**Tracjectory optimisation for terminal flight stage of Urban air mobility vehicles**

A. Barr

Department of Aerospace Engineering, University of Bristol, Queen’s Building, University Walk, Bristol. BS8 1TR. UK.

**aBSTRACT**

**Keywords:** Trajectory Optimisation, eVTOL, Urban air mobility, Aeroacoustics, Noise

**iNTRODUCTION**

With the continuous development of technologies beneficial to Urban Air Mobility vehicles such as improvements to electrical propulsion and batteries, many concept vehicles of different configurations have either been announced as in development **[1]**, all aiming to provide a new form of green transport to tackle problems in the urban environment. Due to the potential of widespread adoption of these UAM aircraft in urban areas around the globe, particular attention must be paid to the noise footprint of these aircraft. Indeed, the effects of noise pollution on health are well known **[2]** and the practice of implementing not just technological improvements **[3]** to aircraft, with a particular focus on reducing noise from their propulsion, to reduce their footprint but also tactical and operational considerations, for example Heathrow’s dual runway operations **[4]**. It is important that these same considerations are applied to the emerging field of eVTOL UAM vehicles to ensure they can be implemented into urban communities without causing adverse health and living conditions. Research to this effect has been conducted on reducing aeroacoustic footprint through design optimization of the vehicles themselves, for example through reducing propeller/rotor noise **[5-9]**. However, research into optimising trajectories on the noise footprint of these vehicles is more limited but cover important observations about some specific configurations. This paper aims to build on top of *Clarke M, et al* **[11]** which used SUAVE, to perform a detailed assessment and comparison of different eVTOL aircraft configurations. The aim of this paper is using this methodology in an optimisation problem with the goal to minimise the noise impacting an area around a UAM landing area.  This is done by utilising SUAVE, which is primarily a library for python specifically for conceptual level aircraft design, however as part of its toolkit, there are many useful features, for example optimisation tools and mission design tools as well as the flexibility to allow for a trajectory optimisation problem to be defined. For a SUAVE optimisation problem to be carried out, the different aspects of the problem need to be thoroughly established.

**SUAVE WORKFLOW**

Every problem within SUAVE follows the same generic structure. The vehicle, analysis to be performed, and mission all need to be defined. In the case of an optimisation problem, the optimisation variables, constraints and objectives also need to be defined. SUAVE works by firstly building the vehicle, and then setting up the analysis defined for the vehicles defined, which is then attached onto the specific mission to be analysed. Therefore, each of these aspects had to be carefully considered to ensure that the results gathered were accurate.

**APPROACH AND LANDING PHASE TRAJECTORY**

**Defining The Trajectory**

The trajectory for the UAM on approach which would be optimised for this paper was chosen with inspiration from the traditional visual circuit. The reason for this choice, is that currently, due to the emerging nature of UAMs as a vehicle class, there is no specific approach methodology for the piloting of these vehicles and that the tradition visual circuit is in essence what forms the approaches for all aircraft types. It also allows flexibility in defining parts of the trajectory which can change as part of the optimisation process. The standard visual circuit along with the terminology for each “leg” is shown in Figure 1.

With the basic legs contained within specifically the approach part of this circuit, namely the downwind, base and final legs, this idea was built upon to add more nuanced legs to the trajectory to capture more of the behaviour we could expect from a UAM. This was done by adding a vertical landing leg, as well as a transitional leg where the transition from forward flight to hover would be achieved. This final defined approach path is shown in Figure 2.

The final trajectory consists of an initial approach leg, 2 nautical miles long, which has a fixed altitude and speed. This is then followed by a turn onto the base leg, which is 1 nautical mile long, while maintaining the same altitude and speed. Next, there is another turn onto the final approach, and the altitude starts reducing at a set descent rate, until a set altitude. Once at this altitude, the aircraft is now transitioning to a hover, while maintaining altitude. The last leg is then a vertical landing from that altitude. This trajectory was then defined in SUAVE using the in-built methods.

