

Gathering Atmospheric Data

Using an Unmanned Air Vehicle

Henry Miskin

March 7, 2014

Abstract

This report looks at three dimensional energy based path planning for unmanned air vehicles in a predetermined area, with particular consideration to quality of data produced.

Contents

| | | |
|----------|---|----------|
| 1 | Introduction | 3 |
| 2 | Plane Properties | 3 |
| 3 | Energy Model | 3 |
| 4 | Latin Hypercubes | 3 |
| 5 | Route Planning | 4 |
| 5.1 | Exact Travelling Salesman | 5 |
| 5.2 | Improving Travelling Salesman | 6 |
| 5.3 | Progressive Travelling Salesman | 7 |
| 6 | Path Planning | 8 |

1 Introduction

The following section outline the process in which a minimum cost route through a sample space can be obtained that provides the best data collection quality

2 Plane Properties

| | |
|-------------|-----|
| Wingspan | 3 |
| Turn Radius | 0.1 |

Table 1: Table of Plane Properties

To consider the minimum cost of circumnavigating a particular route the specifications of a plane must be considered in table 1 the plane detailed in this table is the plane that is used for the entire report

3 Energy Model

$$E = \alpha D + \beta H \tag{1}$$

From these plane properties the following energy model has been defined in equation 1 where α and β are coefficients that are determined by the plane. For the current plane shown in table 1 α and β take values of 10 and 6 respectively.

4 Latin Hypercubes

Latin hypercubes are sampling plans that provide the best space fillingness while limiting the total number of sampling points required. This is generally applied to testing of computer simulations where the collection of each point is expensive. In this situation however the travel between the points is the expensive component.

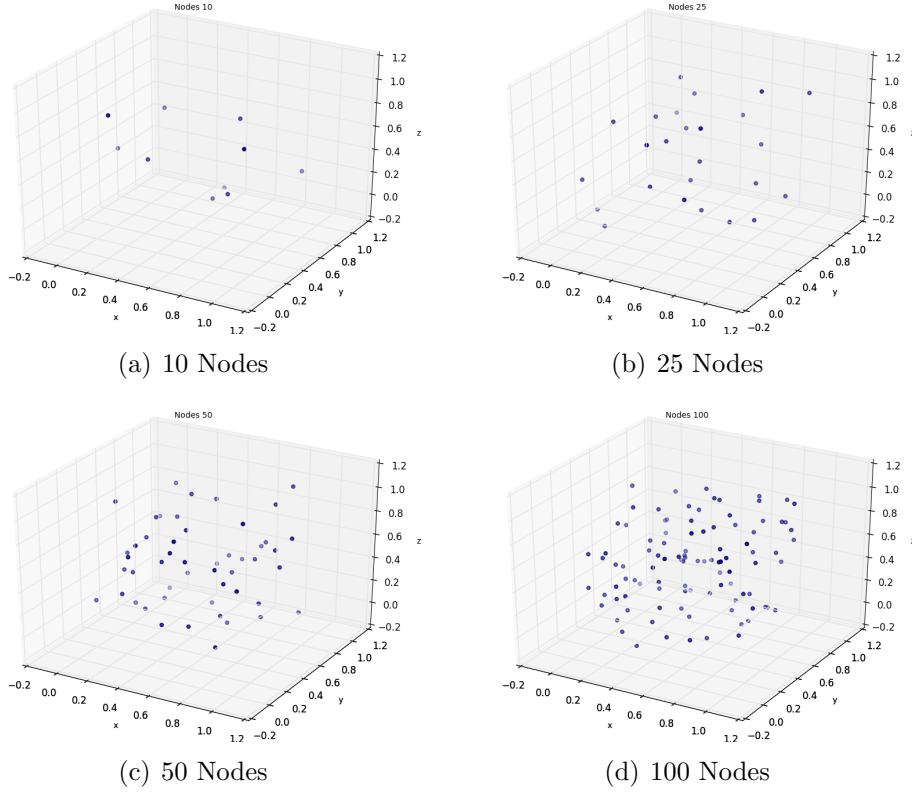


Figure 1: Latin Hypercubes with Varying Numbers of Nodes

Figure 1 shows a number of latin hypercubes with differnt numbers of nodes. All the Latin Hypercubes are within a unit cube. For collection of data in a required area these cubes can be stretched to fill the desired space. This does not provide an even spacing in each direction however means that eeach vertex of data collection is equally considered.

