ERT-S GITHUB README

Filename:	Github README
Prepared by:	Richard Ducker

1 INTRODUCTION

The GitHub for the ERT-simulator can be found here. It regroups the entire code that is up to date for the ERT Matlab simulator as well as some non-simulator related data. The goal of this document is to summarize the use of this GitHub and help any newcomers to navigate it and find the useful codes quickly.

2 REFERENCE DOCUMENTS

Table 2-1 Reference Documents

Ref	Description	Doc. Number	Issue
[RD01]	[RD02] Acceptance procedure.pdf REDV test	S3-D-SET-1-2	1.0

3 CONCEPT EXPLANATION

The GitHub is full of information and very different codes, so it can seem scary at first. This document aims to help in understanding it quickly. Therefore, the first part of this document is dedicated to finding the basic code to simulate a rocket flight with the ERT Matlab Simulator.

3.1 Main

If all you are looking to do is simulate a rocket flight and get basic data such as apogee height, max velocity or acceleration, the code you want to use is Main3D.m or Main3D_SFT.m (same code just different rocket configurations) that can be found in 2018_SP_FLIGHT_SIM-CONTROLLER/tree/master/Simulator_3D (or see mindmap at the end of the document: Figure 1).

Things to watch out for while using this code:

- The configuration of the rocket and environment: these are the two main parameters of the code and are the only parameters that are input, therefore you need to make sure they represent the correct rocket model and area of launch. This especially includes the names of the motors that are given in the Rocket text file. The thrust curves in these files need to represent the correct launch.
- The get_google_maps function (line 146) might not work: this is because google updated their policies. We are working around a solution to this, but it is not important as such, it just gives a cool satellite image for a trajectory plot.

3.2 Operational Code

The code described here is to be used by the operational team of the SI subsystem on a regular basis but more importantly around the time of launch and test flights. The different scripts can be found in 2018_SP_FLIGHT_SIM-CONTROLLER/tree/master/Simulator_3D, just like the Main. The scripts are Confidence_Interval_landing.m, CheckDrift.m ,Best_landing.m and stability_analysisR02.m.

- Confidence_Interval_landing will give the user an estimate of the landing zone of the rocket by making simulations with different wind parameters given in the Environment file and different motor parameters to give maximum randomness in the simulations. Things to watch out for: not give too much deviation in the motor and wind parameters because the simulations don't run properly.
- Stability_analysisR02: gives the user the different stability features of the rocket at different times for two scenarios: nominal and a worst case. The script gives results at rail departure time and at max speed. The worst case is defined by a rail departure velocity of 20m/s which is the minimal velocity at this time for a safe launch. Things to watch out for: the inertia of the rocket given in the Rocket text file seems to influence the results quite a lot, so it is important to have the correct input.

3.3 OpenRocket (OR)

For more detailed use, it is important (especially for the operational team) to check results with other simulators, such as OpenRocket. This open-source program is much more intuitive that our Matlab simulator, and therefore gives a better overview. The folder named OpenRocket in the GitHub contains the necessary files to set up a simulation on OpenRocket: rocket files and motor files. There is also a code called OpenRocketReader.m (see Figure 1) that can take data created in OpenRocket and make a Matlab figure with it. This is useful for creating direct comparisons between OR data and ERT-S data.

3.4 Work in progress

As of June 2021, the work in progress is:

- The atmospheric model (stdAtmos_new.m) is more precise but not very fast to use, it needs to be better integrated into the simulator
- There are several issues with the simulator that do not compromise the results but need to be addressed: the rotational moment of inertia, the norm of the quaternions, and others that are discussed in slides uploaded to Simulations -> Meetings -> Other Meetings (all in BL2)
- The simulator is being transitioned towards Python because of the open access this language has compared to Matlab

3.5 Details on simulator in case of problem

The easiest way to explain the code is of course to start with the main3D. Here, as discussed before, the type of rocket and environment is given to the code. The first step if there is an issue is to check that all the values given in the text files are the right ones.

The next important concept to understand (and therefore look for problems) is the Simulator3D class. This is where all the real calculations are done. As is clear in the first few lines of the class definition, the Simulator3D contains the Rocket, the Environment, some temporary values (tmp_xxxx) used to crate plots, and most importantly, methods.

