POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

EXECUTIVE SUMMARY OF THE THESIS

Noise minimization in the choice of take-off trajectories

TESI MAGISTRALE IN COMPUTER ENGINEERING – INGEGNERIA INFORMATICA

AUTHOR: LUIGI V DE FATICO

ADVISOR: FEDERICO MALUCELLI

CO-ADVISOR: MATTEO CRIPPA

ACADEMIC YEAR: 2020-2021

1. Introduction

The presence of an airport nearby a residential area is a well-known cause of disturbance for the neighboring population. The aircrafts transit generates a continuous noise that can lead to serious damage for the inhabitants.
During the years, the authority's commitment to reduce the noise pollution generated from the aviation industry led to the definition of a balanced approach covering different aspects of the problem.
This thesis aims to contribute to those efforts with an innovative algorithm that can generate a take-off trajectory minimizing the noise perceived on the urban areas surrounding the airport.
The preliminary implementation [1] results in a promising algorithm with some limitation on the context modelling that affects the accuracy of the minimum path found. It considers a flat model in which the aircraft is assumed to be at ground level for the duration of the execution.
Furthermore, the performances of the algorithm are affected by the implementation choices made that turned out to be not optimal for this case. For these reasons it has been defined to be still unsuitable for use in a real case.
The following analysis is oriented to the expansion of the model considered by the algorithm and to the improvement of its performances, to allow a better fitting of the generated trajectories in a real case.
The study produced as complementary output a functioning Python program that implements the innovations proposed. It has been used to better analyze the previous implementation of the algorithm and to test and validate the changes introduced.

2. State of the art

The algorithm is based on two different optimization approaches. A generative phase that allows to create a completely new trajectory and an optimization phase that can optimize a previously existing trajectory, based on a non-linear programming algorithm [2].
The generation of a new trajectory is based on an implementation of A*, a graph traversal and path finding algorithm.

The graph is initially composed of only the start and target nodes and then it is dynamically generated during execution by expanding the existing nodes.
Each time a node is visited, a predefined number of new nodes are generated using a fixed step and a set of congruent turning angles.
The number of successors to be generated at each iteration is arbitrarily decided by settings, as is the maximum tack angle.
The cost to reach each new state is calculated and assigned to the new node through a process of sampling that considers the distance from elements called *sensitive receptors* located in the concurrence of residential areas. The cost of each segment in a context having R sensitive receptors and sampling S points per segment is given by the (2.1), where P is the aircraft engine power, w_i is the weight of each receptor, and r is the distance between the sampling point and a sensitive receptor.

$$\sum_{S}\sum_{R}\frac{P * w_i}{4\,\pi\,r^2} \qquad (2.1)$$

Moreover, A* estimates the remaining cost to reach the target node. This estimate is called a heuristic and it allows to expand the most promising nodes first, by modifying the priority list that regulates the order of expansion of nodes. It is calculated with the same sampling process used for the segment's cost but it uses a dummy configuration to minimize the value: the aircraft heading and the goal direction are aligned and all the sensitive receptors are placed behind the plane.

3. Analysis

The analysis of the algorithm focused on the generative phase, highlighting it as the most critical phase of the previous implementation.

3.1. Performance of the heuristics

A first step of the analysis considered the impact of the heuristics on the overall performances of the algorithm. We can consider Dijkstra's algorithm as a special case of A* in which the heuristic is always zero. Comparing the two algorithms it turns out that the used heuristics had a very bad impact on the time needed to compute a trajectory.

What emerged is that the amount of visited nodes using A* on the considered cases is about seven times higher than the amount obtained using the Dijkstra's algorithm.
Figure 1 and Figure 2 show a comparison of the tree dynamic generation in a study case using A* and Dijkstra's algorithm.
The increase over Dijkstra's algorithm occurs in cases where A* has room to expand into external regions of the search area, where the current implementation of the heuristic requires paths to be protracted much longer before they are discarded. Indeed, the actual heuristics tend to push the search away from the sensitive receptors instead of approaching the paths to the goal node.

