Energy-Optimized Trajectory Planning for High Altitude Long Endurance (HALE) Aircraft

Hamidreza Bolandhemmat, Benjamin Thomsen, and Jack Marriott

Abstract—This paper outlines the energy-optimized trajectory planning problem for high altitude, long endurance (HALE) aircraft and explores both offline and online optimization techniques to address it. The goal is to find the optimal state and input trajectories for the solar-powered airplane, with input and nonlinear state constraints, which maximize net battery and gravitational potential energy storage. Solutions to the energy-optimal trajectory planning problem, using a six degree-of-freedom model of the nonlinear HALE aircraft dynamics, are computed using both an interior point optimization technique and a bounded nonlinear simplex search algorithm. The optimal trajectories, computed offline, are utilized to train an adaptive neuro-fuzzy inference system (ANFIS) which can be implemented on the flight control computer for online, in-flight trajectory planning. Simulation results show up to 15% more energy storage compared to a baseline parametrically-optimized trajectory.

I. INTRODUCTION

Solar-powered high altitude, long endurance (HALE) aircraft aim to remain aloft for long periods of time, and have been investigated for diverse applications including aerial surveillance and rural connectivity. To remain airborne for long durations, the aircraft must be able to remain above a prescribed altitude without fully discharging its batteries overnight. This constraint may be satisfied through a combination of design optimization, seasonal or geographic operational limitations, and trajectory optimization.

The problem considered in this article is the determination of dynamically-feasible state and input trajectories which maximize a solar-powered aircraft’s net energy storage (in both batteries and gravitational potential) over a period in time. We consider the case where the aircraft is constrained within a convex region in three-dimensional space, as well as input (bank angle, airspeed, altitude rate) constraints, and subject to winds in a horizontal plane.

Energy-optimal path planning for solar-powered aerial vehicles has been investigated in the literature in recent years [1], [2]. In comparison to these articles, however, we focus on long-endurance operation and consider winds, changing solar position, energy storage using gravitational potential, and real-time computation.

In this article, we will discuss and present the results from three methods of trajectory optimization. In Section II, we introduce the six degree-of-freedom rigid-body aircraft dynamics and control model which determines aircraft dynamic behavior in the optimization routine. In Section III, we formulate a nonlinear trajectory optimization problem and solve it via single shooting using both an interior-point optimization routine and the nonlinear Nelder-Mead simplex method. In Section IV, we present a learned model (an adaptive neuro-fuzzy inference system) trained using results of the aforementioned optimization techniques to provide a method capable of online trajectory planning.

II. AIRCRAFT DYNAMIC GUIDANCE MODEL

Trajectory optimization is carried out on a simulation model of a HALE aircraft with numerical integration of the dynamics. This major components of this simulation model are a nonlinear rigid-body aircraft dynamics model, guidance/control model, wind, Earth, and atmospheric model, solar model, and power model, which are described presently.

A. Dynamics and Guidance

The nonlinear aircraft dynamics model \( \dot{x} = f(x,u) \) has state vector

\[
x = [p_n, p_e, h, u, v, w, \varphi, \theta, \psi, p, q, r]^T
\]

(1)
corresponding to north and east positions, altitude, linear velocities in a body-fixed frame, roll, pitch, heading angles, and angular rates. For brevity, the full nonlinear equations of motion are not reproduced here, but are derived in [3, Ch. 4–5] and [4] and the mathematical descriptions can be found therein. The forces and moments acting on the aircraft are assumed to be defined fully by gravity, aerodynamics (including actuator state), and propulsion.

