psi \dot{W}_y - c_\theta \dot{W}_z \tag{148}$$

where ϕ is the glider bank angle, and m is the mass of the glider. Lift and drag forces are given by Equations (149) through (151).

$$L = \tfrac{1}{2}\rho V^2 S C_L \tag{149}$$

$$D = \tfrac{1}{2}\rho V^2 S C_D \tag{150}$$

$$C_D = C_{D_0} + kC_L^2 \tag{151}$$

where C_L is the lift coefficient, C_{D_0} is the profile drag coefficient, k is the induced drag coefficient, ρ is the air density, and S is the wing area.

The system model is normalized to enhance the numerical properties of the optimization algorithm. Non-dimensional parameters are defined in Equations (152) through (154).

$$\bar{x} = \frac{x}{t_{ph}V^*} \quad \bar{y} = \frac{y}{t_{ph}V^*} \quad \bar{z} = \frac{z}{t_{ph}V^*} \tag{152}$$

$$\bar{V} = \frac{V}{V^*} \tag{153}$$

$$t = \tau t_{ph} \tag{154}$$

where t_{ph} is the prediction horizon, V^* is a nominal airspeed, and τ is the normalized time. The resulting non-dimensional equations of motion are given in Equations (155) through (162).

$$\bar{x}' = \bar{V}c_\theta c_\psi - \bar{W}_x(\bar{z}) \tag{155}$$

$$\bar{y}' = \bar{V}c_\theta s_\psi - \bar{W}_y(\bar{z}) \tag{156}$$

$$\bar{z}' = -\bar{V}s_\theta - \bar{W}_z(\bar{x}, \bar{y}) \tag{157}$$

$$\bar{V}' = -\bar{g}s_\theta - D_{p_c}C_d\bar{V}^2 + c_\psi c_\theta \bar{W}_x' + s_\psi c_\theta \bar{W}_y' - s_\theta \bar{W}_z' \tag{158}$$

$$\psi' = D_{p_c}\bar{V}C_L\left(\frac{s_\phi}{c_\theta}\right) - \left(\frac{s_\psi}{c_\theta \bar{V}}\right)\bar{W}' + \left(\frac{c_\psi}{c_\theta \bar{V}}\right)\bar{W}_y' \tag{159}$$

$$\theta' = D_{p_c}\bar{V}C_L c_\phi - \frac{\bar{g}c_\theta}{\bar{V}} - \left(\frac{s_\theta c_\psi}{\bar{V}}\right)\bar{W}_x' - \left(\frac{s_\theta s_\psi}{\bar{V}}\right)\bar{W}_y' - \left(\frac{c_\theta}{\bar{V}}\right)\bar{W}_z' \tag{160}$$

$$\bar{g} = gt_{ph}\big/V^* \tag{161}$$

$$D_{p_c} = (1\!/\!_{2m})\rho S V^* t_{ph} \tag{162}$$

where $(\cdot)'$ denotes differentiation with the normalized time variable τ. The normalized air-relative specific energy is given in Equation (163).

$$\bar{E}_{sp} = -\bar{z}\bar{g} + \tfrac{1}{2}\bar{V}^2 \tag{163}$$

The energy rate is given in Equation (164).

$$\begin{aligned} \bar{E}_{sp}' = {} & \bar{W}_z\bar{g} - D_{p_c}C_D\bar{V}^3 \ldots \\ & - s_\theta c_\theta \bar{V}^2(c_\psi \partial_z \bar{W}_x + s_\psi \partial_z \bar{W}_y + c_\psi \partial_x \bar{W}_z + s_\psi \partial_y \bar{W}_z)\ldots \\ & + s_\theta \bar{V}(\bar{W}_x \partial_x \bar{W}_z + \bar{W}_y \partial_y \bar{W}_z) - c_\theta \bar{V}\bar{W}_z(s_\psi \partial_z \bar{W}_y + c_\psi \partial_z \bar{W}_x) \end{aligned} \tag{164}$$

where $\partial_{\bar{z}}\bar{W}_x$ is the partial derivative of $\bar{W}_x$ with respect to $\bar{z}$. It is interesting to note that the thermal wind velocity directly increases the energy rate, and the drag force decreases the energy rate proportional to the cube of airspeed. Other energy transfer mechanisms include individual gradients of wind velocity, and coupled terms relying on both horizontal and vertical winds and their gradients.

Glider model parameters are selected to represent the performance of a typical micro air robot with a small payload.

Table 3-5: Glider Model Parameters

Mass (kg)	1.55
Aerodynamic Area (m^2)	0.3406
Max Lift to Drag	38
Profile Drag Coefficient	0.010

WIND STRUCTURE MODELS

The basic scenarios investigated in this work involve thermals and wind gradients. Optimal trajectories in wind gradients alone are classified as dynamic soaring, and have been investigated by many authors including Sachs, Lissaman, Zhao, and Goto [3-7]. Static and semi-dynamic soaring in stationary thermals has been studied by Wharington [1]. Wharington found that many local minima exist in static soaring problem, and shows that, under certain conditions, maneuvers classified as semi-dynamic result in larger altitude gain per cycle than the typical static trajectories. Wharington did not investigate the effects of wind shear.