For this project the idea is to follow this logic to utilise Latin Hypercubes:

1. Specifiy area of interest to researcher
2. Estimate number of nodes able to be circumnavigated given the UAV total energy and the area of sample space
3. Fit Latin Hypercube of given nodes to sample area
4. Calcualte least energy route through the sample space
5. After first flight asses areas of encertainty to plan route through for next flight

5 Route Planning

Given a set of points within a sample space the next stage of the proceedings is to compute the least cost route through these points. This problem presents itself

in the form of the travelling salesman problem. The travelling salesman problem is the problem of finding the least cost route through a set of points. There is lots of work done on the euclidean travelling salesman problem and introducing heuristics to improve the time taken to compute. This is due to the problem being an NP hard problem (the computing time required increases exponentially with the number of points in the route)

5.1 Exact Travelling Salesman

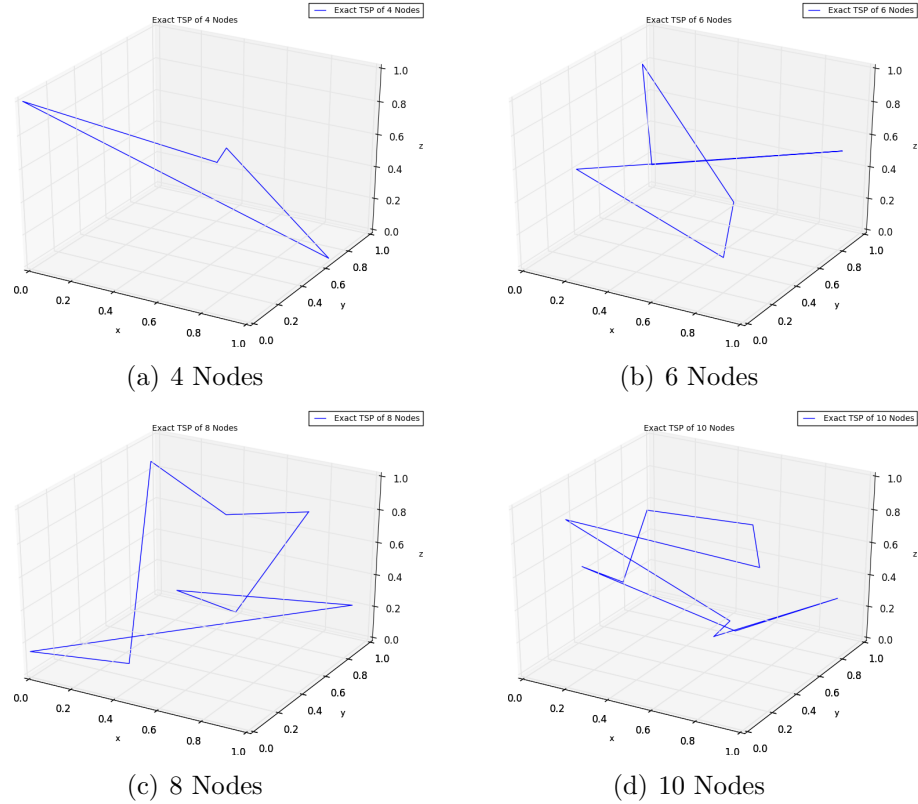


Figure 2: Exact routes calculated by travelling salesman

Figure 4 shows the optimal routes for different numbers of nodes. These optimal routes are found by computing the exact cost of each and every route option. Although this yields the shortest routes this approach is not efficient in terms of the computation time required.

| | | | | |
|---------------------------|--------|--------|--------|---------|
| Number of points | 4 | 6 | 8 | 10 |
| Number of possible routes | 24 | 720 | 40320 | 3628800 |
| Computation time (ms) | 1.0 | 5.0 | 315.0 | 33267.0 |
| Best route cost (J) | 110.38 | 106.71 | 106.72 | 108.76 |

Table 2: Comparison of route calculation

Table 3 shows the number of possible routes and the resulting computation time given different numbers of nodes. It can easily be seen that the number of route options is

equivalent to $n!$ where n is the number of nodes. The number of routes directly relates to the computation time.