The aircraft which we model is assumed to have four control inputs given by \([T, \delta_a, \delta_e, \delta_r]\), corresponding to thrust, aileron deflection, elevator deflection, and rudder deflection. The relationships between these control inputs and the forces and moments acting on the 12-state rigid body model of the aircraft can be found in [3, Ch. 4]. As the primary concern of this article is trajectory optimization, and not inner-loop control design, we define the dynamic response to outer-loop commands using a dynamic guidance

\footnote{The authors are with Facebook Connectivity, Menlo Park, CA, USA. \{nbolandhemmat, benthomsen, marriott\}@fb.com}
model [3, Ch. 9]. The vehicle behavior is thus controlled by three independent commands, given by

\[ u^c = [V_{tas}^c, \dot{h}^c, \varphi^c] \quad (2) \]
corresponding to equivalent airspeed, altitude rate, and bank angle. These commands are fed into shaping filters, giving

\[ u = [V_{tas}^f, \dot{h}^c, \varphi^c] \quad (3) \]
which in turn are the inputs to representative control response models which determine \( V_{tas}, \dot{h}, \) and \( \varphi \) in the nonlinear aircraft dynamics model. Note that in reality these dynamic relationships are governed by control design, actuator dynamics, and any external effects, but we approximate this behavior using linear dynamic models.

B. Wind, Earth, and Atmosphere

The triangle model described in [3] is used as follows:

\[ V_g \begin{bmatrix} \cos \chi \cos \gamma \\ \sin \chi \cos \gamma \\ -\sin \gamma \end{bmatrix} = V_a \begin{bmatrix} \cos \psi \cos \gamma_a \\ \sin \psi \cos \gamma_a \\ -\sin \gamma_a \end{bmatrix} + \begin{bmatrix} w_n \\ w_e \\ w_d \end{bmatrix} \quad (4) \]

where \( V_g \) and \( V_a \) are ground speed and true airspeed, \( \chi \) is the course angle, \( \psi \) is the heading angle, \( \gamma \) and \( \gamma_a \) are the inertial and air-mass-reference flight path angles, respectively. Wind speed is given by \( V_a = \sqrt{w_n^2 + w_e^2 + w_d^2} \) where \( w_n, w_e, \) and \( w_d \) are the wind components in the north-east-down (NED) frame. Latitude and longitude \((\xi, \lambda)\) are computed from \((p_n, p_e, h)\) using a WGS84 Earth model.

Air density, \( \rho \), is computed according to the 1976 US Standard Atmospheric Model [5], and assuming equivalence between height above sea level and aircraft altitude, \( h \). Relationships between the vehicle’s angle of attack \( (\alpha) \), altitude \( (h) \) and aerodynamic coefficients are pre-computed using computational fluid dynamics software. A description of the full aeroservoelastic model from which this data is derived can be found in [6]. The lift force, \( L \), acting on the vehicle in translational flight is computed from the relationship

\[ \frac{L}{\cos \varphi} = \frac{mg}{\cos \gamma} = \frac{m V_a^2}{r_c + h} \quad (5) \]

where \( mg \) is the aircraft weight and \( r_c \) is Earth’s local radius [7]. Using \( C_L = L/({\dot{q}}S) \), where \( \dot{q} \) is the dynamic pressure, the angle of attack \( (\alpha) \) is solved for using the pre-computed aerodynamic data, as is \( C_D = D/({\dot{q}}S) \). The sum of parallel forces about the center of gravity for the aircraft in translational flight, with center line thrust, is given by

\[ T - D - mg \sin \gamma = m \dot{V}_a \quad (6) \]

which provides the value of thrust, \( T \), used in the energy and power model. The current article considers only the case of trajectory optimization in the absence of winds. Latitude and longitude \((\xi, \lambda)\) are computed from \((p_n, p_e, h)\) using a WGS84 Earth model.

C. Solar Model

A solar model is used to enable computation of solar flux at the surface of the vehicle’s onboard photovoltaic (PV) cells. For our purposes, the solar flux is defined as the amount of solar power radiated through one direction of a given surface (without subtracting any flux in the opposite direction), given in W/m². Solar angles of azimuth \( (\theta_a) \) and elevation \( (\epsilon_s) \), and Julian day \( (j_d) \), are computed from \((\xi, \lambda, h)\) and date/time per the algorithm described by Reda and Andreas [8]. Solar irradiance is then adjusted to account for annual variation (due to eccentricity of Earth’s orbit) and atmospheric absorption, with the formula

\[ I = I_0 \left(1 + 0.034 \cos \frac{2\pi j_d}{365} \right) \quad (7) \]

where \( f(h, \epsilon_s) \) is an atmospheric absorption factor [9].