A simple thermal model is sufficient for this investigation [18], given by Equations (165) and (166).

$$r_g^2 = \left(e_T(\bar{x} - \bar{x}_T(\bar{z}))\right)^2 + \left((\bar{y} - \bar{y}_T(\bar{z}))/e_T\right)^2 \tag{165}$$

$$\bar{W}_z = \bar{w}_T \cos(\pi r_g^2 / 2r_T^2)\,\mathrm{e}^{(r_g^2 \bar{f})} - \bar{w}_d \tag{166}$$

where $\bar{w}_T$ is the thermal wind velocity at the core, r_T^2 is the thermal radius, $\bar{x}_T$ and $\bar{y}_T$ define the location of the thermal, $\bar{w}_d$ is the surrounding downdraft, $\bar{f}$ is the thermal decay parameter, and e_T allows elliptic thermals to be modeled. An example thermal profile is shown Figure 3-3.

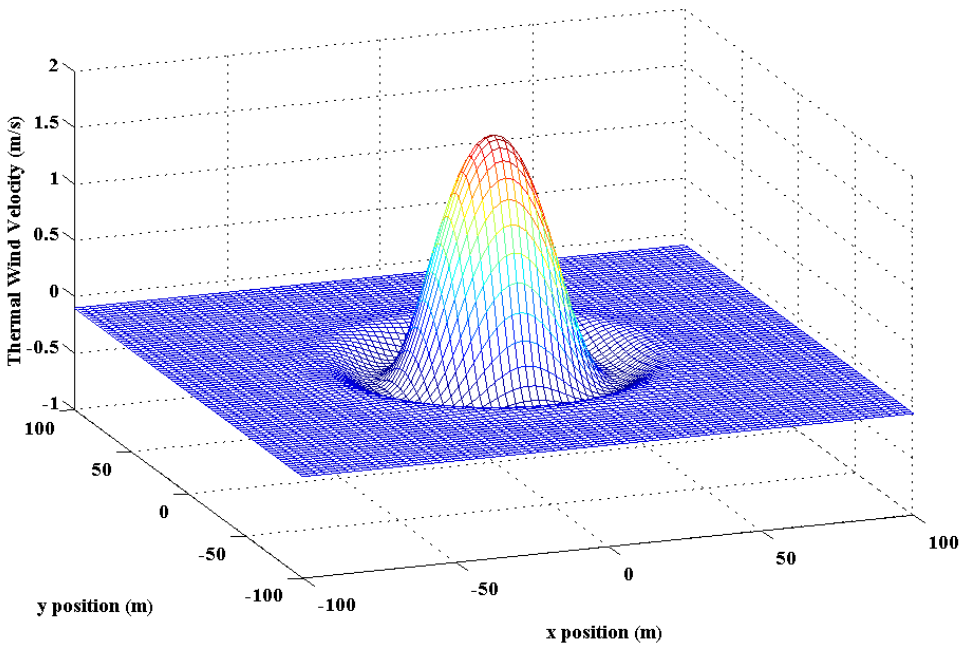

Figure 3-3: Thermal Model Profile

A wind gradient model developed by Zhao [5] is used in this study. The general form models linear, exponential and logarithmic gradient profiles. Horizontal wind is given as a function of altitude by Equation (167).

$$w_{hg} = \beta\left[a(h - h_{\min}) + \frac{1-a}{h_{\max} - h_{\min}}(h - h_{\min})^2 + w_{h\min} \right] \tag{167}$$

where β is the average slope, $h_{\min}$ is the height the gradient begins, $h_{\max}$ is the height the gradient ends, $w_{h\min}$ is the minimum horizontal wind velocity, and a is the shape parameter in the range 0 to 2. The gradient profile is shown for different values of a in Figure 3-4.

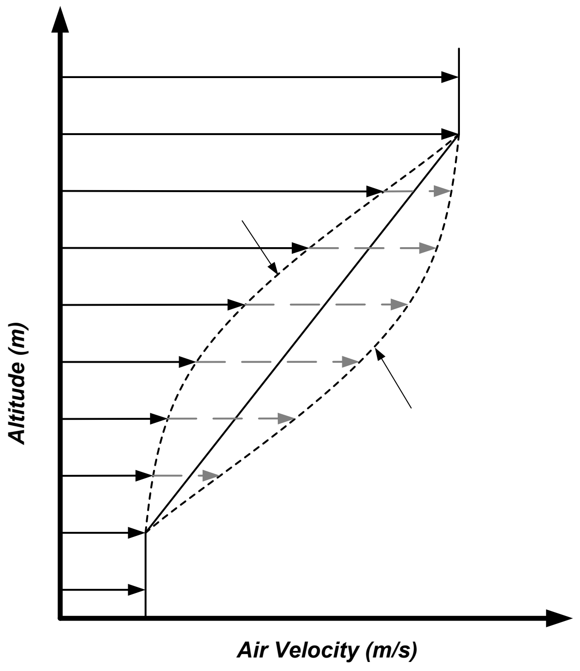

Figure 3-4: Wind Gradient Model Profile

EXAMPLE RESULTS

The first scenario considered involves a single thermal located at the origin, with wind model parameters given in Table 3-6. A generic cost function is used which balances the potential and kinetic energy gains.

$$g_i = E_p(\overline{z}_i - \overline{z}_0) + \tfrac{E_k}{2}(\overline{V}_0^2 - \overline{V}_i^2) \tag{168}$$

where $\overline{z}_o$ and $\overline{V}_0$ are the initial position and airspeed, and the scalar constants E_p, and E_k specify the relative importance of potential and kinetic energy gains. Also, the distance from the thermal core is lightly penalized to increase the robustness of the controller. The bank angle trajectory initialized to zero, and the lift coefficient is initialized to $C_{L\max}$.

The glider begins outside of the thermal heading directly to the thermal core. The resulting trajectory is shown in Figures 3-5 through 3-10.