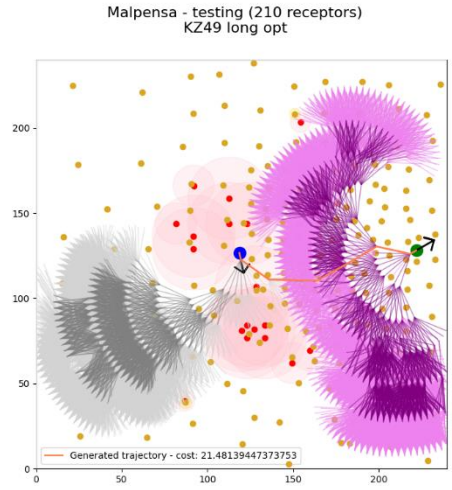

Figure 1: Execution with A*

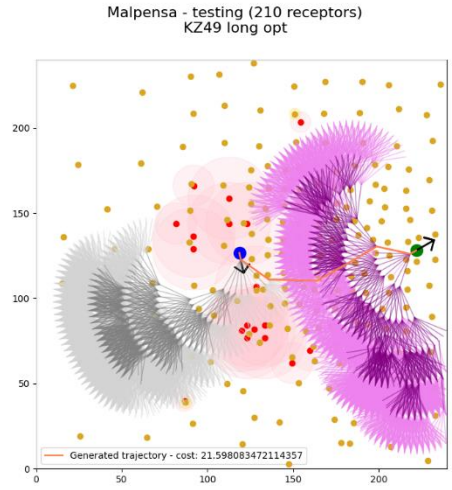

Figure 2: Execution with Dijkstra's algorithm

The problem, as well as an increase in the number of nodes visited, lies in the excessive calculations

needed to obtain the value of the heuristic for each node basing it on a sampling process. That lead the required time to increase by about eleven times.

3.2. Bidirectional search

Bidirectional search, although it improves the performances of the algorithm in its current state, presents some issues.
The basic principle is based on the alternating expansion of two opposite trees, one starting from the first node and the other starting from the goal. The search ends when the trees meet each other. First, when the search area is narrowed the expansion of the trees evolves in a different way causing the trajectory to be different from the one found in the absence of restrictions. This introduces a grade of uncertainty in the solution found.
Moreover, the bidirectional search has been found incompatible with an introduction of the altitude in the algorithm. The incompatibility is caused by the fact that the final position altitude is not known a priori. Indeed, it is not possible to start the generation of the opposite tree from the goal node, being its altitude unknown.

4. The new approach

4.1. Minimum distance

It has emerged the need to define the minimum distance between two nodes in the presence of a minimum radius of curvature. This distance can be calculated with Dubins' paths [3], which allow its computation using simple equations. Dubins' paths have found several applications in the improvement of the algorithm.
A first use of them is in the computation of the heuristic. For each node visited, it is computed by creating an ideal case in which the aircraft goes through the minimum distance from the new node to the target node, moving away from the receptors as if it had them all behind it. By using the real minimum distance instead of the cartesian distance we can calculate the heuristics in a more proportionated way. Although a performance improvement has been observed, it is not sufficient to justify the presence of the heuristic.
A second use of Dubins' paths is found in controlling the overlap between two nodes. This is a problem because of the possible combinations of distance and direction of the two nodes but still necessary to decree the termination of the algorithm. Checking that the Dubins' distance is below a given threshold turned out to be an efficient way to exclude paths that meet the goal in an unfeasible way.

4.2. Third dimension introduction

The introduction of the third dimension is key for the correctness of the solution found. Figure 3 and Figure 4 show two trajectories generated with same input but travelling at different steady altitudes.

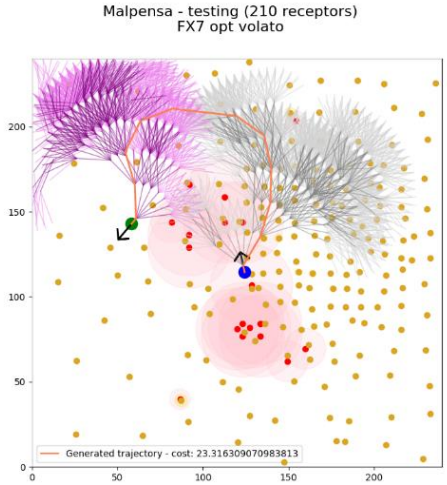

Figure 3: Trajectory generated at ground level

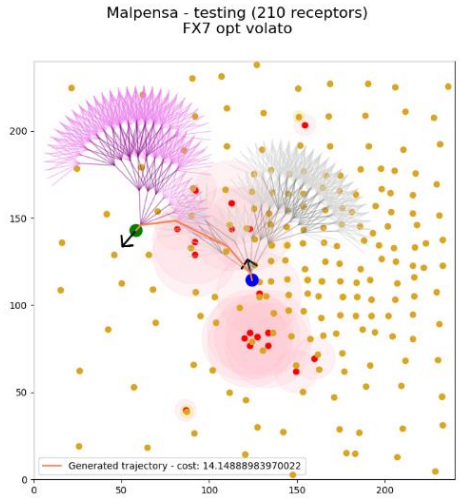

Figure 4: Trajectory generated at 3000 m

The flat model gives too much weight to advanced segments of trajectories that have instead a greater distance from the sensitive receptors considering the increment of altitude as the path progresses. The main problem in the introduction of the third dimension is that the computational complexity of the algorithm grows exponentially with the number of levels of different heights considered at each step, due to the additional degree of freedom given to the algorithm.
Taking advantage of the fact that the sensitive receptors are all located on the ground, the third dimension was introduced by considering only the maximum altitude increment for each step. This resulted in a constant growth of the altitude which agrees with the take-off regime of the aircraft. The introduction of the constant growth led to a major improvement of the accuracy of the trajectory generated. The paths are modified to avoid sensitive receptors in the first part where the aircraft is closer to the ground and move directly towards the target when the altitude becomes higher.