Solar flux through the vehicle’s photovoltaic cells can then be calculated based on geometric relationships between the vehicle coordinate frame, PV cells, and local solar vector (magnitude and direction of the solar irradiance). For a PV cell with area \( A_k^c \) and unit normal vector \([x_k, y_k, z_k]^T\) in the Forward – Right Wing – Down coordinate system, the solar flux through this cell area is given by

\[ \Phi_k^c = A_k^c \left[ -c \phi_s c \epsilon_s \quad -s \phi_s c \epsilon_s \quad s \epsilon_s \right] R_k^T \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} \quad (8) \]

where

\[ R_k^T = \begin{bmatrix} c \psi & -s \psi & 0 \\ \psi & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c \theta & 0 & s \theta \\ 0 & 1 & 0 \\ -s \theta & 0 & c \theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c \varphi & -s \varphi \\ 0 & s \varphi & c \varphi \end{bmatrix} \quad (9) \]

using shorthand \( c\alpha \) and \( s\alpha \) to represent \( \cos (\alpha) \) and \( \sin (\alpha) \). Note that the airframe model we use has area-weighted PV cell normal vector at a 7° pitch angle relative to the vehicle’s Forward – Right Wing – Down coordinate frame.

D. Power and Energy Storage

Modeling of the flow of electric power in the aircraft model is done as follows. Electric power input is given by

\[ P_{in} = \eta_s \Phi_s \quad (10) \]

where \( \eta_s \) is the solar collection system efficiency and \( \Phi_s = \sum_k \Phi_k^c \) is solar flux through the areas of the vehicle’s PV cells.

Power may be stored in the vehicle’s batteries, or expended through either the electric propulsion system or “accessory” power system (modeled with a constant power draw).

Power output is given by

\[ P_{out} = P_{acc} + \eta_p TV_a \quad (11) \]

where \( P_{acc} \) is accessory system power and propulsion system efficiency, \( \eta_p \), is given by

\[ \eta_p = \frac{2\eta_{p,0}}{1 + \sqrt{\frac{q}{NA_{acc}}}} + 1 \quad (12) \]
where $A_d$ is propeller disk area, $N$ is the number of propellers, and $\eta_{0, 0}$ is a constant efficiency factor which accounts for motor controller and motor efficiency. These relationships allow for the computation of power requirements to produce a given thrust at a given airspeed.

The battery system is assumed to have efficiencies $\eta_{in}$ and $\eta_{out} < 1.0$ for charging and discharging, respectively, as well as a maximum energy storage capacity. Energy stored in the batteries is thus

$$E_{batt}(t) = \int_{t_0}^{t} \lambda P_{net} + E_{batt}(t_0)$$

$$\lambda = \begin{cases} 
\eta_{in} & P_{net} \geq 0 \text{ and } E_{batt} < E_{max} \\
\eta_{out} & P_{net} < 0 \text{ and } E_{batt} > 0 \\
0 & \text{otherwise}
\end{cases}$$

(13)

Note that it is possible to store energy as gravitational potential by converting electric power to thrust used to increase the vehicle’s altitude. Kinetic energy storage is not included in this model due to its relatively small impact within aircraft velocity constraints.