The computation time of the exact TSP approach can be reduced in a number of ways. Primarily the start node of the calculation can be defined however this only reduces the complexity to the equivalent of removing a node from the computation. Other approaches involve producing a best guess and improving upon that. The approach taken in this project is taken from considering the best routes in figure 4 generally comprise of a climbing component and a descending component.

5.2 Improving Travelling Salesman

The progressive travelling salesman is an approach to computing a best guess route for the least energy route for a number of points. The computation process is as follows

1. Order the nodes by their vertical location from lowest to highest
2. Considering the lowest N nodes compute all possible combinations to produce two routes from the given nodes
3. Compare all permutations of routes to return the least cost combination
4. Add first nodes of both routes to final route
5. Consider lowest N nodes that are not in the final route and reiterate
6. Work through all nodes until no nodes are left without a route

| | | | | |
|-----------------------|--------|--------|--------|--------|
| Nodes in each route | 2 | 3 | 4 | 5 |
| Computation time (ms) | 2.0 | 3.0 | 52.0 | 4091.0 |
| Best route cost (J) | 120.49 | 115.48 | 115.48 | 114.79 |

Table 3: Comparison of route calculation

Table 3 shows the varying computation time for routes through a 10 node latin hypercube that have a different numbers of nodes in each route. This refers to the number of nodes that are considered in each progressive iteration of the code.

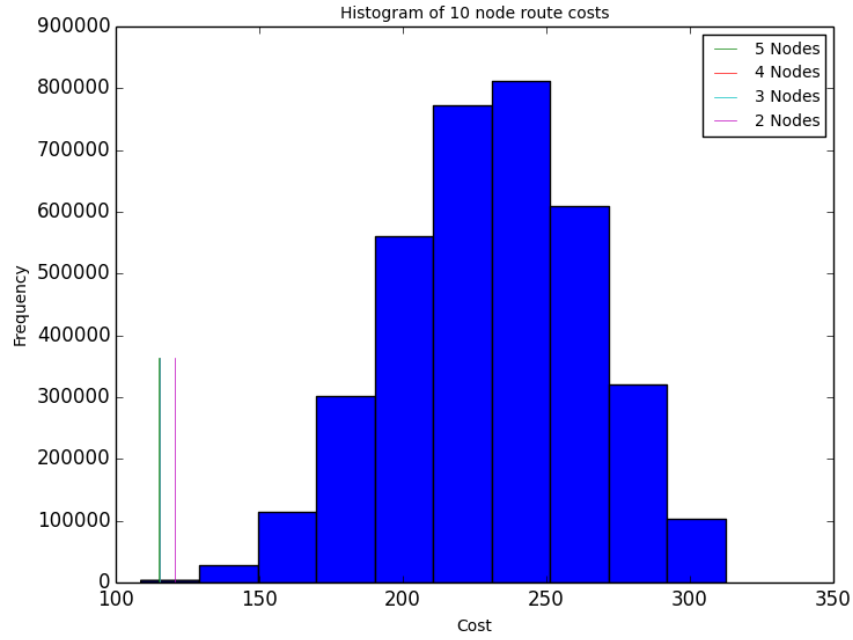


Figure 3: Histogram of 10 node route costs

Figure 3 shows a histogram of different route costs for a 10 node latin hypercube. The lines on this histogram plot represent the best cost routes with different numbers of nodes in the route

5.3 Progressive Travelling Salesman

In section 5.3 a best guess approach to the travelling salesman was presented and compared with the exact results. Given the results were close to the optimum even when the progressive algorithm only considered 2 nodes in the route it can be brought forward to be considered in reference to larger route problems. In these routes 4 nodes are considered for both the up route and the down route are used at each stage of the progressive algorithm.

