



Optimal trajectories for aircraft using state space search

M.C. Selvestrel & S. Goss

DSTO - Aeronautical Research Laboratory, Air Operations Division, 506 Lorimer Street, Fishermens Bend, Victoria 3207, Australia

ABSTRACT

This paper presents a method for generating optimal flight paths across defended terrain. The terrain is described by a digital map specifying the elevation in each grid cell. Threats are evaluated from line of sight calculations and the range to the threat. The path of the vehicle is constrained by the physics of motion of the vehicle. The costs associated with the vehicle's motion are expressed by a multi-objective cost function that balances conflicting objectives. The solution procedure considers both hard obstacles such as the ground which cannot be passed through, and soft obstacles such as threats which can be passed through at increased cost.

The problem was formulated using a state-space representation in which changing state, by moving the vehicle, corresponds to dynamically generating an arc of a graph. The cost associated with traversing each arc of the graph is a function of the terrain crossed, threats encountered, and the manoeuvre required to cause the change of state.

The optimal trajectory is found using the A* algorithm. Unlike previous work using A* where solution optimality was given up to gain computational feasibility, optimality of solution was preserved by using knowledge to combine compound manoeuvres (effectively limited breath first search), search over regions of the map rather than over individual grid cells of the map and intelligent pruning of inferior states.

1 Introduction

This paper presents a method for generating optimal flight paths across defended terrain. The terrain is described by a digital map specifying the elevation in each grid cell. Each grid cell corresponds to approximately 100 meter squares on the surface of the earth. Threats are evaluated from line of sight calculations and the range to the threat. The path of the vehicle is constrained by the physics of motion of the vehicle.

The costs associated with the vehicle's motion are expressed by a multi-objective cost function that balances conflicting objectives. For example, to minimise fuel usage one would like to minimise distance travelled. However, to minimise exposure to threats a winding route between hills, ridges and through valleys may be needed.

The solution procedure considers both hard obstacles (e.g. the ground which cannot be passed through) and soft obstacles (e.g. threats which can be passed through at increased cost).

The problem was formulated using a state-space representation in which changing state, by moving the vehicle, corresponds to dynamically generating an arc of a graph. The state of the vehicle is described by its position, orientation and speed. The cost associated with traversing each arc of the graph is a function of terrain crossed, threats encountered, and the manoeuvre required to cause the change of state.

The resulting graph was searched using the A* algorithm. Unlike previous work using A* where solution optimality was given up to gain computational feasibility, optimality of solution was preserved through the combined use of compound manoeuvres (effectively limited breath first search), searching over regions of the map rather than over individual grid cells of the map and intelligent pruning of inferior states.

1.1 Prior Approaches

Several approaches have been applied to determining optimal flight paths. However, few have attempted to simultaneously address all the issues outlined above.

Voronoi diagrams have been used to determine paths for aircraft in both two dimensions (Krozel⁶) and three dimensions (Aurenhammer²). The methods allow paths to be determined quickly, but fail to adequately model the physics of flight or soft obstacles. The rates of turn required by a vehicle to follow the Voronoi diagram vertices are often unobtainable in practice.

Geometric solutions using two dimensional splines (Cheng⁴), or three dimensional Free Space Representation (Rao¹⁰), have addressed the physics of motion but have limited ability to deal with soft obstacles.

Several algorithmic approaches have been applied (Mitchell⁸, Schwartz¹¹) but these often require special properties between the digital terrain map and vehicle physics. The most common requirement is that the vehicle must be able to move from its current grid cell to any of its adjacent eight cells in one step. This movement may be possible for slow moving vehicles, such as helicopters, but is unsuitable for high speed vehicles which require a large turning radius.

Control theory approaches (Menon⁷) have been developed but these tend to be computationally expensive.

Several researchers have approached the problem using state space search or graph search techniques. It is generally accepted that the A* algorithm is one of the most efficient methods for searching graphs. Unfortunately it is hindered by exponential requirements in storage space (Nilsson⁹). In order to avoid these problems, non optimal solutions have been accepted. These were found by using inadmissible heuristics (Bate³), coarse to fine refinement methods (Sudkamp¹³) or by pruning of paths believed to be inferior.