### TABLE I

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Station-keeping lat, lon ($\xi, \lambda$)</td>
<td>(34.2°, −118.5°)</td>
</tr>
<tr>
<td>Battery energy density</td>
<td>320 Wh/kg</td>
</tr>
<tr>
<td>Battery capacity ($E_{max}$)</td>
<td>64 kWh</td>
</tr>
<tr>
<td>Maximum thrust ($T_{max}$)</td>
<td>1000 N</td>
</tr>
<tr>
<td>$\eta_{0, 0}$, $\eta_{in}$, $\eta_{out}$</td>
<td>0.79, 0.22, 0.93, 0.97</td>
</tr>
<tr>
<td>Accessory power draw ($P_{acc}$)</td>
<td>360 W</td>
</tr>
<tr>
<td>Solar constant ($I_0$)</td>
<td>1367 W/m²</td>
</tr>
<tr>
<td>Wing planform area</td>
<td>136.75 m²</td>
</tr>
<tr>
<td>Wing aspect ratio</td>
<td>4.1</td>
</tr>
<tr>
<td>Aircraft mass ($m$)</td>
<td>571 kg</td>
</tr>
<tr>
<td>Surface area of solar cells</td>
<td>143.3 m²</td>
</tr>
<tr>
<td>Max $N$–E plane distance ($d_{max}$)</td>
<td>3000 m</td>
</tr>
<tr>
<td>$h_{min}$, $h_{max}$</td>
<td>18290 m, 24574 m</td>
</tr>
<tr>
<td>$v_{max}$</td>
<td>0.8 m/s</td>
</tr>
<tr>
<td>$\phi_{max}$</td>
<td>10°</td>
</tr>
<tr>
<td>$V_{min}$ (EAS), $V_{max}$ (EAS)</td>
<td>7.1 m/s, 11.5 m/s</td>
</tr>
</tbody>
</table>

### III. Optimization Framework

In this section, the mathematical programs which we use to optimize the HALE trajectory are described. The operating scenario which we consider is a single HALE aircraft on a **station-keeping** mission, where the vehicle must remain within a certain distance from a fixed vertical axis, and must remain above a minimum altitude. The trajectory optimization is applied the **winter solstice** for this station-keeping mission, defined as the day with the lowest peak solar elevation angle. In particular, we define our optimization goal as the maximization of net energy storage for a time period $(t_i, t_f)$, for example a twenty-four hour period starting at sunrise on winter solstice. Recall that energy is stored both within the batteries and as gravitational potential energy.

The optimization problem is expressed mathematically as

$$\max_{u(t)} J = [E_{batt}(t_f) - E_{batt}(t_i)]$$

$$\text{s. t. } [\dot{h}^c(i) \leq \dot{h}_{max}$$

$$|\dot{\varphi}^c(i)| \leq \varphi_{max}$$

$$V_{min} \leq V_{eas}^c(i) \leq V_{max}$$

$$0 \leq E_{batt}(i) \leq E_{max}$$

$$0 \leq T(i) \leq T_{max}$$

(14)

where $J \in \mathbb{R}$ is an energy difference cost function, $u(t)$ is defined in Eq. 2, $E_{batt}$ and $E_p$ are energies stored as electrical energy in the batteries and as gravitational potential energy, respectively, and $\kappa$ is a scaling term on the potential energy.

In order to solve this problem, the optimization time period is discretized into $n$ equally-spaced segments. The optimization variables are thus the magnitudes of bank attitude command $\varphi^c[i]$, altitude rate command $\dot{h}^c[i]$, and airspeed command $V_{eas}^c[i]$ at knot points $i \in (1, \ldots n)$. These comprise the $(3 \times n)$ optimization variables used for the shooting method of trajectory optimization [10], which was selected over other trajectory optimization methods due to the complex and nonlinear dynamics, and the lack of final-state constraints. The discretization of command trajectories is in general coarse compared to the time scales of the system dynamics, and thus piecewise polynomial interpolation is used for the command trajectories. In particular, unless otherwise stated, our optimization programs use a first-order polynomial (linear) interpolation of commands between knot points.