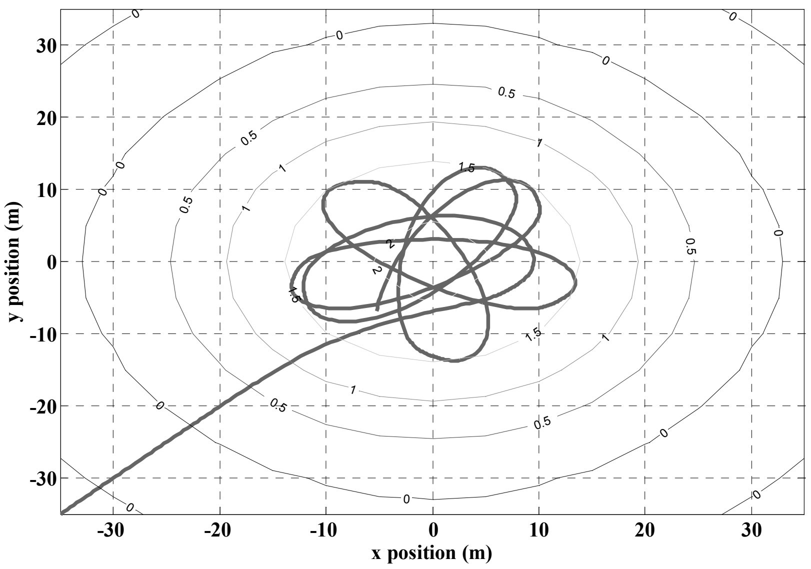

Figure 3-5: Single Thermal Scenario Trajectory

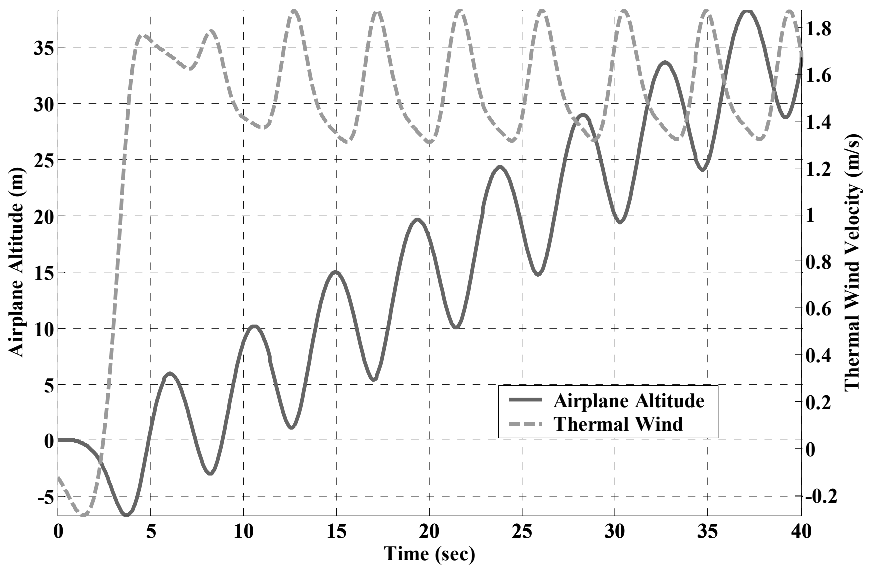

Figure 3-6: Single Thermal Scenario Altitude and Thermal Velocity Time History

A repeatable cycle with an average altitude gain of 1 m/s is achieved very quickly after the glider enters the thermal. The trajectory is perfectly centered about the thermal core.

