Candidate Mini Project 2

Given a flight zone, no fly zones, a start point, and an end point, our algorithm finds a path from the start point to the end point that stays in the fly zone and does not go through no-fly zones. To do so, we divide the space into a three dimensional grid of block, each block a cube of sidelength grid_size, and finds a path through the grid from the block containing the start point to the block containing the end point. To do so, it only travels through blocks that are contained in the fly-zone and do not intersect the no-fly zones.

The underlying algorithm is the A-star algorithm, which is used because it is simple - though potentially not optimal. The algorithm will accurately find a path if one exists as long as the assumptions below hold, but the solution may not be optimal (we'll talk about optimality below).

Assumptions:

As long as these assumptions hold, the algorithm will find a legal path if one exists, and will properly detect if no path exists.

- Each no fly zone has non-zero volume (i.e. their top is greater than their bottom). The fly zone must have a non-zero volume as well.
- If there exist a path, there exist a pass that avoids grid block that intersects the forbidden zone or the edge of the fly zone. If this assumption does not hold, our algorithm may miss a path that includes blocks that intercect no fly zones or the edge of the fly zone and incorrectly claim that there is no path.

The first assumption seems to be implied from the way the problem was phrased. The second assumption is non-trivial, but if the grid size is sufficiently small it does not matter so much (though smaller grid size means higher computational cost for route-finding). Thus, we let the user enter a grid size parameter when establishing a map, which should be dependent on how close together the no-fly zones are, how large the aircraft is and how far it should stay from no fly zones, and how much computation power they wish to use. The default grid-size is $1/30$ times the hight of the fleight zone.

Which path does the algorithm choose?

The problem specs do not provide a condition as to which path to choose. For simplicity, we have dicided to limit the space of possible paths to paths that travel the pixelated grid in the following way:

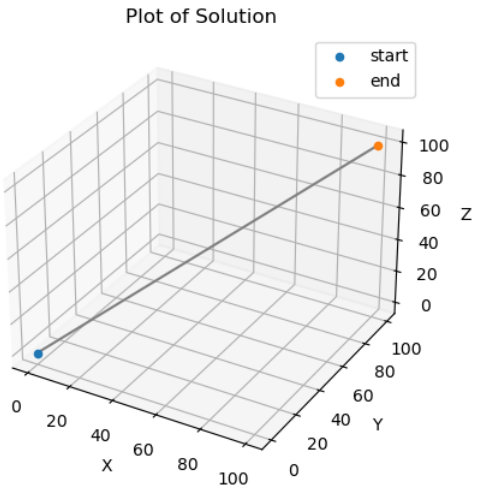
- Set up the grid coordinates so that the start point is at the origin, and is considered to be at the bottom left of block $(0, 0, 0)$.
- At each step, travel from the bottom left of the current block to the bottom left of one of the 26 adjacent block (9 above, 6 around at the same z-value, and 9 below). It avoids blocks that intersect the outside of the flight-zone, and blocks that intersect a no-fly zone.
- Once we have reaced the block containing the end point, travel directly to the end point

Note that this approach only allows paths that move in parallel to the axis, or in 45-degree diagonal paths to the axis.

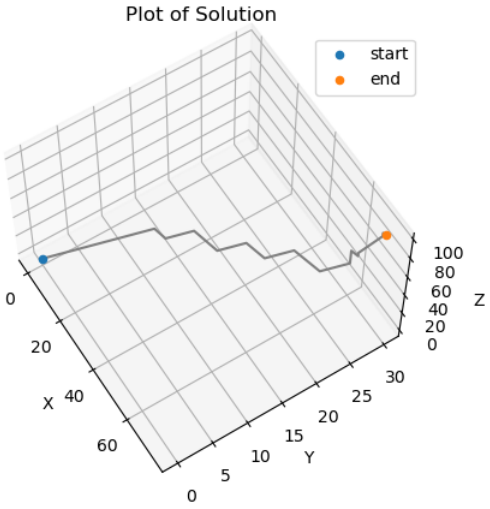
We chose to limit ourselves to such routes for simplicity: it allows us to turn this problem into a graph search problem, where each block can be seen as a node that has edges to adjacent nodes, and we seek to find the node containing the end-point from the node containing the start point. To implement the graph search we used a variation of the A-star algorithm[1]. While this algorithm does not guarentee that it will find the shortest solution (in the restricted class of legal solutions), it approximates it well and is fairly efficient. Also note that we have tweaked the parameters of the algorithm (in particular, the "estimated cost function" of each node, which estimates how long the minimal path containing each node is), to further prioritize searching nodes that are closer to the end point, which usually makes the algorithm run significantly faster but may provide solutions that are farther away from optimal.

[1] A simple extension to Dijkstra's algorithm that prioritizes searching nodes that are assumed to be closer to the end point first, and is thus very useful for graph-search in graphs that represent a Eucledean space.

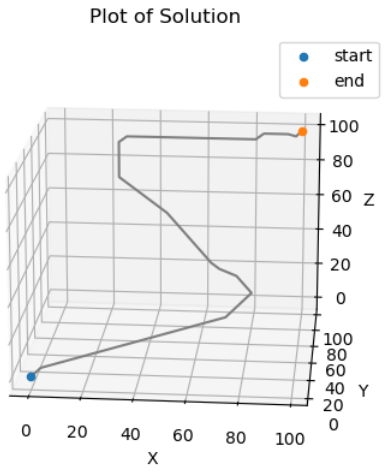
The following example is for the path found from the point (0,0,0) to (100, 100, 100), both inside a convex fly zone without no-fly zones. Note that this is indeed the shortest path:



The next example is of a path found from the point (0, 0, 0) to the point (75, 30, 99), both inside a convex fly zone without no-fly zone. Note that the solution follows "45 degree diagonals" and not the true shortest path between the points.



Lastly, the following example finds a path between the points (0, 0, 0) and (99, 99, 99), both inside a convex plane. Not graphed are two big no-fly zones that the path has to wrapp around: one consisting of the rectangle ((0,0), (0,100), (70,100), (70,0)), with z-coordinates ranging from 10 to 20; the other consisting of the rectangle ((30,0), (30,100),(100,100), (100,0)) with z-coordinates ranging from 80 to 90.


Plot of Solution
start
end
100
80
60
40
20
0
Z
100
80
60
40
20
0
Y
0
20
40
60
80
100
X