The nonlinear optimization problem is constrained as described in Eq. 14, where it is noted that the first three constraints comprise upper and lower bounds on the optimization variables, while the remaining constraints are nonlinear in the optimization variable space. These nonlinear constraints are handled via quadratic penalization, in which the optimization cost function is augmented with quadratic penalties on the violation of these constraints. The discretized optimization problem, with the quadratic penalties, becomes

$$\max_{u(t)} J = [E_{batt}[n] - E_{batt}[1]] + \kappa [E_p[n] - E_p[1]]$$

$$+ C_h \sum_i R(h_{min} - h[i])^2 + R(h[i] - h_{max})^2$$

$$+ C_d \sum_i R(d[i] - d_{max})^2$$

s. t. $[\dot{h}^c[i] \leq \dot{h}_{max}$$

$|\dot{\varphi}^c[i]| \leq \varphi_{max}$$

$V_{min} \leq V_{eas}^c[i] \leq V_{max}$$

$0 \leq E_{batt}[i] \leq E_{max}$$

$0 \leq T[i] \leq T_{max}$$

(15)
where $[\cdot]$ represents evaluation at all knot points $i \in (1, \ldots, n)$, $R(\cdot)$ represents the unit ramp (rectifier) function, $d[i] = \sqrt{p_{x[i]}^2 + p_{y[i]}^2}$, and $C_\alpha$ and $C_\delta$ are penalty weights.

In Section III-A, we describe a solution to this nonlinear optimization problem via single shooting using the gradient-based interior point (IP) algorithm, and present results from this optimization. In Section III-B, we describe a solution to the same problem using the Nelder-Mead Nonlinear Simplex algorithm, and present results. We then present results on trajectory optimization in the presence of wind, and discuss real-time, online optimization considerations.

### A. Interior Point Optimization

Starting with a general nonlinear optimization problem

$$\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad f(x) \\
\text{s. t.} & \quad c_i(x) \geq 0, \quad i \in (1, \ldots, m),
\end{align*}$$

(16)

slack variables $w$ can be introduced to transform the inequality constraints, $c_i(x) \geq 0$, into an equality constraint of the form $c(x) - w = 0$. The constraint on the slack variable itself can be absorbed into the cost function by defining logarithmic barrier cost function $\sum_{i=1}^m \ln(w_i)$:

$$\begin{align*}
\min_{\mu, w, x} & \quad f(x) - \mu \sum_{i=1}^m \ln(w_i) \\
\text{s. t.} & \quad c(x) - w = 0
\end{align*}$$

(17)

where $\mu$ is a small positive scalar. Introducing Largrange multiplier vector $\lambda$, the Lagrangian function is given by

$$L = f(x) - \mu \sum_{i=1}^m \ln(w_i) - \lambda^T(c(x) - w).$$

(18)

The first-order optimality conditions are found by setting the derivatives of the Lagrangian with respect to $x$, $w$, and $\lambda$ equal to zero (Karush-Kuhn-Tucker conditions):

$$\begin{align*}
\nabla f(x) - \nabla c(x)^T \lambda &= 0 \\
W \lambda + \mu e &= 0 \\
c(x) - w &= 0
\end{align*}$$

(19)

Search directions $d^e$, $d^w$, and $d^\lambda$ can then be computed using the Newton-Raphson method as described by

$$\begin{bmatrix}
H & 0 & -\nabla c(x)^T \\
0 & W & 0 \\
\nabla c(x) & -I & 0
\end{bmatrix}
\begin{bmatrix}
d^e \\
d^w \\
d^\lambda
\end{bmatrix} =
\begin{bmatrix}
-\nabla f(x) + \nabla c(x)^T \lambda \\
\mu e - WA \lambda \\
-c(x) + w
\end{bmatrix}$$

(20)

with

$$H = \nabla^2_{xx} L(x, w, \lambda) = \nabla^2_{xx} f(x) - \sum_{i=1}^m \lambda^T \nabla^2_{xx} c(x)$$

(21)

$$W = \begin{bmatrix}
w_1 & 0 & 0 & \cdots & 0 \\
0 & \ddots & 0 \\
0 & 0 & w_m
\end{bmatrix}, \quad \Lambda = \begin{bmatrix}
\lambda_1 & 0 & 0 & \cdots & 0 \\
0 & \ddots & 0 \\
0 & 0 & \lambda_m
\end{bmatrix}$$

The optimization can then be carried out until successive iterations satisfy some pre-defined convergence criteria. It is noted that due to the nonlinearity of the problem, it is not possible to guarantee global optimality of the solution.