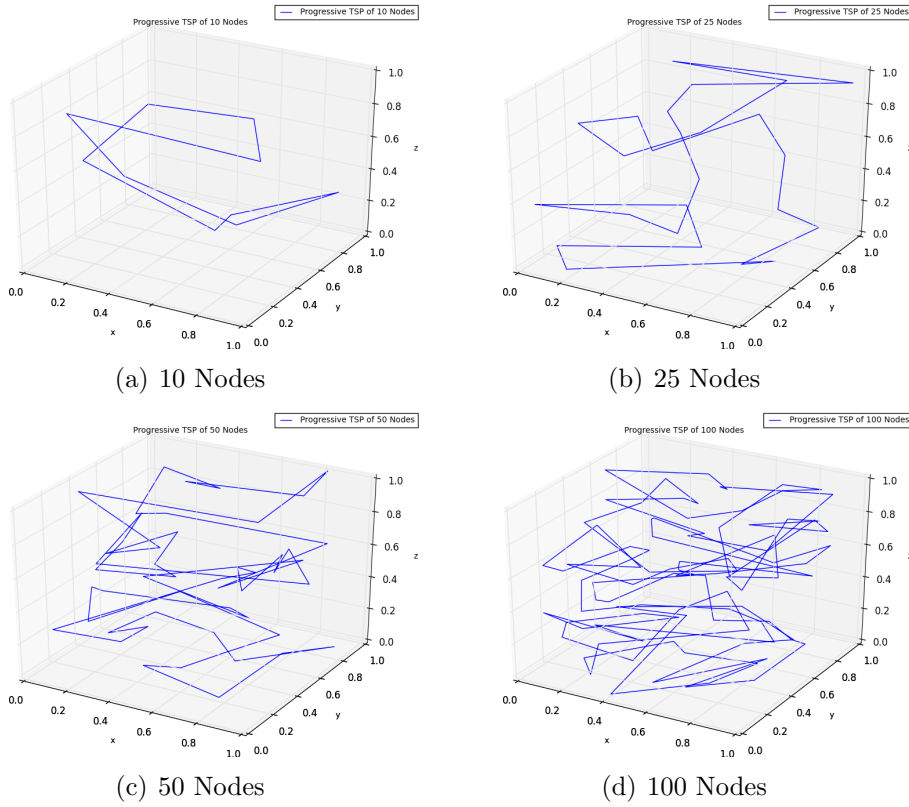


Figure 4: Exact routes calculated by travelling salesman

Figure 4 shows a number of optimal routes for varying numbers of nodes whose order is defined by the progressive travelling salesman algorithm. Though the 100 node route is difficult to see the exact routing the other routes show a logical approach to the routing problem

| | | | | |
|-----------------------|--------|--------|--------|--------|
| Number of points | 10 | 25 | 50 | 100 |
| Computation time (ms) | 52.0 | 236.0 | 591.0 | 1330.0 |
| Best route cost (J) | 115.48 | 143.62 | 202.65 | 333.73 |

Table 4: Comparison of progressive route calculation

Table 4 shows the computation time and route cost for the different routing situations. The computation time is far below that of the exact travelling salesman algorithm due to the complexity of this problem being broken down into 4 node chunks

6 Path Planning

The least energy route through a number of nodes has been defined however this route assumes that the UAV is able to turn on the spot and is not constricted by turning radius. Therefore to compute the actual energy cost of circumnavigating a route the turning radius of the UAV needs to be considered. Dubins paths are the minimum distance paths given a start and end direction.

Duubin paths are comprised of maximum rate turns and straight line segments. The following defines all the routes that are possible made up of maximum rate turns and straight line segments

- RSR - Right Turn, Straight Flight then Right Turn
- RSL - Right Turn, Straight Flight then Left Turn
- LSR - Left Turn, Straight Flight then Right Turn
- LSL - Left Turn, Straight Flight then Left Turn
- RLR - Right Turn, Left Turn then Right Turn
- LRL - Left Turn, Right Turn then Left Turn