The approach presented here is based on A* search, but unlike previous work, optimality of solution was maintained through the use of compound manoeuvres (effectively limited breath first search)(4.3), searching through regions on the map rather than individual grid cells (4.4) and intelligent pruning of paths known to be inferior (4.5).

2 Modelling of the Physics

Currently the vehicle is modelled by a simple three degree of freedom point mass aircraft model. The vehicle is permitted to move for a fixed duration of time at a fixed speed. In the horizontal plane the vehicle is allowed to travel straight ahead, or to maintain a constant acceleration turn to the left or right (Figure 1). A simple terrain following algorithm is used to determine the vehicle's elevation above the terrain (Asseo¹). For the purpose of this work the vehicle speed is 300 knots (150 m/s), turns are of 3g and duration of manoeuvres is two seconds.

The vehicle's position and heading are maintained in real space (i.e. physical space). To allow the use of state space search techniques they are then mapped onto a quantised space. The quantised position is described by the 100 meter square grid cells. The vehicle heading is quantised to the nearest 22.5 degrees (i.e. there are 16 quantised headings). States within the same grid cell having the same quantised heading are considered identical. The use of this mapping prevents the accumulation of quantisation errors which often occur with state space search



techniques. The basic three manoeuvres result in a search graph with a branching ratio of three (Figure 2).

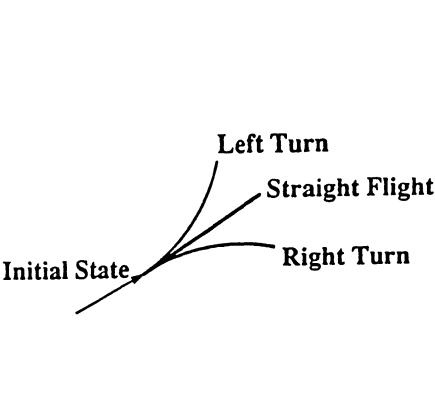


Figure 1 Aircraft Manoeuvres

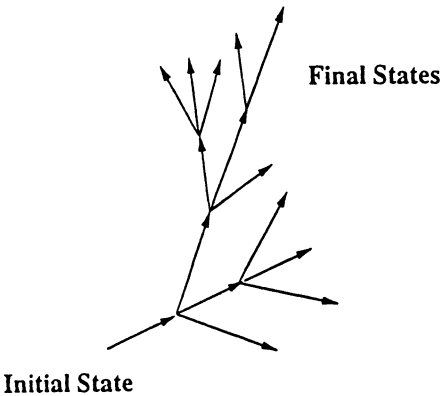


Figure 2 Generated Search Tree

3 The Cost Function

The cost associated with a particular path is determined by the integral of the effective elevation over the distance travelled by the vehicle (Figure 3). The effective elevation is calculated by multiplying the vehicles elevation by a constant. To encourage a straight line path a small constant can be used, to reduce the effect of high terrain. To encourage a path which stays low and avoids high ground a large constant can be used.

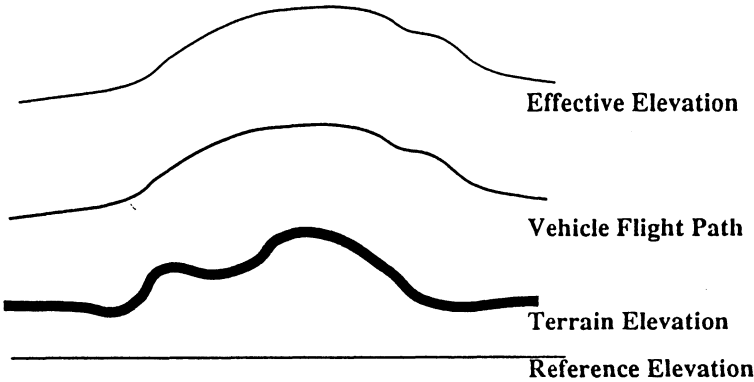


Figure 3 Cost Function Parameters

Threats are modelled by increasing the effective elevation of the terrain in the regions where a vehicle is exposed to threat. The effective elevation is used only in the cost function, it does not impact on the movement of the vehicle. Line of sight calculations and weapon ranges are used to determine if a vehicle is exposed to threats.