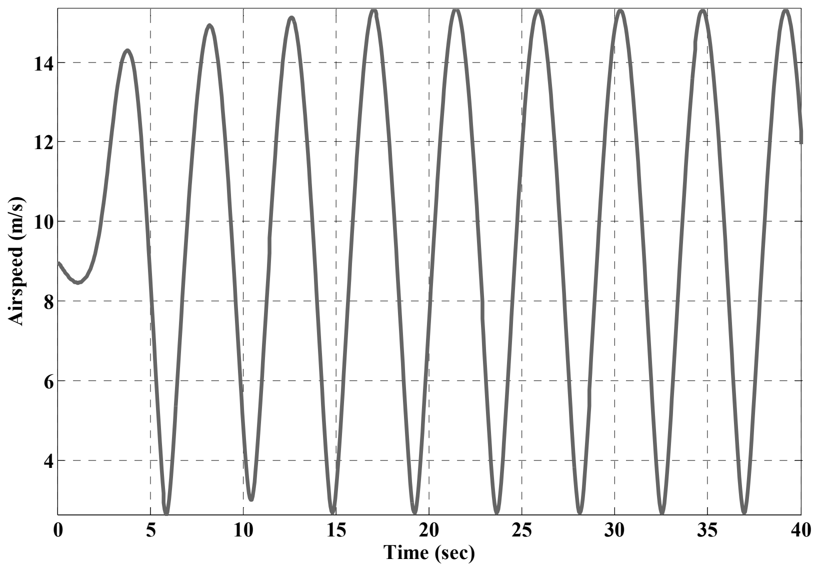

Figure 3-7: Single Thermal Scenario Airspeed Time History

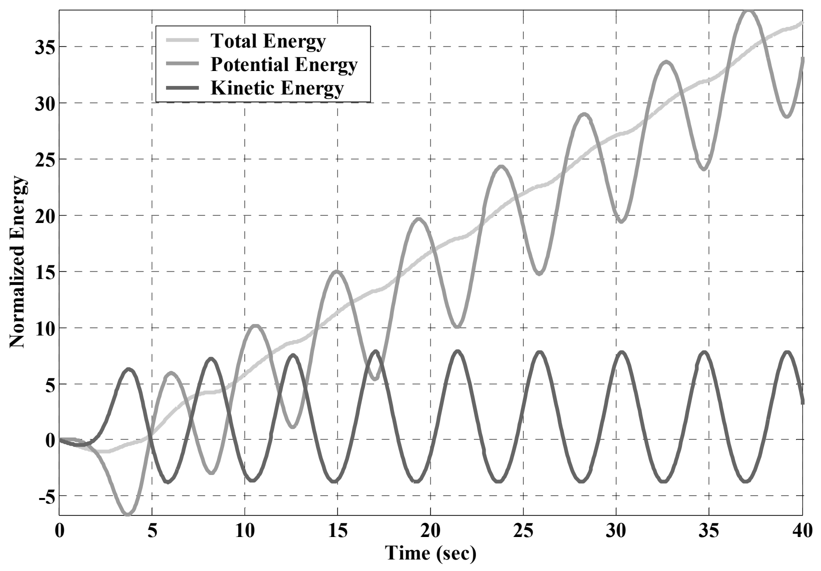

Figure 3-8: Single Thermal Scenario Energy Gain Time History

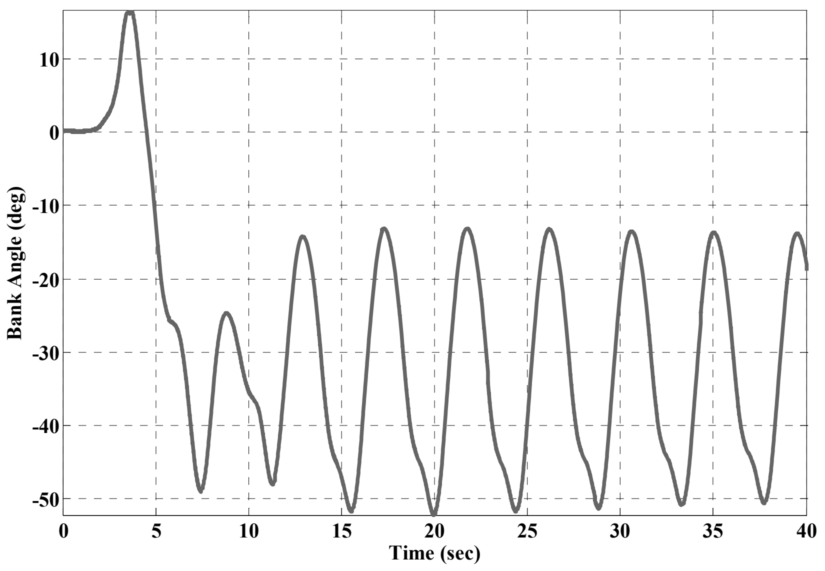

Figure 3-9: Single Thermal Scenario Bank Angle Time History

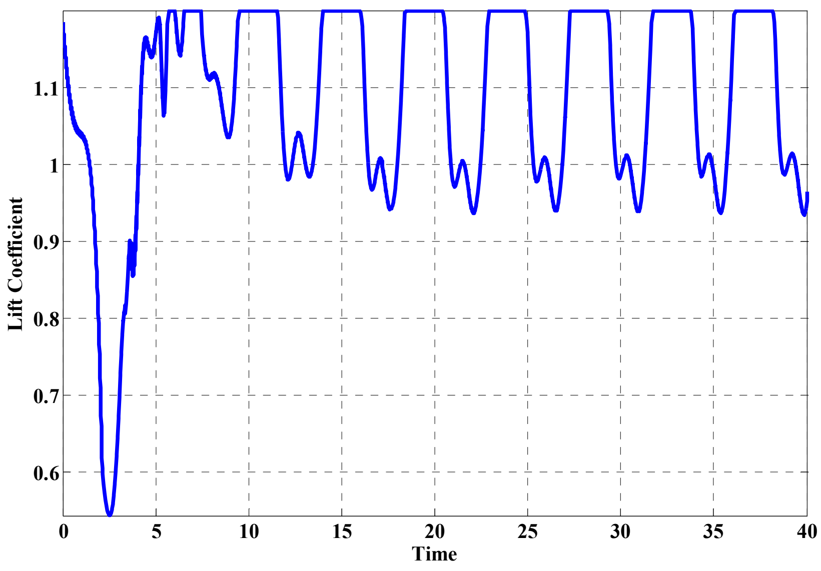

Figure 3-10: Single Thermal Scenario Lift Coefficient Time History

Airspeed in this case varies between 3 m/s and 15 m/s, very near the lower bound of 2.5 m/s. The maximum bank rate required in this example is 38 deg/s.

The second scenario investigated is a single oval shaped thermal with parameters given in Table 3-7. A semi-dynamic trajectory is executed similar to the circular thermal, however the trajectory aligns itself with the thin axis of the thermal in this case. Altitude gain for the oval thermal is nearly identical to the altitude gain in the circular thermal.