### 1) Trajectory Optimization with the Interior Point Method

The nonlinear trajectory optimization problem is solved using the interior point method provided by MATLAB’s commercial solver fmincon. As the problem dimension is large and highly nonlinear, initial input trajectories are provided to the nonlinear solver to aid in convergence to a feasible solution. The initial input trajectory (which is also our baseline for benchmarking purposes) produces a circular ground-track of radius $d_{max}$ at altitude $h_{min}$, and at the equivalent airspeed $V_{eq}$, which minimizes drag power. This baseline is the constant-command trajectory which minimizes power draw while satisfying all constraints. The simulation results presented in this article are obtained for the worst-case scenario corresponding to operation during the winter solstice.

Figure 2 visualizes the results of trajectory optimization over an hour-long period ($t_i = 9$am, $t_f = 10$am) on the winter solstice day. The time interval is split into $n = 180$ knot points, such that the knot points are spaced at 20 second intervals, and there are 540 resulting optimization variables. In this figure, the black circle represents the bottom of a cylinder in which the trajectory is constrained via quadratic penalization. In the $x - y$ trajectory plots, the vertical axis is representative of the north direction and the horizontal axis is representative of the east direction. The altitude axis is also included in the $x - y - h$ plots, where it is noted that the $h$-axis is not to scale with the $x$ and $y$ axes. The color of the trajectory represents a relative airspeed, where warmer colors indicate a lower velocity. The direction of the sun in the $x - y$ plane is illustrated by yellow arrows (moving clockwise as the day progresses).

In this hour-long example, the optimized trajectory increases net energy storage by approximately 12% compared to that of the baseline trajectory (constant command trajectory which minimize power draw). The solver reaches convergence in approximately 19 hours on a workstation with parallelization on 6 CPU cores, which is a baseline for comparison with other optimizations discussed in this article.

Figure 3 shows the result of trajectory optimization for another hour-long period ($t_i = 2$pm, $t_f = 3$pm) on the winter solstice day. This optimized trajectory improves net energy storage by 15% compared to the baseline trajectory. In both of these examples, the energy-optimized trajectories have some periodic behavior, combined with a rotation to follow the sun. A period of this behavior can be described qualitatively as: climbs at relatively low velocity away from the sun, rapid turn towards the sun, descent towards the sun at relatively high velocity, rapid turn away from the sun. The relatively low velocity when climbing away from the sun produces a higher angle of attack and therefore higher solar flux through the area of the vehicle PV cells. While descending, the higher velocity and lower angle of attack similarly tilts the solar cells towards the local solar vector.

### B. Nonlinear Simplex Optimization

In this section, we will present results of trajectory optimizations solved using the nonlinear simplex optimization
algorithm, developed by Nelder and Mead [11]. The nonlinear simplex optimization algorithm is gradient-free (it is a direct search method), and is thus effective in many practical optimization problems in which Jacobians and Hessians may be computationally expensive or non-existent.

The basis of the method is to estimate the objective function gradient by the use of simplex. Simplex $S \in \mathbb{R}^n$ is defined as a convex envelope of $n + 1$ vertices $x_0, \ldots, x_n \in \mathbb{R}^n$. The simplex-based search algorithm is initiated with a nondegenerate simplex and the associated set of cost function values at its vertices, $f_j = f(x_j)$. For a general minimization problem, the goal is to decrease the cost function values at the working simplex vertices with successive transformations of the simplex, consisting of

1) Reflection of vertex $x_h \in S$ with worst cost function value ($f_h$) to new vertex $x_r$ through the line defined by $x_h$ and the simplex centroid ($c$).

2) Depending on value $f_r$:
   a) Expansion along line ($c, x_r$) if $f_r$ is best, to new vertex $x_r$. Use expanded simplex if $f_e$ is best.
   b) Contraction along line ($c, x_r$) if $f_r$ is worst or second-worst.

3) Shrinkage of the simplex by moving vertices towards best vertex $x_l \in S$.

