

FLIGHT PATH OPTIMISATION

BOB VERGAUWEN AND MORITZ WOLTER



Project Report

Supervised by Prof. dr. ir. Toon van Waterschoot
December 15, 2014 – Version 1

MOTIVATION AND IMPLEMENTATION

1.1 INTRODUCTION

In this project we are going to find the optimal path for a solar powered airplane ¹. During flight the plane's electrical engines drain power from it's battery, while at the same time solar panels on the plane's wings convert energy from the sun's rays into electric energy. However clouds may block the sun's rays from reaching the panels. In addition the sun's intensity increases when flying closer to the earth's equator. In order to keep as much energy in the plane's batteries as possible the flight path has to be optimized.

1.2 PROBLEM FORMULATION AND ANALYSIS

1.2.1 *Time invariant case*

In the optimization process we use randomly generated weather data and sun data corresponding to a flight between 48° and 50° latitude. We generate the weather data by cubically interpolating a random matrix of size n over a grid. The larger n becomes the more complex the weather will be. This functionality is implemented in `Static_Weather.m` **repo is still private...** ² The class returns a weather data matrix W .

Data for the sun comes from a couple of established equations that relate the local time and position to the sun's intensity which happens in `sun.m`. The class returns a sun data Matrix S . We define low values in the weather Data Matrix as cloudy and high values as sunny weather. As clouds reduce the electric yield of our plane's solar cells we define the overall intensity as:

$$I = W.*S \quad (1)$$

While following a convention where `.*` denotes element wise multiplication we end up with an intensity matrix I , which contains sun intensity data weighted according to the random weather.

¹ Like the one from <http://www.solarimpulse.com/>

² All matlab code can be found on <https://github.com/double2double/OptimisationProject/> in the MATLAB folder.

Next we took a closer look at the airplane. Here we consider the degree to which the plane can recharge its battery during flight, which is the path integral over the intensity data:

$$E(\mathbf{x}) = \oint_{\Gamma} I \, d\mathbf{x}. \quad (2)$$

With $d\mathbf{x}$ being the length of each step. Having found the additional energy we are able to convert during flight we now look at losses, which are due to drag and acceleration. We define acceleration cost in the time invariant case as the scalar product of the vectors containing local accelerations:

$$A(\mathbf{x}) = \mathbf{a}^T \cdot \mathbf{a} \quad (3)$$

To calculate the derivatives in the time invariant case we approximate dt as the total time over distance traveled.

However we only count cases where the acceleration is positive since deceleration does not drain battery power. Finally we consider the drag force which the plane experiences on its path, which we approximate as the dot product of the local speed vector:

$$D(\mathbf{x}) = \mathbf{s}^T \cdot \mathbf{s} \quad (4)$$

Having investigated the approximate battery power lost and gained on a specific path, we can now proceed to formulating a cost function for the time invariant case.

1.3 COST FUNCTION FORMULATION

1.3.1 Time invariant case

In the time invariant case we define the optimal path as the path from start to destination where we lose as little battery power as possible. The problem parameter is a path vector $\mathbf{x} \in \mathbb{R}^m$. Which contains m points on the path. Introducing weights for the different power gains and losses we define the cost function:

$$V(\mathbf{x}) = -E(\mathbf{x}) + 0.01A(\mathbf{x}) + 200D(\mathbf{x}). \quad (5)$$

Which is subjected to the constraint that the path may not leave the weather data grid. This translates into $0 \leq x \leq 1$ and $0 \leq y \leq 1$ for the elements of $\mathbf{x} = (x \, y)^T$. We solved this problem starting with the straight line from (0.1,0.5) to (0.9,0.5) as initial input using matlab's `fmincon` solver. This cost function is implemented in the `Airplane.m` class file.

RESULTS

2.1 TIME INVARIANT CASE

The optimized path (shown in red) for the time invariant case is shown in figure 1. We observe that the path leads nicely through sunny weather grids including a detour to the sunniest spot on the grid at (0.5,0.35), indicating that our problem is well formulated.