**Variables to Optimise**

With the trajectory established, the next step was to select appropriate parts of the trajectory to modify as part of the optimisation. These would intuitively include variables which would have a direct effect on the noise output from the vehicle for example the forward flight speed of the vehicle but would also include variables which would affect the physical routing of the aircraft, for example the initial altitude of the approach. It was decided to include the turn angles between the downwind, base and final legs as part of this, to allow more variation than just a simple “box” circuit as this would allow our analysis to consider more interesting approach routes, for example straight in. The full list of variables that would be changed is contained in Table 1.

|  |  |  |
| --- | --- | --- |
| Property | Legs Affected | Reasoning |
| Approach speed, from initial approach up until hover | Initial Approach, Base, Final, transition | Speed will have a big effect on noise, due to increased aerodynamic noise, and increased engine noise due to higher engine throttle. |
| Descent rate during the final leg | Final | Similar to speed, the rate of the descent will affect the aerodynamic noise, and also the throttle setting and hence the engine noise |
| Hover Height | Hover | The higher the hover, the more time at a higher throttle setting and therefore this could affect the footprint |
| Turn angle to base | Base | Effects the routing of the aircraft over the ground |
| Turn to final | Final | Effects the routing of the aircraft over the ground |

**Mission Setup Within SUAVE**

The definition of this trajectory is achieved within SUAVE using a Class file, Missions.py, which sets up the mission flown by the vehicle through different segments and attaching the analyses to be preformed to said mission. The segments are defined using preset options, such as constant speed and constant altitude legs, and then the settings of those legs are set within the code, for example the altitude, air speed, and length of the leg. The class file then returns the mission, with the attached analyses. Unknowns, such as the throttle settings, are solved later when the problem itself is evaluated as part of the analysis.

**AIRCRAFT FOR ANALYSIS**

**Background**

“Urban Air Mobility” as a term encompasses a large variety of different designs, which have different handling characteristics, number of engines, etc. The main categories currently emerging in the market are: <**INSERT HERE WITH REFERENC>**. In theory, the methodology defined in this paper could be used to analyse any variation of these designs given that the design could be accurately defined within SUAVE. This could be useful in future to observe if there is a difference in how approaches should be flown for the different design types, in order to minimize noise. However, this falls outside of the scope of this paper and for our purposes a multi-rotor type vehicle is used for the analyses.

**Aircraft Definition Within SUAVE**

As part of the SUAVE workflow, the aircraft needs to be “built”. This then allows for proper aerodynamic and noise analysis. This is done by defining set properties utilizing the SUAVE library. This includes defining the geometry of the vehicle’s fuselage, rotor positions, number of engines, battery capacity etc. For this paper, a slightly modified version of the multirotor vehicle used in <reference Clarkes PHD> research paper is used. This provides a simplistic multirotor design to allow for a general analysis of a multirotor type vehicle to be performed. As mentioned previously, this could be easily changed to perform the same trajectory optimisation on another specific UAM if desired, however this is sufficient for the purposes of carrying out a general trajectory optimisation.

Practically, this is implemented through defining a class file, Vehicles.py, which defines all the relevant parameters of the vehicle, along with any configurations of that vehicle. Configurations allow variation of things such as flap angles, so that different setups of the same vehicle can be used for different parts of the mission. In our case, different rotor tilts of 0, 3 and 5 degrees were defined. These would be used for standard forward flight, transition to hover, and hover respectively.

**NOISE ANALYSIS**

**SUAVE Fidelity-One**

In order to accurately minimize for noise, the different relevant sources of noise needed to be simulated. For this paper, this achieved using SUAVE’s Fidelity-One noise simulation module. This uses a Frequency-Domain approach to solve the Ffowcs-Williams and Hawkings governing equation of acoustics. This consists of Harmonic noise components, solved using Hanson’s formulation, which is noise due to the rotor blades movement through the air, Broadband noise, which includes contributions from several phenomena, such as the interaction of a blade with a preceding blades wake known as Blade Wake Interaction. SUAVE also captures noise due to Self-Noise, however currently it only models Turbulent Boundary Layer-Trailing Edge noise, which is due to the shedding of turbulence on the surfaces of the blade. Research by <reference the PHD paper> shows however that omitting the other components of Self-Noise, would still lead to relatively accurate prediction of noise, unless at high speed as in these conditions, predictions of some components would be too low.