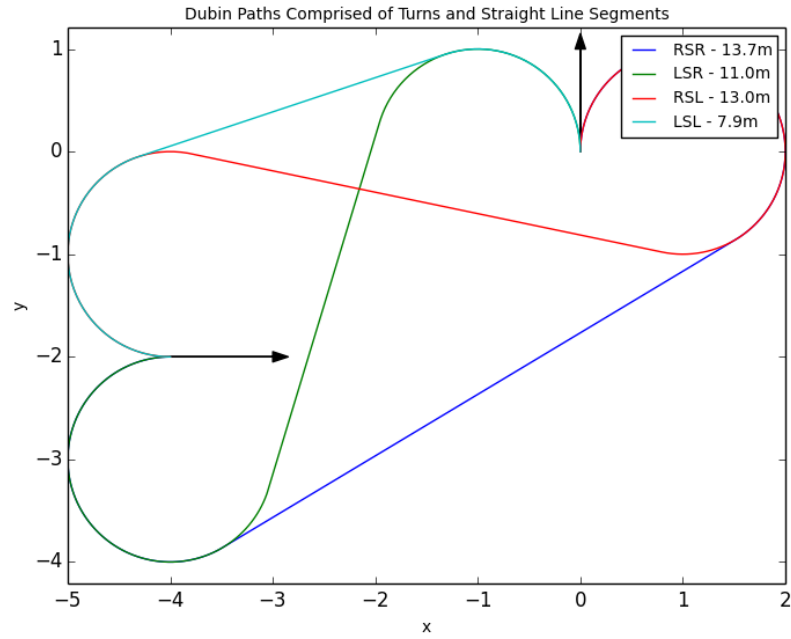


Figure 5: Dubin Paths Comprised of Turns and Straight Line Segments

Figure 5 shows the possible routes from the point $(0, 0)$ in direction $(0, 1)$ to the point $(-4, -2)$ in direction $(1, 0)$. The arrows symbolise the start and end headings and the different coloured route symbolise the different routes. In this example the routes comprise of maximum rate turns of radius 1 and straight line segment. At close proximity these routes are not always possible

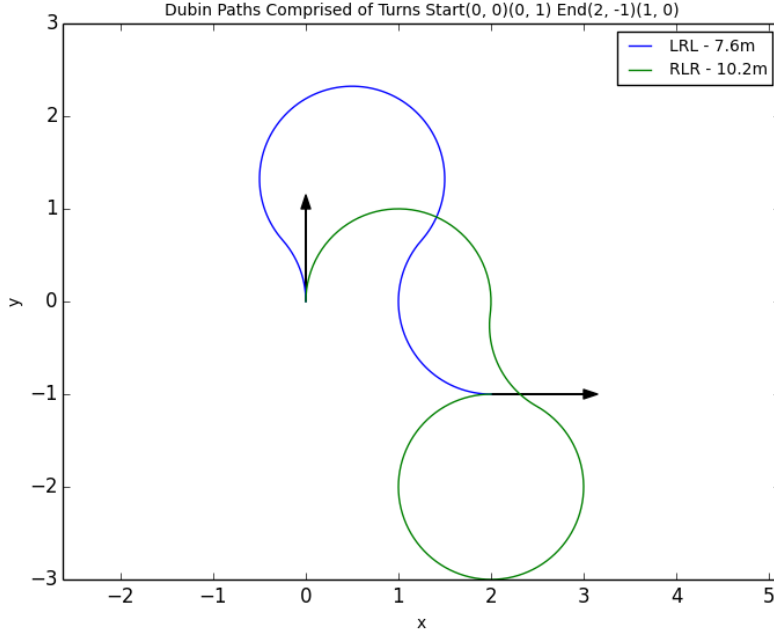


Figure 6: Dubin Paths Comprised of Turns Start(0, 0)(0, 1) End(2, -1)(1, 0)

Figure 5 shows the possible routes from the point (0, 0) in direction (0, 1) to the point (2, -1) in direction (1, 0). The arrows symbolise the start and end headings and the different coloured route symbolise the different routes. In this example the routes only comprise of maximum rate turns of radius 1. These routes are only viable when the distance between points is $D < 4r$ where D symbolises the distance.

