Łukasz Fulek
April 8, 2016

Flight recorder simulator

The aim of this project was to simulate the flight parameters recorder. It has to work all the time during the flight, store the parameters for an inspection and have the possibility of presenting the data after landing, e.g. by drawing plots.

The application has two modes of operation:

- RECORD MODE: data is collected every time interval (given by the user) from take-off to landing. Additionally, the flight is simulated in this part (it will be discussed later).
- READ MODE: read data collected by the recorder and present them by drawing plots.

RECORD MODE

As it was mentioned above this mode has two parts: recording the data and simulating the flight. Both are performed every time interval Δt (the same for both), which is given by the user. Because of that, two threads are created. If one wants to replace the simulation of the flight by e.g. real plane, one thread can be excluded.

1. **Simulation of the flight**
   The flight simulation uses a very simple pseudo-random number generator. It has been assumed that the maximum speed of the plane is $v = 900\frac{km}{h}$, maximum altitude $h_{max} = 10\,000\,m$. First the distance to fly is generated between 1000 and 10000 km. The distance to take-off and land x is assumed to be one tenth of the entire distance. Also the flight number is generated and it is between 10000 and 100000. The acceleration/deceleration value a for take-off/landing is calculated as:

$$a = \frac{v_{max}^2}{2x}$$

   and the angle of the plane θ is:

$$\theta = \tan^{-1}\frac{h_{max}}{x}$$

   The flight consists of three parts: take-off, cruise and landing. Below is the description of each of those steps.

   a. **Take-off/landing:** in each step (every given Δt) those variables are calculated:
      - **Velocity:** $v_i = v_{i\text{-}1} + a\Delta t$
      - **Distance gone:** $s_i = s_{i\text{-}1} + v\Delta t\cos\theta$
      - **Altitude:** $h_i = h_{i\text{-}1} + v\Delta t\sin\theta$

   b. **Cruise:**
      - **Distance gone:** $s_i = s_{i\text{-}1} + v\Delta t$

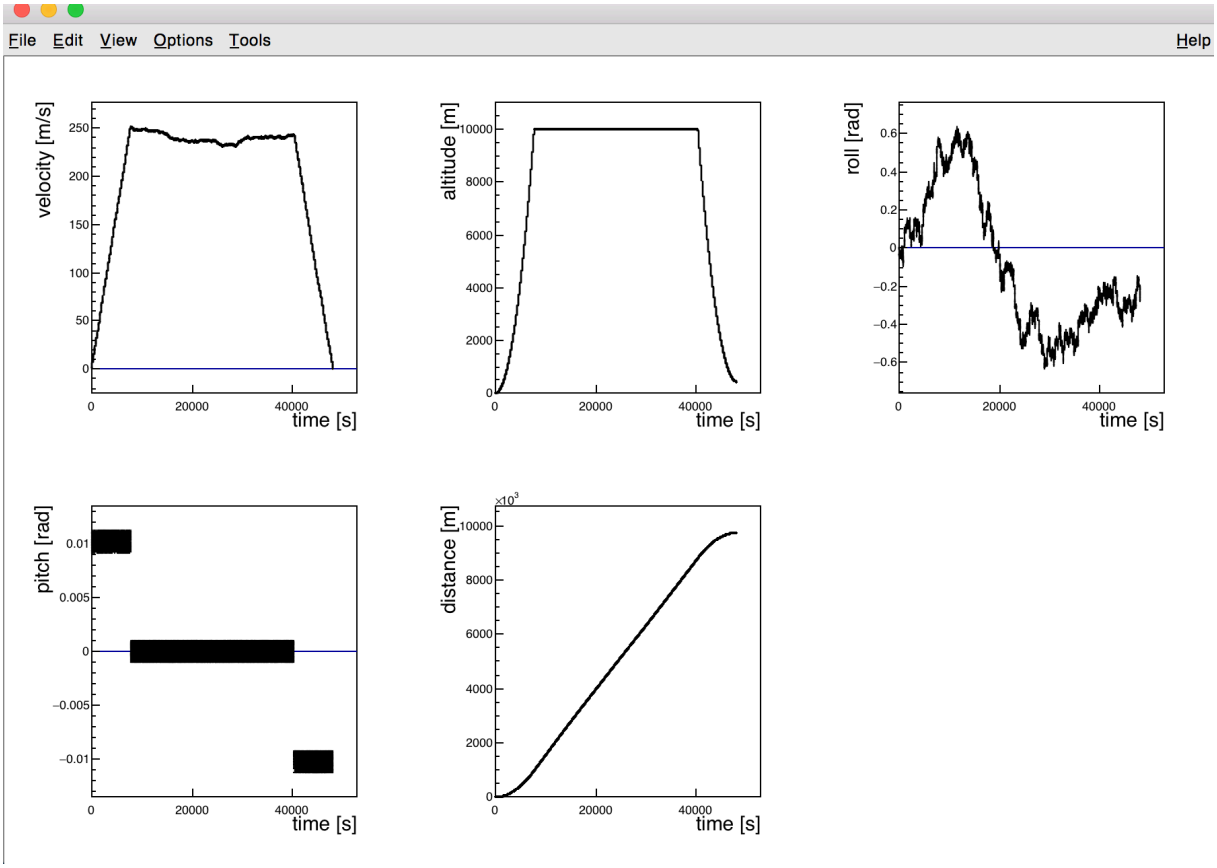
Roll and pitch is calculated for each of these steps. Roll is assumed to be an angle between -0.005 and 0.005. The pitch can vary about $\frac{1}{10}\theta$ from the center value: θ for take-off/landing, 0 for cruise.

2. **Flight parameters recorder**
   The data from the simulation (separate thread) are received as a dictionary and next written by the recorder to the text file. It is also required and checked that in one single file there is up to 10000 events. At the end of the flight, after being packed in tar.gz file, all text files are removed from the disk.

READ MODE

In the reading mode the user is required to give the flight number. Next the application is looking whether tar.gz file for that flight exists. During unpacking it is checked and validated if the file contains dictionaries for each time interval and whether the values are float types. If not, the wrong time interval is omitted and user will see an alert in the terminal. After that, the plots of velocity, distance gone, altitude, roll and pitch as a function of time are drawn using PYROOT package.

Below is a sample output from the reading mode.



PROGRAM SCHEME

- mgr.py: a manger code, which controls the execution of the program
- mode.py: checks the mode in which program should run
- modes.py: enum for READ/RECORD, NEW PLANE/QUIT
- myThread.py: a class inheriting from Thread with additional functionality, i.e. keyboard interruption
- variablesIntoDB: enum for variables written to the file
- flightSimulator.py: simulate the flight
- dataStore.py: creates/removes txt/tar.gz files, searches for files and reads them
- inputReaderValidator.py: asserts if the input parameters make sense
- runDaemon.py: creates two threads for recording and simulating, counts number of events in a single file, main class for record mode
- drawPlots.py: draws plots, main class for read mode.

SUMMARY

The flight parameters recorder simulation has been presented. The application fulfills all the objectives of the project, i.e. simulating flight, recording the parameters and presenting the data after the simulation. During gathering the data and simulating the flight, two threads are created for using the program with different input, e.g. real plane. The main downside of the program is simplified simulation of the flight. This part should modified in the future by adding more realistic description of the flight.