The suitability of specific cost functions is an operations research issue currently being studied.

4 Reducing the Search Space

As A^* is exponential in space requirements (Nilsson⁹), it is often deemed impractical to determine optimal solutions to real problems that have large branching ratios and path lengths.

For example, using the simple three manoeuvres described above over a relatively short path of 150 steps, would require, in the worst case, in the order of 10^{71} states to be expanded. Empirical studies across a number of terrain maps using A^* , with Euclidean distance as the heuristic, showed the number of states examined to be less than 10^6 , an effective branching ratio less than 1.1.

However, a problem formulation with a branching ratio of three is not sufficiently practical as it limits the vehicle's manoeuvrability. A more realistic solution would require a theoretical branching ratio of 25 to 75 to allow a wider range of quantised manoeuvres in three dimensions with changes in vehicle speed.

In order to reduce such a large branching ratio to a manageable effective branching ratio, the search space must be significantly reduced.

The search space examined by A^* can be reduced by a number of techniques. However, the solutions found may be non optimal. Such solutions can be obtained using inadmissible heuristics Bate³ or using a coarse to fine method Sudkamp¹³. On the other hand, optimal solutions can be found in a reduced search space, by using good heuristics to guide the search or reducing the path length (i.e. the number of steps needed to reach the target state) by modifying the operators used to dynamically generate the graph.

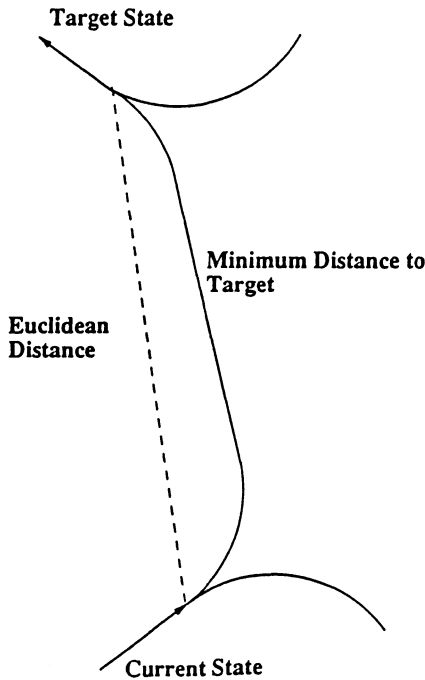
The search space can also be reduced by pruning states which are believed to be inferior. This results not only in the saving of storage space for the current state but also for those which would be future descendants. If the states pruned are known not to be on the optimal path the solution found will be optimal.

This study is restricted to methods leading to optimal solutions.

4.2 Finding Good Heuristics

Good heuristics have the effect of reducing the effective branching ratio of the search space. However, in practice it is difficult to develop good heuristics which will reduce the search space without significant computational overheads. In the problem examined Euclidean distance from the current state to the target state, was found to be the most effective heuristic.

Other heuristics developed included taking into account the radius of turn required to obtain the target state heading given the current state heading (Figure 4), and the use of a look ahead function to examine the terrain in front of the vehicle to improve the heuristic cost estimate (Figure 5). In both cases the A* algorithm examined fewer states when compared to the Euclidean distance heuristic, but the computational overheads outweighed these benefits.



The actual distance a vehicle must travel to arrive at the required target state must consider the rate of turn of the vehicle. This distance is obviously greater than the Euclidean distance.