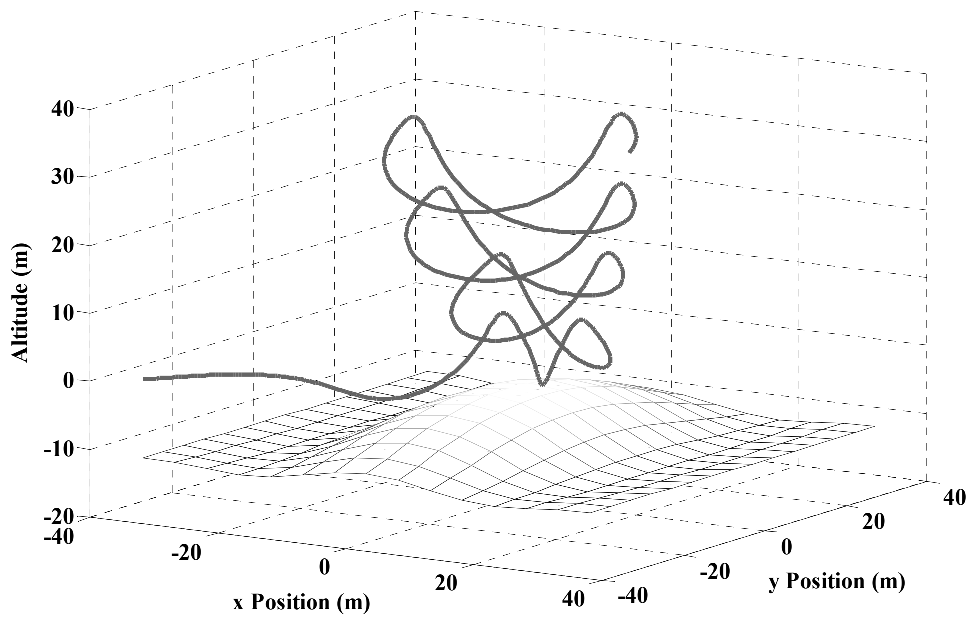

Figure 3-11: Oval Thermal Scenario Trajectory

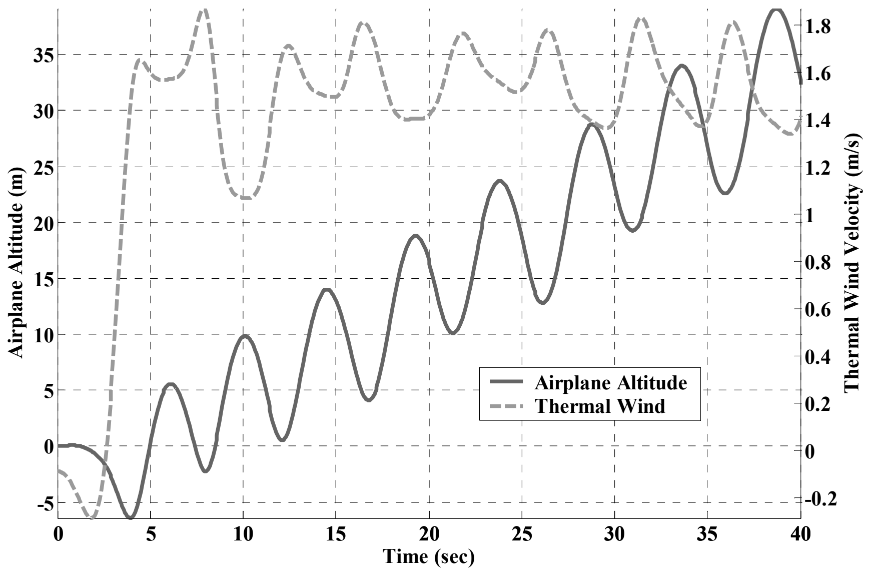

Figure 3-12: Oval Thermal Scenario Altitude and Thermal Velocity Time History

Finally a combination scenario with a single circular thermal and a horizontal wind gradient is investigated. Wind structure parameters are given in Table 3-6 and 3-8.