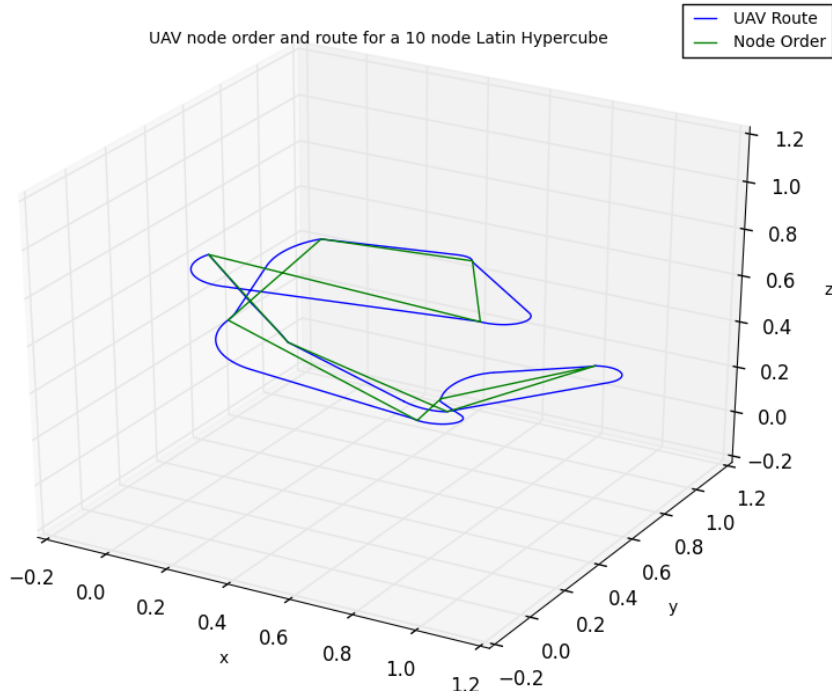


Figure 7: UAV node order and route for a 10 node Latin Hypercube

Figure 7 shows the optimal path and route for a UAV to circumnavigate a 10 node Latin Hypercube. The route is calculated before the path and then the path is calculated from the heading at each node in the route.

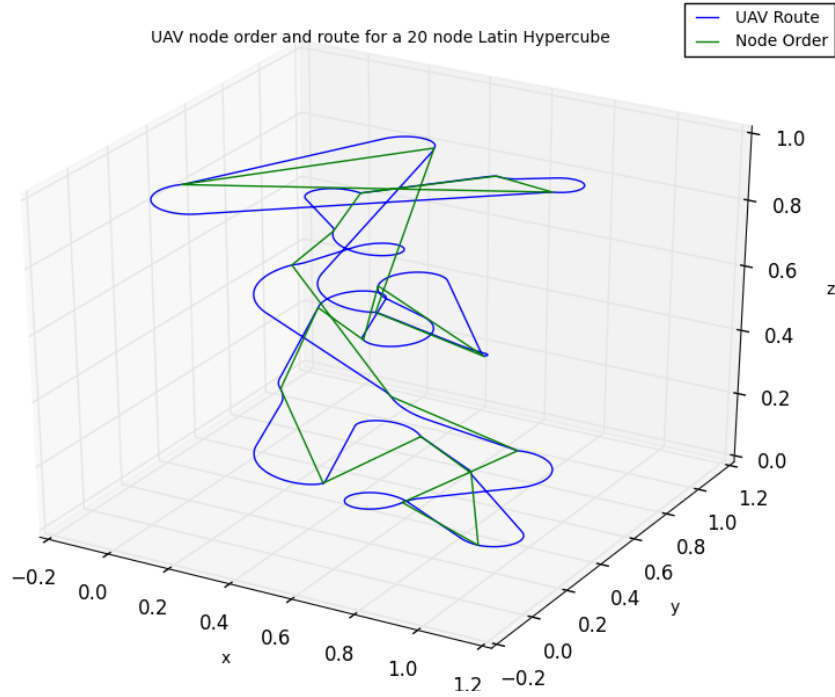


Figure 8: UAV node order and route for a 20 node Latin Hypercube

Figure 8 shows the optimal path and route for a UAV to circumnavigate a 20 node Latin Hypercube. The route is calculated before the path and then the path is calculated from the heading at each node in the route.