Figure 4 Considering Vehicle Heading

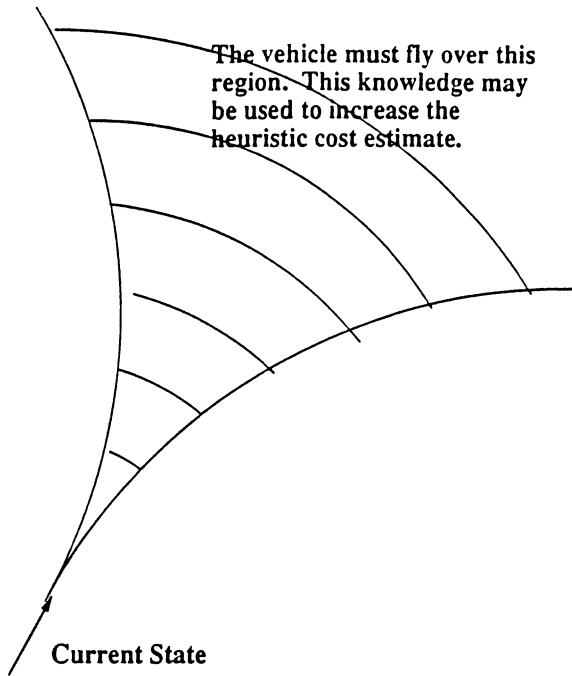


Figure 5 Using Look Ahead Function

4.3 Modifying Operators to Reduce Path Length

As the path length increases the number of states examined by A* increases exponentially. Reducing the path length will significantly reduce the number of states generated during the search. In this application the path length can be reduced by using compound manoeuvres (i.e. a sequence of several basic manoeuvres) to replace several basic manoeuvres. This sequencing of the basic manoeuvres can be thought of as limited breath first search.

The basic manoeuvres described above can be sequenced to the extent that the same final state cannot be reached via two paths in the search graph. For the basic manoeuvres described above this sequencing allows two basic manoeuvres to be completed without arriving at the same state via different paths (Figure 6). Consequently, the states generated at a depth of two do not need to be checked for uniqueness. This can be thought of as a depth first search to a depth of two.

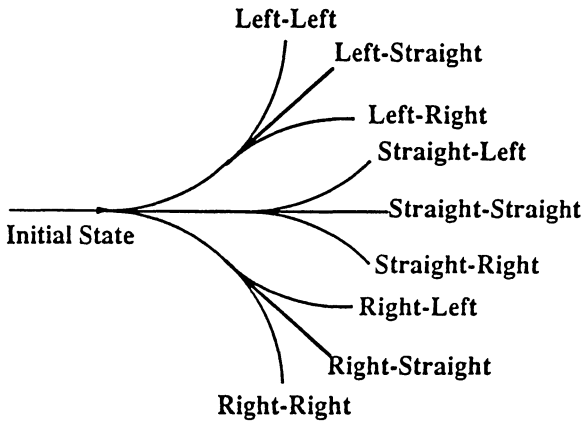


Figure 6 Compound Manoeuvres

Only the valid states at the end of the compound manoeuvre need to be maintained in the A* queues. With the basic manoeuvres described above, the effect of using a compound manoeuvre of this form is to increase the branching ratio from three to nine and halve the path length. Each step in the search graph travels twice the distance of the basic manoeuvres. The net result is to reduce the number of nodes that need to be examined.

The methods presented above can be expanded to depths where a given state can be reached via different paths in the search graph. If states can be reached via different paths, each path must be examined and the lowest cost path recorded in order to maintain optimality.

As with compound manoeuvres generating unique states the branching ratio is increased and the path length is decreased. However, overheads are encountered, because redundant paths must be examined and pruned. Experimentation showed a search depth of four provided the best compromise between storage requirements and computational overheads.

4.4 Search Through Regions

Examining the nodes expanded by A* showed that a significant amount of effort was required to generate and store nodes which lay in regions of uniform cost. Regardless of the order in which the nodes were expanded, the cost was uniform. In such cases the heuristics used were not powerful enough to differentiate

between good and inferior child states. This effectively lead to an exhaustive search in regions of uniform cost.

Since the cost in moving through such a region is constant per unit distance travelled, regardless of direction travelled, only the entry and exit states to a region are significant. The manoeuvres between entry and exit states can be described by one arc in the search graph (Figure 7).