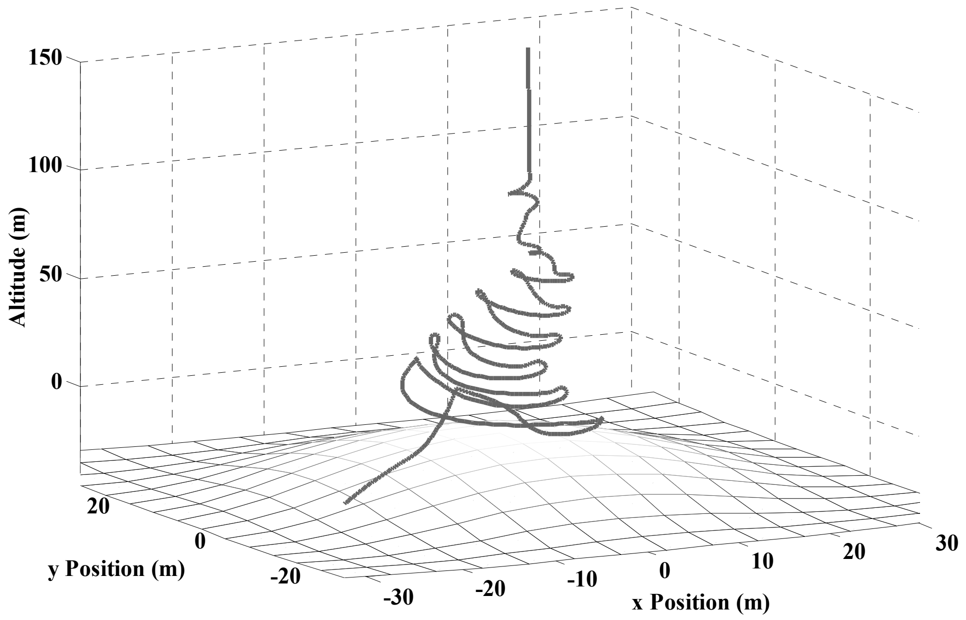

Figure 3-13: Thermal and Wind Gradient Trajectory

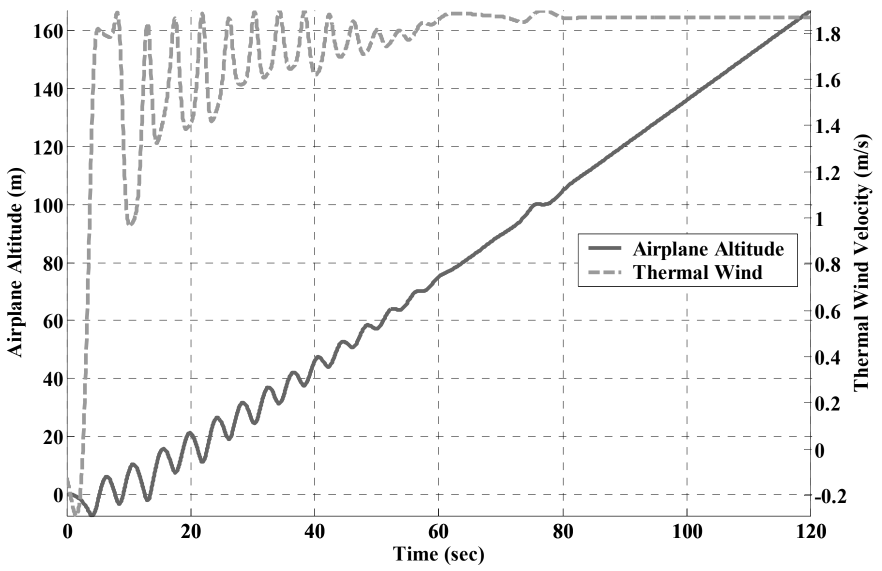

Figure 3-14: Thermal and Wind Gradient Scenario Altitude and Thermal Wind Velocity Time History

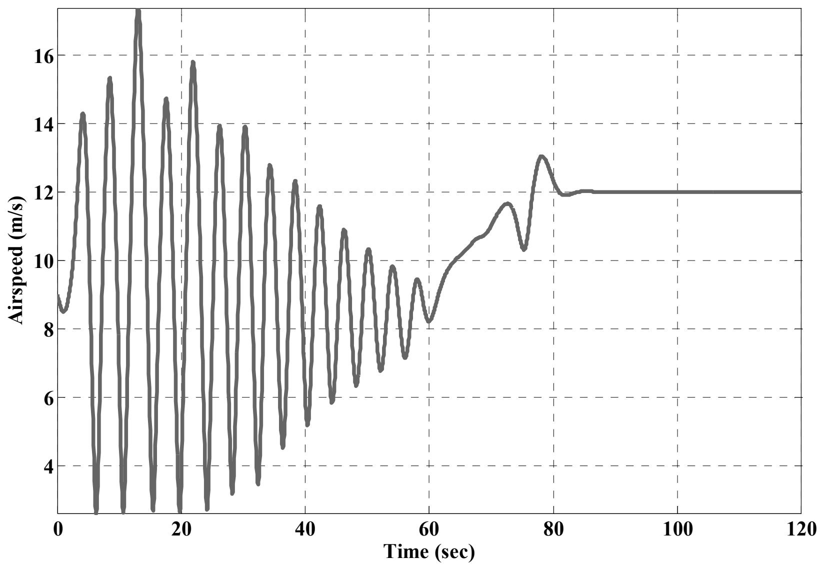

Figure 3-15: Thermal and Wind Gradient Scenario Airspeed Time History

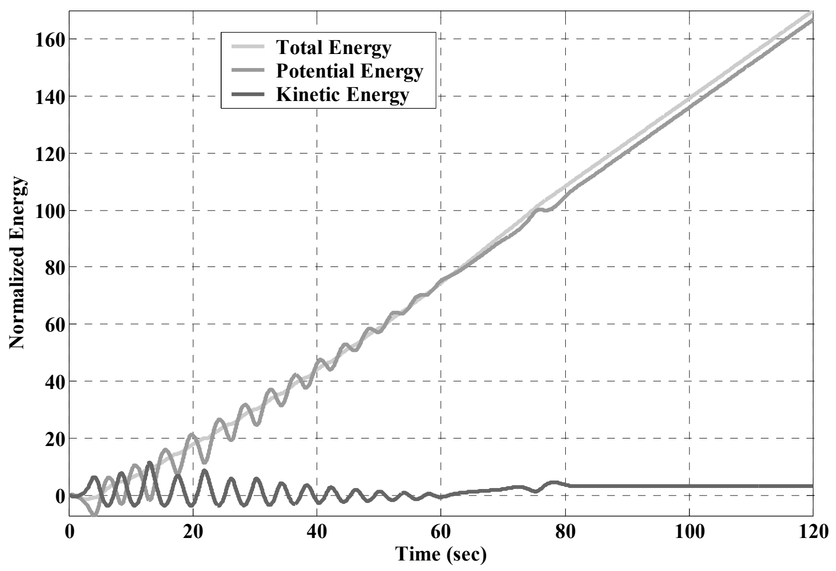

Figure 3-16: Thermal and Wind Gradient Scenario Energy Gain Time History

Initially a semi-dynamic trajectory is achieved at the center of the thermal, where the glider banks back and forth facing into the horizontal wind. The bank trajectory in this case is slowly damped out until the glider is left hovering at the core of the thermal gaining nearly 2 meters of altitude each second.