**Implementation within SUAVE**

In terms of implementation in the code, the Fidelity-One module is straight forward to implement. Similarly, to how the aircraft and trajectory were setup, a Class file, analyses.py is used to define the analyses to be used. This is done by passing through the vehicle and configurations of said vehicle and defining which analyses withing SUAVE to perform. In order to carry out an accurate simulation of the vehicles flight through this approach phase, the vehicles weights had to be analysed. This was done through the SUAVE weights\_eVTOL method. This then allowed analysis of the aerodynamics through the aerodynamic fidelity zero method. Then, the noise was analysed using the noise fidelity one method. This method works by setting up a grid of ground microphones, around the starting location of the aircraft. The number of these microphones, as well as the resolution of the microphone placement is defined.

**Computational Issues**

As mentioned above, the number of microphones and resolution of the placement is defined for the noise analysis. During coding, it was found that these settings had a major impact on memory use of the simulation. This meant that increasing the simulation above quite a small number of microphones would not be possible with the current setup without running the computer running the simulation our of memory. The computer in question has 32Gb of RAM. As a result of this, an alternative way to run the simulation was designed as to optimise memory use and allow for a more accurate (higher resolution) of microphones.

The solution consisted of running the same simulation several times, however only considering a small segment of the overall grid per run of the simulation. This is depicted graphically in figure x. This was initially developed using a quadrant system, where the overall grid was split into four, and then the simulation was run four times with the results of each grid then being assembled at the end for the overall noise capture by all the microphones. Further in addition to splitting the grid, the grid was also centred about the end of the trajectory, the landing point, rather than the start point. This made it easier to ensure that the whole trajectory remained within the grid.

This functioned well, with the trade-off being that the simulation would now take longer to run, due to the increased times the simulation needed to be ran. This would not be an issue for single run cases, however in the case of optimisation, where the simulation would need to be run potentially thousands of times in order to find a minimum, this would take an unreasonably long amount of time for the scope of this project.

**Quantifying Overall Noise to Optimise**

As part of any optimisation problem, there needs to be an objective function to minimise or maximise. In this case, the general goal of the problem was to reduce overall noise disturbance for the area surrounding a landing site for UAM vehicles. To complete the optimisation problem, there needed to be a way to represent this general goal as one numerical value that the optimiser would attempt to minimise.

Firstly, it was decided to represent the sound output recorded by each microphone in the grid using the A-weighted system. This is a method for representing sound pressure levels in a way that accounts for relative loudness as heard by the human ear. This makes it suitable for this task, as our goal is to reduce the impact on people living around a landing site. **(More detail on A weighted system)**

SUAVEs Fidelity-One method already outputs the A-weighted SPL for each microphone in the grid, for each time step as part of its simulation. Once we have each microphones A-weighted SPL for each timestep of the simulation, the last step is to represent these results in a way representative of the noise experienced by people living near this landing site. This is achieved by taking the RMS of the recorded A-weighted SPLs of each microphone for each time step of the simulation. RMS is used as this provides a good measure of the overall sound energy emitted by the aircraft over the duration of the flight. Another method considered was to use the average of the maximum recorded A-weighted SPL for each microphone, however this was decided against as this would not capture information regarding the duration or frequency content of the noise.

**OPTIMISATION**

**Optimisation Within SUAVE**

Optimisation in SUAVE is handled using a class called Nexus. This stores all information regarding the optimisation, for example input variables, constraints and objective information as well as the information regarding the analysis to be carried out which is defined as described in earlier chapters of this paper. The Nexus links together all subfunctions of the problem and allows them to interact and communicate through the use of aliases. SUAVE then packages the problem defined and described by the nexus, in order for it to be handled by a third-party optimiser, for example SciPy (a scientific python library) optimisers. This provided us with a few options for which type of optimiser we could select for our problem.