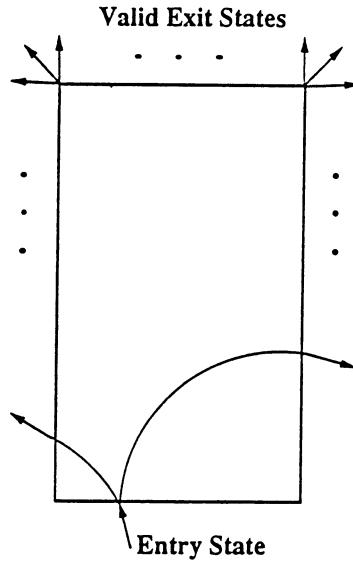


Figure 7 Search Through Regions

Searching through regions greatly reduces the path length, as only one step is required to cross a region made up of many grid cells which would otherwise require many basic manoeuvres to cross. However, for every entry point to the region there can be several exit points, thus increasing the branching ratio. A map described by regions is often called a tile world map.

Significant work has been carried out using tile world maps when restricted to two dimensional binary maps (Holmes⁵). Binary maps have the advantage of limiting the number of exit points due to the existence of no go areas, where there are no valid exit states. The use of tile world maps, in practice, is extremely sensitive to the terrain profile. There is little benefit in using a tile world representation in mountainous areas with many small valleys and ridges, whereas areas with plateaus, lakes or wide valleys can benefit greatly from such a representation.

To overcome this problem multiple strategies were used. Associated with every digital terrain map is a list of regions where this method may prove useful. When new states are generated a quick test is made to determine if it lays within a region: if it does the region is searched; if not the compound manoeuvre method is used.

4.5 Pruning

A simple modification to the A* algorithm can be used to prune paths which are not known on the optimal path and thus maintain optimality (Figure 8).

```
procedure modified_a_star
  initialise the open queue to start state
  initialise the closed queue to an empty queue
  while there are states in open
    remove the first state in open (p)
    if p is the goal
      return the solution path that lead to p
    calculate the cost from p to the goal (min_cost) [1]
    remove all states in open whose heuristic cost is greater than min_cost [2]
    generate all the children of p
    for each child of p
      if the child was already in open
        keep the child with the cheapest cost
      if the child was already on closed
        if the child on closed was reached by a cheaper path
          bring this cheaper child to open
      if the child was not in open or closed
        assign a heuristic cost to the child state
        if the heuristic cost is less than min_cost [3]
          add the child to the open queue
    put p on the closed queue
  return failure
end modified_a_star
```

Figure 8 The Modified A* Algorithm

Figure 8 presents the A* algorithm with three slight modifications. Step [1] involves calculating the cost from the current state to the target state and assigning it to min_cost. min_cost will thus hold what is currently the minimum cost for a path between the starting state and goal state. Step [2] will free the space occupied by states in open which are known not to be on the optimal path. Step [3] will prevent states with a heuristic cost greater than min_cost entering the open queue.



Optimality is maintained with this form of pruning because only states with a heuristic cost (which is an under estimate to the actual solution path cost) greater than a known solution path cost are removed.

As with search through regions, this form of pruning is sensitive to terrain profile. It performs especially well when a piece of generally low ground is found between the current state and the target state. In such cases inferior paths which attempt to explore high cost areas are quickly pruned.

However, as calculating the cost to goal takes negligible computational resources compared to queue management it is a useful technique to use all the time. A useful side effect of this form of pruning is to give A* the ability to behave as an anytime search algorithm, as the current cheapest cost path is known at all times.

5 Results

The techniques described above were applied to determining optimal flight paths over a digital terrain map approximately 100 km square (1000 by 1000 grid cells). The terrain examined contained several significant features (valleys, ridge lines, lakes, and rolling hills). Several starting and target positions were used to expose the algorithm to the terrain features from various orientations. This is required as each technique is sensitive to terrain profile. For example, a valley leading to a target may prove useful, whereas one found across a path to the target provides negligible benefits.