2.2 TIME VARYING AIR PLANE

In the previous section we performed an elaborate explanation of the a first model of the air plane. This model did not include time and it was assumed that the air plane flew with a constant speed. This however is not realistic, we want to stay longer in sunny regions and fly fast through cloudy ones. To compensate for this need it is a good idea to include the time as an input parameter as well. This new freedom will introduce a new dimension and thus make the optimisation problem more complex.

2.2.1 Formulation of the problem

Lets start by reformulating the optimisation problem in a formal way.

$$\min_{x,y,t} (-\text{Energy}(x,y,t)) \quad \text{s. t.} \quad (6)$$

$$x \in [0,1]^m \quad (7)$$

$$y \in [0,1]^m \quad (8)$$

$$[x[0],y[0]] = [x_0,y_0] \quad (9)$$

$$[x[m],y[m]] = [x_m,y_m] \quad (10)$$

$$t_0 = 0 \quad (11)$$

$$t(i) \leq t(i+1) \forall i \in \{1,2 \dots m\} \quad (12)$$

The conditions for the spacial coordinates have stayed the same, the plane has to stay in a certain region. By introducing time dependency we have to make sure that the time is increasing throughout the flight, without this constrain the solution would be nonsense.

Lets now look at how the introduction of time changes the Energy function, we are still taking in to account the same quantities as in the previous model; the positive acceleration energy, the air resistance and the solar gain. The only thing that really changed is the way the

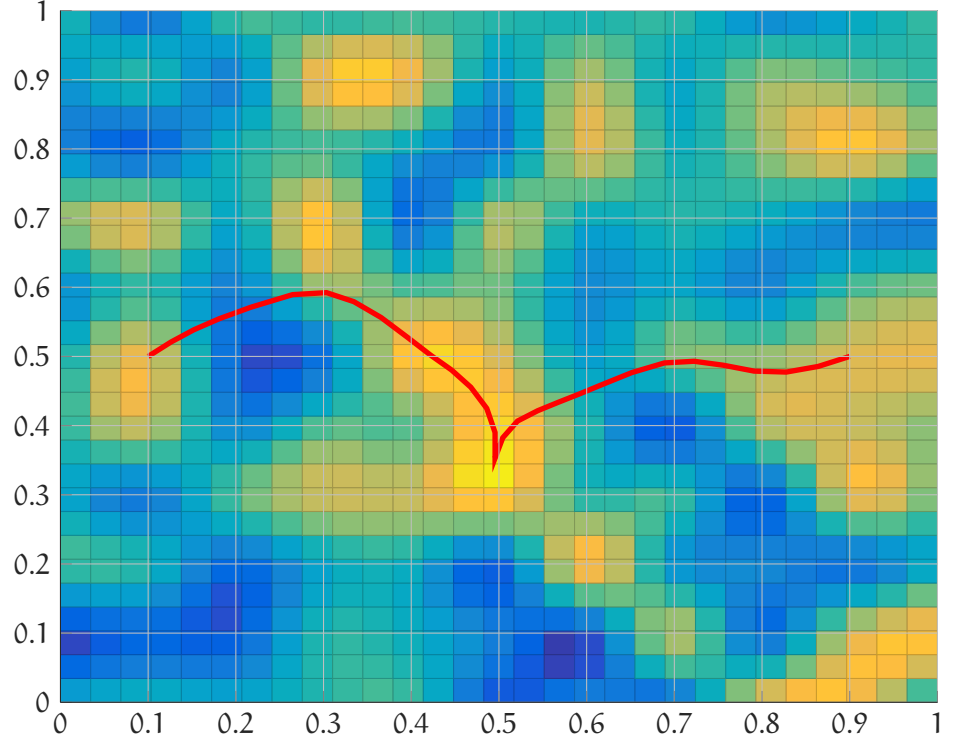


Figure 1: Optimized flight path from the coordinates 0.1,0.5 to 0.9,0.5, with different weather data.

speed of the air plane is defined. In the previous model we assumed that each step taken by the plane took an equal time step regardless of the length of this step, this has changed in the time varying model, the time step is now a parameter that could be chosen optimal. In a more mathematical formulation we took the speed to be defined as

$$v[i] = \frac{l[i+1] - v[i]}{l[i+1] - t[i]}, \quad (13)$$

in this formulation is l the step path length