In the Simulator3D class, the most important methods start around line 580 (might have changed since this document was written) and are separated based on what part of the launch they are simulating. For example, the first method, RailSim, simulates the rocket's flight from takeoff to the exit of the launch rail. Then follows FlightSim and the different parachute simulations.

Simulator 3D class, methods

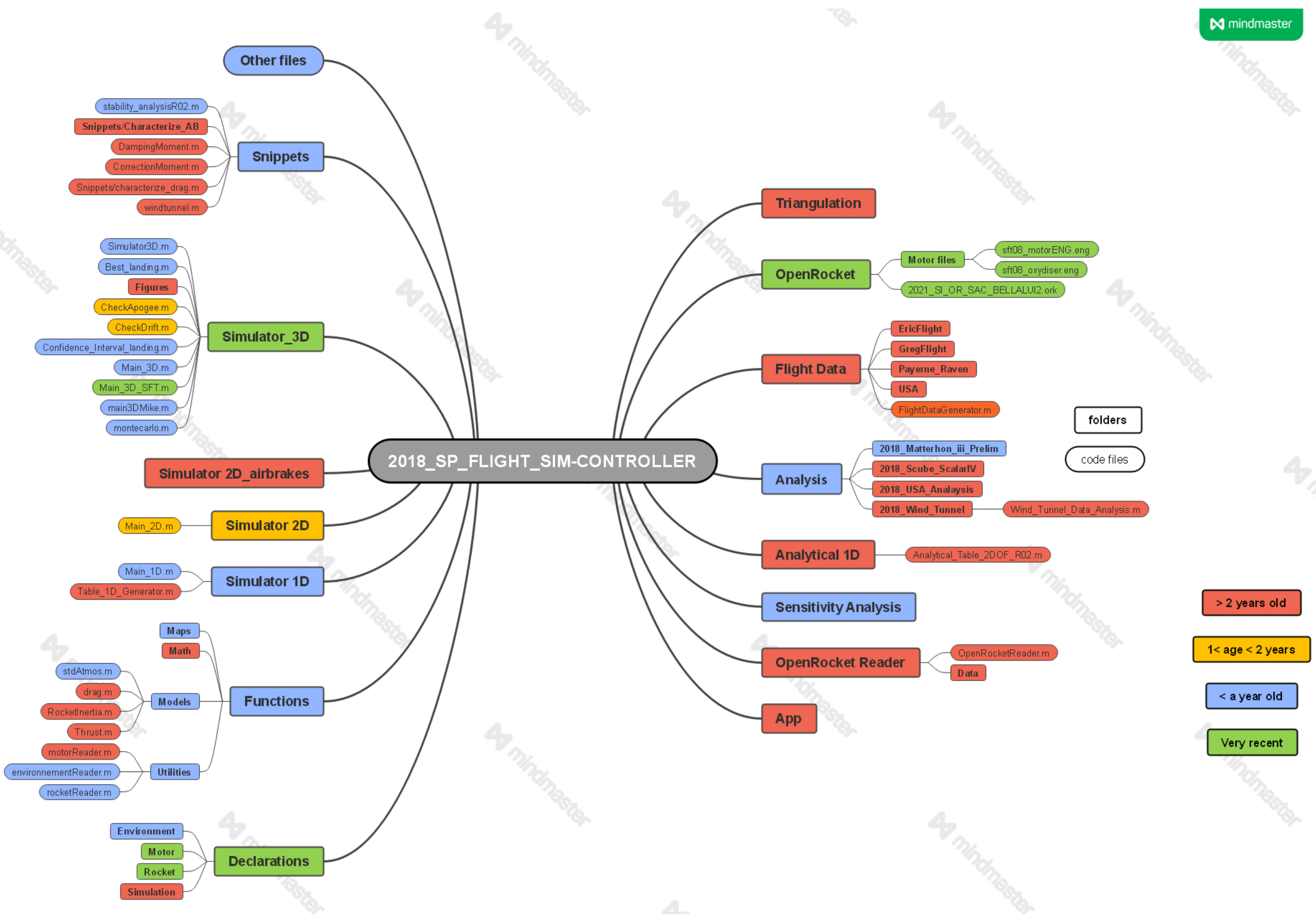

Figure 1: Mindmap of the GitHub