Table 1 summarises empirical results obtained using a SUN sparcSTATION 2 with 32 Mbt of RAM and 100Mbt of swap space. The results are averaged out over approximately 500 runs.

	Time (minutes)	Nodes expanded	Path length
Simple (Euclidean Distance)	94	1 102 456	193
Consider Heading	106	985 157	193
Look Ahead	98	979 978	193
Search Regions	54	845 987	137
Compound Manoeuvre (depth 2)	25	656 954	96
Compound Manoeuvre (depth 4)	34	567 927	85
Search Regions with			
Compound Manoeuvre (depth 4)	19	535 093	76

Table 1 Summary of Result

6 Conclusions

In practice Euclidean distance proved the most useful heuristic to determine optimal paths over digital terrain maps. More sophisticated heuristics such as considering the radius of turn required to obtain the target state heading given the current heading and the use of a look ahead function to examine the terrain in front of the vehicle to improve the heuristic cost proved computationally expensive without significantly reducing the search space.

Examination of various terrain maps show there is little benefit in using a tile world representation in mountainous areas with many small valleys and ridges, whereas areas with plateaus, lakes or wide valleys can benefit greatly from such a representation. However, the use of a dynamic representation to switch between compound manoeuvres and region search is useful.

The dynamic representation discussed above, using compound manoeuvres and search through regions has allowed optimal solutions to be found for problems which could otherwise not be solved.

As with search through regions, pruning as discussed above, is sensitive to terrain profile. However, as calculating the cost to goal requires negligible computational resources compared to queue management it is a useful technique to use all the time. A useful side effect of this form of pruning is to give A* the ability to behave as an anytime search algorithm.

7 References

- 1 S Asseo, *"Terrain Following / Terrain Avoidance Path Optimisation Using The Method of Steepest Descent"*, Proceedings of National Aerospace and Electronic Conference 1988, pp 1128-1136
- 2 F Aurenhammer, *"Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure"*, ACM Computing Surveys, Vol 23, No 3, pp 333-405, September 1991
- 3 S Bate, K Stanley, *"Heuristic Route Planning: An Application To Fighter Aircraft"*, Proceedings of National Aerospace and Electronic Conference 1988, pp 1114-1120
- 4 V Cheng, *"Concept Development of Automatic Guidance for Rotorcraft Obstacle Avoidance"*, IEEE Trans Robotics Automation, Vol 6, No 2, pp 252-257, April 1990



- 5 P Holmes, E Jungert, *"Symbolic and Geometric Connectivity Graph Methods for Route Planning in Digitised Maps"*, IEEE PAMI, Vol 14, No 5, pp 549-565, May 1992
- 6 J Krozel, D Andrisani, *"Navigation Path Planning for Autonomous Vehicle: Voronoi Diagram Approach"*, Journal of Guidance and Control, Vol 13, No 4, pp 1152-1154, November - December 1990
- 7 P Menon, E Kim, V Cheng, *"Optimal Trajectory Synthesis for Terrain Following Flight"*, Journal of Guidance and Control, Vol 14, No 4, pp 807-813, July-August 1991
- 8 J Mitchell, *"An Algorithmic Approach to Some Problems in Terrain Navigation"*, AI 37, pp 171-201, 1988
- 9 Nilsson N, *"Principles of Artificial Intelligence"*, Springer-Verlag, 1982, pp 93
- 10 T Rao, R Arkin, *"3D Path Planning for Flying/Crawling Robots"*, SPIE Vol 1195, Mobile Robots IV, pp 88-96, 1989
- 11 J Schwartz, M Sharir, *"A Survey of Motion Planning and Related Geometric Algorithms"*, AI 37, pp 157-169, 1988
- 12 Z. Shiller, Y Gwo, *"Dynamic Motion Planning of Autonomous Aircraft"*, IEEE Trans Robotics Automation, Vol 7, No 2, pp 241-249, April 1991
- 13 T Sudkamp, Courte J, *"Selection of Sub domains in Hierarchical Path Generation"*, SPIE Vol 1095, Applications of Artificial Intelligence VII, pp 426-434, 1989