Table 3-6: Round Thermal Model Parameters

Thermal Radius (m)	30.0
Core Velocity (m/s)	2.0
Downdraft (m/s)	-0.1
Elliptic parameter e	1.0
Decay Rate $\overline{f}$	-0.0015

Table 3-7: Oval Thermal Model Parameters

Thermal Radius (m)	30.0
Core Velocity (m/s)	2.0
Downdraft (m/s)	-0.1
Elliptic parameter e	0.7
Decay Rate $\overline{f}$	-0.0015

Table 3-8: Horizontal Wind Gradient Model Parameters

Minimum Height (m)	-1.0
Maximum Height (m)	100.0
Maximum Wind Velocity (m/s)	12.0
Minimum Wind Velocity (m/s)	0.0
Shape Parameter A	1.0

The control algorithm runs significantly faster than real time in all scenarios investigated. A prediction horizon of 10.0 seconds was used, although any prediction horizon longer than 9 seconds was also found to work well. Two algorithm iterations were conducted for each control update requiring an average of 0.0125 seconds per iteration for a real time update rate of 0.05 seconds leaving a buffer of 0.025 seconds per iteration for other necessary tasks. It was observed that the Newton step was used in nearly every case, however the line search strategy described previously is easily achieved within the buffer time. All computation time measurements were made with the CPU time functionality in Fortran 95, on a laptop running an Intel Centrino Duo T2300 CPU.

SENSITIVITY INVESTIGATION

Performance of the receding horizon algorithm is excellent where the environment is known perfectly; unfortunately this is never the case especially where wind is concerned. Performance in scenario 3 involving a single thermal and a wind gradient is investigated where various parameters are incorrectly known. The controller is executed with parameters described in the previous section for scenario three. The real glider path is determined with model parameters representing the "real world" scenario. Parameters are given in Tables 3-9 through 3-12, the "real world" parameters are crowned with a ~. It was necessary to increase the lower bound on airspeed from 2.5 m/s to 4.0 m/s to insure the lower bound on airspeed is enforced strictly. A one-minute real time simulation is compared, and performance is measured in total altitude gain. Altitude gain in the "perfect world" scenario is 73.9952 meters in 60 seconds.

Several significant parameters in the thermal model are investigated and results are given in Table 3-9.

Table 3-9: Thermal Parameter Sensitivity Investigation

[Nominal Performance 73.9952 m]

Thermal Location [Performance / Position Error]	**Thermal Magnitude [Performance / $\frac{\tilde{w}_T}{\bar{w}_T}$]**	**Thermal Radius [Performance / $\frac{\tilde{r}_T}{r_T}$]**
73.8573 m / 0.3 m	60.1629 m / 0.9	73.0033 m / 0.9
72.9379 m / 1.5 m	38.3686 m / 0.75	70.1632 m / 0.75
52.7159 m / 7.5 m	6.5348 m / 0.5	46.2809 m / 0.5
18.6594 m / 15.0 m	86.3094 m / 1.1	74.6947 m / 1.1
-34.9270 m / 30.0 m	104.0877 m / 1.25	75.3775 m / 1.25

Error in the location of the thermal has a negligible effect on performance until it approaches 25% of the thermal radius. Performance varies with the thermal magnitude error as expected. The thermal radius is relatively insensitive to errors, as expected due to the tight pattern flown around the thermal core.

Several significant parameters in the wind gradient model are investigated and results are given in Table 3-10.

Table 3-10: Wind Gradient Parameter Sensitivity Investigation

[Nominal Performance 73.9952 m]

Gradient Profile Parameter a [Performance/ $\frac{\tilde{a}}{a}$]	**Minimum Velocity [Performance / $\tilde{V}_{\min} - V_{\min}$]**	**Maximum Velocity [Performance / $\tilde{V}_{\max} - V_{\max}$]**
58.6699 m / 0.0 m	42.0162 m / -10 m/s	56.2243 m / -10
65.0532 m / 0.5 m	69.1021 m / -5 m/s	57.2663 m / -5
77.0420 m / 1.5 m	58.6210 m / 5 m/s	74.4580 m / 5
73.3062 m / 2.0 m	-8.4857 m / 10 m/s	61.0661 m / 10

Significant performance loss is found where the minimum velocity error is above 5 m/s, otherwise errors in all of the wind gradient parameters have a minimal effect on performance of the controller.

All performance parameters in the glider model are investigated and results are given in Table 3-11 and 3-12.

Table 3-11: Glider Aerodynamic Parameter Sensitivity Investigation

[Nominal Performance 73.9952 m]

Profile Drag Coefficient [Performance/ $\frac{\overline{C}_{do}}{C_{do}}$]	**Induced Drag Coefficient [Performance/ $\frac{\overline{k}}{k}$]**
46.6307 m / 5.0	36.0991 m / 2.0
68.6396 m / 2.0	52.8589 m / 1.5
76.1627 m / 0.5	89.3808 m / 0.5
77.8955 m / 0.1	96.8201 m / 0.25

Glider model aerodynamic parameters have a predictable effect on performance of the controller.

Table 3-12: Glider Parameter Sensitivity Investigation

[Nominal Performance 73.9952 m]

Normalized $\bar{g}$ [Performance / $\frac{\tilde{g}}{\bar{g}}$]	**Normalized Dpc** [Performance / $\frac{D_{pc}}{\tilde{D}_{pc}}$]
58.3549 m / 1.25	81.0874 m / 1.5
67.2711 m / 1.125	79.1965 m / 1.25
81.1604 m / 0.75	60.2248 m / 0.8125
78.4131 m / 0.5	57.9424 m / 0.75

Glider model physical parameters have a predictable effect on performance of the controller.