Note that unlike the interior point method, the explicit computations of Jacobians and Hessians are not required, yet the transformations of the simplex are designed in such a way that the working simplex should roughly follow the direction of steepest descent. A detailed description of the procedure is omitted in this article, but can be found in [11] and [12].

1) Trajectory Optimization with Nonlinear Simplex: Figure 5 illustrates the trajectory optimization for the hour-long period ($t_1 = 9$am, $t_f = 10$am) on the winter solstice day,
solved using the Nelder-Mead nonlinear simplex algorithm. The optimization problem is identical to that presented in Fig. 2, with the difference being the use of the nonlinear simplex method instead of the interior point method. The time to convergence is approximately 1 hour on the same workstation used for interior point benchmarking (a 20× improvement over the interior point method). Using the nonlinear simplex algorithm, the optimal trajectory improves net energy storage over the baseline trajectory by approximately 7% (compared to 12% with the interior point method).

Figure 6 compares the hourly net energy storage from the optimized and baseline trajectories. Note that the baseline trajectory in these simulations undergoes a mode switch when the batteries are full: any excess solar power collection is used to increase the altitude of the vehicle to store gravitational potential energy, which is then expended after sunset until the vehicle reaches \( h_{min} \). The overall improvement in net energy storage of the energy-optimized trajectory planning adds up to 10.5 kWh of additional energy storage over the winter solstice day (5.9 kWh with nonlinear simplex), which is enough to extend the maximum night (the time with no solar power collection) by approximately 160 minutes. At the latitude chosen for analysis, both the baseline and optimized trajectories are able to survive the winter solstice night, however the battery state-of-charge margin is significantly larger with the energy-optimized trajectory.

C. Trajectory Optimization in Wind

Winds introduce both additional complexities and opportunities for the energy-optimized trajectory planning problem. Wind is modeled according to the equations given in Eq. 4, and trajectories are optimized using the interior point algorithm provided by MATLAB’s fmincon solver. Optimizations have been carried out for different wind directions and magnitudes (given by gray vectors), up to the 99th percentile wind magnitude provided by a wind distribution model at the tested location and altitude. For the latitude, longitude, and altitude of the station-keeping orbit considered in this article (see Table I), this magnitude is approximately 24 m/s. It is assumed in all cases that \( w_d = 0 \), producing wind vectors in the \( x - y \) plane.

Figure 7 portrays the trajectory optimization for \((t_i = 9\text{am}, t_f = 11\text{am})\) with the wind vector with magnitude \( V_w = 22 \text{ m/s} \) and direction 260° (the solar azimuth angle varies from 138° to 151°). The optimizer is initialized with an input trajectory which will approximately follow a circular ground track at the drag-power-minimizing airspeed (as long
as the magnitude of the airspeed is higher than that of the wind). This corresponds to the baseline initialization used in preceding sections, with commands modified to account for the magnitude and direction of the wind. As the aircraft tries to maintain its velocity close to the optimal airspeed, it will spend the majority of its time flying in the straight paths upward. The trajectory optimization in this example improves net energy storage over the baseline trajectory by less than 1%. We note that the improvement in energy storage achieved by the optimizer decreases with wind speeds, which may be due to the increased complexity involved in solving with a single shooting method, or the baseline trajectory may approach optimal behavior in high winds.

One alternative is to encapsulate behavior of the trajectory optimization algorithm into a number of rules based on the trajectory optimization via shooting, which could then be implemented as a state machine on the onboard flight control computer. For low wind speeds, these rules could be summarized at a high level as:

1) With the sun above the horizon, and $E_{\text{batt}} < E_{\text{max}}$, perform “racetrack”-shaped periodic climbs and descents, with the major axis aligned with the sun, and with periodic airspeed variation, until $E_{\text{batt}} = E_{\text{max}}$

2) With the sun above the horizon, and $E_{\text{batt}} = E_{\text{max}}$, climb at the maximum rate possible which will not draw power from the batteries

3) With the sun below the horizon, and $h > h_{\text{min}}$, descend (using an optimal combination of electric and potential energy) to $h = h_{\text{min}}$