4.3. High-pass filter for noise

The computational cost required to reach a solution depends linearly on the amount of sensitive receptors located in the studied configuration. However, the weight of sensitive receptors in the total cost decreases quadratically with respect to distance from the aircraft. For this reason, the impact on the cost of sensitive receptors at long distances is almost zero.
A high-pass filter has been implemented considering the contribution of sensitive receptors within a certain radius from the aircraft position, corresponding to the distance necessary to reach a given threshold I_{min} of sound intensity, and discarding the others. The result of the filter implementation led to a major decrease of the time required to compute the cost of each node which dramatically increased the amount of nodes visited per second without affecting the shape of the path found.

4.4. New heuristic

The necessity to change the heuristic arises from the fact that the one used up to now is counterproductive in terms of execution time. Two strategies have been tried:

- Make the heuristic more proportional to the distance from the node to the goal.
- Simplify the calculation to reduce the time it takes to evaluate it.

The first strategy has led to slightly modify the heuristic, so that it uses the Dubins' distance instead of the Cartesian distance in the dummy configuration. The sampling process is left unchanged. As shown in Table 1, this solution led to a more proportional distribution of the heuristics value that reduced the total amount of visited node in almost all the configuration tested. However, the time taken to compute the heuristics kept execution times very high.

Rates	3 - 5 angles	7 - 9 angles
Nodes	82%	49%
Time	86%	48%

Table 1: Rates of nodes visited, and time spent using Dubins' distance and Cartesian distance

In comparison to Dijkstra's algorithm, execution times of this new heuristics is still on average five times higher, even if the number of nodes visited is halved.
The other strategy led to the simplification of the heuristic computation, calculating it as the (4.1), where R is the number of receptors, I_{min} is the minimum noise threshold, and L_{samp} is the sampling step length, over the Dubins' distance from the node to the goal.

$$h(n) = \frac{Dubins_i}{L_{samp}} * R * I_{min} \qquad (4.1)$$

This implementation turned out to be ineffective because of the small value taken by heuristics compared to the total cost of the path, although the visited node per second was comparable to the Dijkstra's algorithm ratio.

4.5. Directional noise function

The function used to estimate the noise intensity at a certain distance is indicated in the (4.2).

$$\frac{P * w_i}{4\,\pi\, r^2} \qquad (4.2)$$

It assumes that the noise generated by an aircraft engine propagates as a sphere, expanding equally in every direction. This assumption is based on a strong approximation of how sound behaves in

this context. A parallel thesis to this project allowed the creation of a new noise model based on the direction with respect to the source [4]. The model has been integrated in this work and led to an increase of the generated trajectories accuracy.
The function's implementation slightly affected the time performance of the algorithm thanks to the introduction of the high-pass filter for noise that limited the number of times that the function is called in total.

5. Conclusions

Two case studies based on the Malpensa airport, situated in Lombardy (Italy), have been tested to compare two versions of the algorithm implementation output to the official take-off route. The results show a decrease of the perceived noise for the new implementation mainly because of the introduction of the third dimension that drastically reduced the length of the generated trajectory.
Expansion of the model allowed us to generate paths that more plausibly minimize the perceived noise at ground level. The three-dimensional model introduced, in addition to the directional noise model, resulted in a substantial modification of the generated path, achieved by an increase in the realism of the choices made by the algorithm. The application of Dubins' paths has made it possible to discriminate more accurately between viable and non-viable paths.
As far as the performances are concerned, using Dijkstra's algorithm was found to be more efficient compared to A* due to the complex context that cancelled the effects of the heuristics. The introduction of the high-pass filter made it possible to reduce the impact that the number of sensitive receptors has on the sampling process, decreasing the total time taken by the algorithm. Bidirectional search was eliminated as it was not compatible with constant altitude increase.
The new configuration offers better performance in generating a trajectory in a general context. The algorithm still has issues, like the expansion of the graph towards areas that move away from the goal. The current development does not yet allow the algorithm to be used in real-world contexts, this thesis is intended as a starting point for further development of the project.

References

[1] Gullo, M. (2021). Air Trajectory Optimization for Noise Reduction. Milano: Politecnico di Milano.

[2] Python. (s.d.). Minimize method SLSQP. Tratto da docs.scipy.org: https://docs.scipy.org/doc/scipy/reference/optimize.minimize-slsqp.html

[3] Dubins, L. E. (1957). On Curves of Minimal Length with a Constraint on Average Curvature, and with. American Journal of mathematics, 497-516.

[4] Zepeda, P. (2022). Aircraft sound modeling through simulation for route optimization. Milano: Politecnico di Milano.