CONCLUSIONS

An efficient optimal model predictive algorithm has been presented and extended to handle simple bounds on the control variables and nonlinear inequality constraints on the state variables. The controller was exercised in a receding horizon formulation on the problem of autonomous soaring and performs faster than real time for this problem.

Significant altitude gain was achieved for several scenarios involving relatively small thermals and a stiff horizontal wind gradient. Performance loss when parameters of the wind structures or glider model are in error was investigated. The controller is shown to perform well when knowledge of the wind structures and glider model are reasonable.

REFERENCES

[1] Wharington J., and Herszberg I., "Optimal Semi-Dynamic Soaring," Royal Melbourne Institute of Technology, Melbourne, Australia, October 11, 1998.

[2] Allen M., "Autonomous Soaring for Improved Endurance of a Small Uninhabited Air Vehicle," 43rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 10-13, 2005.

[3] Sachs G., and Costa O., "Optimization of Dynamic Soaring at Ridges," AIAA Atmospheric Flight Mechanics Conference and Exhibit, Austin, Texas, 2003.

[4] Lissaman P., "Wind Energy Extraction by Birds and Flight Vehicles," AIAA 43rd Aerospace Sciences Meeting and Exhibit, Reno NV., 2005.

[5] Zhao Y., "Optimal Patterns of Glider Dynamic Soaring," Optimal Control Applications and Methods, Vol. 25, 2004, pp 67-89.

[6] Zhao Y., and Qi Y., "Minimum Fuel Powered Dynamic Soaring of Unmanned Aerial Vehicles Utilizing Wind Gradients," Optimal Control Applications and Methods, Vol. 25, 2004, pp 211-233.

[7] Goto N., and Kawable H., "Direct Optimization Methods Applied to a Nonlinear Optimal Control Problem," Mathematics and Computers in Simulation, 2000.

[8] Betts, J., "Survey of Numerical Methods for Trajectory Optimization," Journal of Guidance, Control, and Dynamics, Vol. 21, No. 2, 1998.

[9] Warner M., and Hodges D., "Solving Optimal Control Problems Using hp-Version Finite Elements in Time," Journal of Guidance, Control, and Dynamics, Vol. 23., No. 1, 2000.

[10] Hull, D., "Conversion of Optimal Control Problems into Parameter Optimization Problems," Journal of Guidance, Control, and Dynamics, Vol. 20, No. 1, 1997.

[11] Bertsekas D., "Nonlinear Programming," Athena Scientific, Belmont, Mass., 2nd edition, 1999.

[12] Hargraves C., and Paris S., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," Journal of Guidance, Control, and Dynamics, Vol. 10, No. 4, 1987.

[13] Conway B., and Larson K., "Collocation Versus Differential Inclusion in Direct Optimization," Journal of Guidance, Control, and Dynamics, Vol. 21., No. 5, 1998.

[14] Petit N., and Milam M., and Murray R., "Inversion Based Constrained Trajectory Optimization," NOLCOS, St. Petersburg Russia, 2001, pp 189-195.

[15] Schwartz A., "Theory and Implementation of Numerical Methods Based on Runge-Kutta Integration for Solving Optimal Control Problems," Ph.D Thesis, University of California at Berkeley, 1996.

[16] Slegers, N., Kyle, J., and Costello, M., "Nonlinear Model Predictive Control Technique for Unmanned Air Vehicles," Journal of Guidance, Control, and Dynamics, Vol. 29, No. 5, 2006.

[17] Yakimenko O., "Direct Method for Rapid Prototyping of Near-Optimal Aircraft Trajectories," Journal of Guidance, Control, and Dynamics, Vol. 23, No. 5, 2000.

[18] Kyle J., Evans K., and Costello M., "Atmospheric Wind Energy Extraction by a Micro Autonomous Glider," AIAA Atmospheric Flight Mechanics Conference and Exhibit, San Francisco, CA, Aug. 15-18, 2005.

4 GENERAL CONCLUSIONS

A multiple input multiple output model predictive controller was developed with the purpose of enabling autonomous aircraft energy extraction from atmospheric winds through static soaring. The flight control law requires that external information is provided on the local wind structure and full state feedback has been assumed. Numerical results exercising this control law on a 6DOF model show an aircraft altitude gain in two thermals of known location and size. Climb rates of ½ m/s and ¼ m/s were achieved in the subsequent thermals. It was found that for smaller thermals more aggressive bank angles still resulted in impressive climb rates.

A hybrid solver was developed and shown to be robust to initial guess, while performing only slightly slower than direct shooting alone. Optimal trajectories for typical soaring scenarios were investigated. The importance of the robust solver is highlighted by investigating a small modification to a typical scenario, which results in convergence failure for the gradient method alone.

An efficient optimal model predictive control algorithm has been presented and extended to handle simple bounds on the control variables and nonlinear inequality constraints on the state variables. The controller is exercised in a receding horizon formulation on the problem of autonomous soaring and performs faster than real time for this problem.

Significant altitude gain was achieved through "Semi-Dynamic Soaring" trajectories for several scenarios involving small thermals and a stiff horizontal wind gradient. This new approach to the autonomous soaring problem performs exceptionally well in the scenarios investigated and offers several advantages including a fast update rate, and no requirements of prior knowledge of optimal trajectories in similar scenarios.

Performance loss when parameters of the wind structures or glide model are in error was investigated. The controller is shown to perform well when knowledge of the wind structures and glider model are reasonable.