4) With the sun below the horizon, and $h = h_{\text{min}}$, perform circular orbits at the constant drag-power-minimizing airspeed

IV. MACHINE LEARNING FOR TRAJECTORY PLANNING

One particularly promising approach to “compress” the trajectory optimization for use online is to take advantage of reinforcement learning techniques to learn the optimal behavior based on a set of inputs. This has the effect of shifting the computational burden from the online planning phase to an offline training and validation phase, where computational power is cheap and readily available. One such method, which we consider here, is the adaptive neuro-fuzzy inference system (ANFIS) [13], [14]. ANFIS is a particular type of universal approximator which combines a multi-layer feedforward neural network with fuzzy logic. The idea is to

1) Generate a large set of trajectories using the optimization methods provided above, for a variety of different conditions;

2) Train and validate the ANFIS system using this offline trajectory optimization data;

3) Use the ANFIS system for trajectory planning online.

Six input variables are provided for the trajectory planning ANFIS, namely

- $(u_1)$ The angular difference between the solar azimuth and aircraft course angle $\Delta \chi = (\chi - \phi_a)$
- $(u_2)$ The aircraft distance from the station-keeping maximum radius $d_{\text{perim}} = d_{\text{max}} - \sqrt{r_n^2 + r_e^2}$
- $(u_3)$ The rate of change of $d_{\text{perim}}$
- $(u_4)$ The battery state of charge ($E_{\text{batt}}/E_{\text{max}}$)
- $(u_5)$ Direction of the wind
- $(u_6)$ Magnitude of the wind

Outputs of the trajectory planning ANFIS are

- $(y_1)$ Command for rate of change of heading $(\dot{\chi}_{\text{cmd}})$
- $(y_2)$ Command for altitude rate $(\dot{h}_{\text{cmd}})$

The airspeed command is taken to be constant for trajectory planning using ANFIS.

Using ANFIS, we are able to produce a closed-loop formula to compute the desired commands, which can potentially be implemented in-flight on the aircraft flight control computer.
A. Trajectory Planning with ANFIS

Figure 8 demonstrates the performance of the trained ANFIS in energy-optimized trajectory planning for the three-hour period (t_i = 9am, t_f = 12pm) on the winter solstice day. The improvement in net energy storage over the baseline trajectory is approximately 12%, which is comparable to the results of the offline optimization routine. The use of the learned ANFIS model to compute trajectories given the inputs provided earlier in this section results in processing time reduction of roughly four orders of magnitude compared to running the offline optimization routines (providing outputs in seconds instead of hours).

Fig. 8. Result of application of learned ANFIS model to trajectory optimization from 9:00am–12:00pm.

To validate the ANFIS trajectory planner and examine its robustness, more simulations were carried out in which initial conditions – including initial heading of the aircraft – are varied. This is to ensure that the trained ANFIS provides invariance to these variables, and that it is able to handle flight scenarios which have not been included in its training data. The results of these simulations indicate that the ANFIS is effective in handling a variety of initial conditions and maximize net energy storage as it is intended to do.

V. Conclusions

This work investigates the problem of energy-optimized trajectory planning for solar-powered HALE aircraft. Optimization via shooting methods is first employed to optimize command trajectories, with solvers using the interior point method and the nonlinear simplex method. Results are presented and compared for these two solvers, and the resulting optimal trajectories for the winter solstice day are discussed. The problem of trajectory optimization in the presence of constant winds is addressed. These trajectory optimization routines are used to train an adaptive neuro-fuzzy model, which is demonstrated to produce trajectories for the station-keeping mission without wind, with nearly the same energy saving benefits as the nonlinear optimization procedures, but approximately four orders of magnitude faster. This provides a potential avenue for online energy-optimized trajectory planning which can be carried out at low computational cost on an onboard flight computer.

VI. Acknowledgments

The authors would like to thank Bruce Lin, Nicholas Roberts, David Liu, Xiangfei Meng, and Martin Gomez of Facebook for their important contributions to this work.

References